A Source Routing Solution to Non-Transitive Connectivity Problems in Distributed Hash Tables

Ivan Dedinski, Andreas Berl, Alexander Hofmann, Sebastian Heglmeier,
Bernhard Sick and Hermann de Meer
University of Passau , Faculty of Computer Science and Mathematics
94030 Passau, Germany
{dedinski, berl, hofmann, heglmeie, sick, demeer}@fmi.uni-passau.de

Acknowledgement: This project was partly funded by the German Research Foundation (Deutsche Forschungsgemeinschaft - DFG), contract number ME 1703/4-1, by EPSRC, contract number GR/S69009/01, and by the EURO-FGI - Network of Excellence, European Commission grant IST 028022.

Abstract

Distributed hash tables are popular third generation P2P protocols which are well understood in theory. These protocols usually assume that every node in the overlay is able to exchange messages with any other overlay node. However, this assumption is not always true for real-world networks, including the PlanetLab or the entire Internet. In these networks, the non-transitive connectivity phenomenon is experienced, in which some overlay nodes are able to exchange messages with a certain node and others are not. This turned out to be a serious problem, particularly for structured P2P overlays. Non-transitive connectivity issues were mainly ignored by P2P research for a long time, but have been intensively discussed recently. This paper suggests a new measure for the degree of non-transitive connectivity and presents a comprehensive, source routing based solution, to overcome non-transitive connectivity problems in distributed hash tables.

1 Introduction

Distributed hash tables (DHTs) are third generation P2P protocols forming structured overlays which have several advantages compared to other P2P paradigms. Their theory is well understood and most of their properties are provable. DHTs provide a lookup service and are able to find a route from a source node to a target node within a limited number of overlay hops. Particularly, DHTs are implementing a tradeoff between overlay management load and information location load. Therefore, every peer maintains a fixed, comparatively small number of connections to other overlay nodes, predefined by the DHT structure. These connections, especially the ones to direct logical overlay neighbors, are essential to the stability and efficiency of a DHT. To establish an overlay connection, two nodes have to be able to exchange messages using the underlaying network. If they are able to exchange messages they are said to have *connectivity*.

However, recent research [3, 5] has revealed that real-world networks, involving the PlanetLab [8] and also the global Internet, tend not to provide connectivity among all pairs of overlay nodes. Instead, in these networks the nodes experience *non-transitive connectivity* (NTC), which is defined as follows: Let A, B, and C be three overlay nodes. Let A have connectivity to B and let B be have connectivity to C. If A has no connectivity to C, the three nodes exhibit NTC in this situation.

The NTC phenomenon is a serious problem for DHTs. Their provable features rely mostly on the assumption that all pairs of nodes have connectivity, thus problems are arising in practice. Freedman et al. [3] have named several concrete DHT problems in networks with NTC, involving invisible nodes, routing loops, broken return paths, and inconsistent roots. The problems have been explained in detail and a set of specialized algorithms has been suggested to overcome each single problem. A comprehensive solution, however, is still missing.

As a measure for the degree of NTC, Gerding and Stribling [5] suggested to analyze all possible triples of overlay nodes in a network, and divide the triples that experience NTC by the total number of triples. They found out that approximately 9.9% of all considered triples in PlanetLab experienced NTC. In [3], additionally *transient* NTC has been analyzed, in which node triples temporarily experienced NTC. These time periods, affecting

about 2.3% of all considered triples in PlanetLab, occur for many reasons, e.g. link failures or BGP routing updates. Unfortunately, this node triple based measure does not precisely predict the impact of NTC effects to the DHT, as explained in Section 4.

In this paper a comprehensive solution is presented to overcome the above mentioned problems caused by NTC for DHTs, instead of fixing each of the problems separately. The remainder of the paper is structured as follows: In Section 2 source routing is applied to DHTs, solving the NTC problem in static situations. Additional heuristic strategies, needed to overcome NTC in more realistic dynamic networks, are presented in Section 3. Section 4 introduces a new measure for NTC and argues why this measure better predicts the impact of NTC. A statistical and a measurement evaluation of the suggested solutions based on the new measure is given there. In Section 5, differences of the presented approach to other solutions are outlined. The paper is concluded in Section 6.

2 Applying Source Routing to DHTs

Source routing is a well known routing mechanism being used in *mobile ad-hoc networks (MANETs)*, for instance. MANETs can be considered as networks with a high degree of NTC, because only physically neighboring hosts within radio transmission range have connectivity. Fixed networks require different route discovery mechanisms since the degree of NTC is usually lower than in MANETs.

A *source route* (SR) contains the addresses of the source overlay node, the destination overlay node, and all *intermediate nodes* (INs) in the overlay, on the route from the source to the destination. The SR is carried in the header of a packet. Thus, packet forwarding is done by looking up the next hop towards the destination in the header. An SR is represented in this paper by [A:BC$\cdots$:Z], where A is the address of the source node, Z is the address of the destination node, and B, C, ... are the addresses of the INs. If an SR involves one or more INs it is called a *indirect route*. An SR without INs is called a *direct route* and represented by [A::Z] (A and Z have connectivity in this case). An SR consists exclusively of *segments* which are pairs of nodes having connectivity. For simplicity, connectivity has been defined to be bidirectional in this paper. With this precondition, an SR consisting of bidirectional segments also enables a bidirectional message exchange. In practice, communication paths may exist which are not bidirectional. Although these unidirectional paths are not considered in this paper, they can potentially be used by the suggested mechanisms and will be researched in future work.

The NTC problems in DHTs are caused by the exchange of addresses, which are possibly not reachable from other overlay nodes. When receiving such addresses, nodes get an inconsistent view of the overlay. Thus, avoiding the exchange of addresses which are not reachable from everywhere in the overlay is the key to eliminate problems resulting from NTC in DHTs.

The suggested solution in this paper is to exchange SRs in DHTs, instead of exchanging potentially unreachable addresses. A node A which has connectivity to node B, for instance, constructs the SR [A::B] and exchanges it with other nodes. At the first glance this seems rather similar to an address exchange, because source and destination addresses are usually carried in the header of a packet. However, only the information of node B being reachable from node A is expressed by the SR, not B being reachable from anywhere else. Using this mechanism, every node exclusively exchanges information that can be used by other nodes.

To successfully apply source routing mechanisms to DHTs, the SRs have to by assembled in a consistent way. An SR containing a source node A, is at first usable by A only. When other nodes learn this SR from A, they have to assemble a new SR with this information. If, e.g., node A has connectivity to B, it constructs the SR [A::B]. When node C receives this SR from A, it has to assemble a new SR to B. Since C has obviously an SR to A, e.g. [C::A], it assembles the new SR [C:AA:B]. Applying a loop elimination mechanism to this SR, it is simplified to [C:A:B]. A is now the IN for packets routed from C to B. A more complex scenario is shown in Figure 1. Node A receives an SR [C:DE:F] from node C. The SR from node A to node C is [A:B:C]. Node A can now assemble a new SR [A:BCDE:F] from A to F. In [4] similar mechanisms are suggested to run DHTs in MANETs.

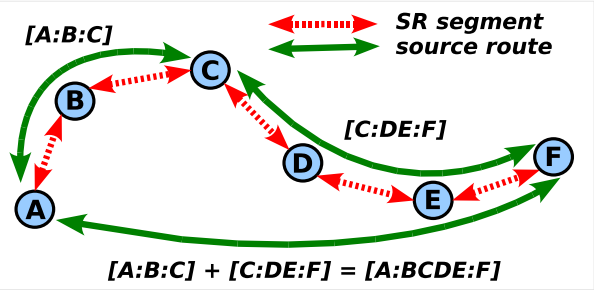

Figure 1: Node A assembles the SR [C:DE:F] and the SR [A:B:C] to a new SR [A:BCDE:F].

Note that exchanging SRs instead of addresses does not affect the advanced operation of a DHT, SRs fulfill a similar task as addresses do. Thus existing DHTs, can be modified to use SRs without imposing major changes to their core algorithms, preserving their stability, scalability, and converging properties.

With the assumption of a *static* network scenario,

not considering leaving nodes, failing nodes, the load of nodes, or transient NTC, the so far suggested mechanisms (SR exchange and SR assembly) are sufficient. Modified DHTs operate in this static scenario as if all addresses were reachable from all overlay nodes.

In practice however, networks tend to have *dynamic* properties, in which the above mentioned assumptions are not valid. Further problems have to be considered. SRs might contain several INs, since they have not been shortened, yet. Nodes may become overloaded if they are used as IN in too many SRs. Furthermore, SRs are getting invalid if INs are leaving the overlay. DHTs will find alternative routes in this case, but only if the overlay is not already partitioned. The only source for direct routes so far is the bootstrap process, which itself is usually not very flexible (e.g. the user configures a bootstrap node for his application). As long as no additional direct routes are explored, only direct routes specified at bootstrap time are available. In the worst case this leads to unacceptable overlay topologies, e.g., all SRs using a single bootstrap node as IN. Thus, the first problem to be solved is to shorten the assembled SRs. The second problem is to apply load balancing among the INs. And the third problem is to increase the direct route diversity. Section 3 suggests solutions to these problems.

3 Source Routing under Dynamic Network Conditions

Two known heuristic approaches are applied to DHT environments. They solve the problems mentioned in Section 2 that arise in dynamic environments. *Route shortening* and *load balancing* aim at creating mainly short or direct SRs, in which the additionally created source routing load is balanced evenly among all possible INs. During their operation the two mechanisms also actively discover new direct routes.

The route-shortening mechanism reduces the number of INs in a newly assembled SR by detecting connectivity (shortcuts) between pairs of INs in the SR. In the DHT context route shortening can be applied in a straightforward way. Let a node A have a newly assembled, loop-eliminated route [A:B:C] to node C. The SR [A:B:C] might still not be the optimal route, since node A and C possibly have connectivity. Thus, node A actively probes the route [A::C] to shorten the SR. If no connectivity is available, the longer route [A:B:C] is used. Figure 2 shows a more complex scenario with three INs. A assembles the new SR [A:BCD:E]. Before using it, it probes (in this order) [A::E], [A::D], and [A::C]. In this example node A has connectivity to node C, thus the resulting shortened SR is [A:CD:E].

An SR is not exchanged until the shortening procedure is completed, in order to exclusively exchange short SRs. This is important, because the overhead for probing routes gets higher with increasing SR length. Note that an SR shortened with this heuristic is not necessarily the globally shortest SR to a destination but is a good approximation to it in fixed networks having relatively low degree of NTC (cf. Section 4).

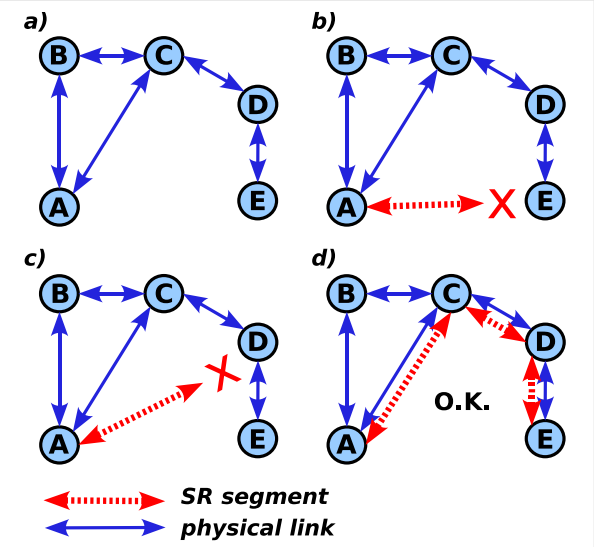

Figure 2: Shortening of [A:BCD:E] by node A: a) Connectivity of the nodes. b) Node A fails to reach node E directly. c) Node A fails to reach node D directly. d) Node A succeeds to reach node C directly, the source route [A:CD:E] is usable.

Similar route-shortening mechanisms have been applied to other source routing systems before [6]. In networks with a high degree of NTC (e.g. MANETS) its effectiveness is limited, because the probability to find a shortcut within an SR is low. In networks with a low degree of NTC (e.g. fixed Internet) however, the suggested route-shortening approach is highly effective, as shown in Section 4.

The load-balancing mechanism (to balance the load on INs) suggested in this paper relies on locally available information. In more detail, a load-balanced SR of any node X has three important properties: First, the SR contains at most one IN. Second, if the SR contains an IN, the IN is an overlay neighbor of X. Third, if the SR contains an IN, X has connectivity to the IN. If a node has an SR in its routing table, which does not fulfill these conditions, it attempts to load-balance it. Therefor it first chooses overlay neighbors as candidates, to which it has a direct route. Then, it defines a random ordering of the candidates. It probes, whether the first candidate can be used as new IN in the SR (it has to have connectivity to the destination). This procedure is repeated after a certain time period (e.g., the keep-alive interval) until all available candidates are probed or a suitable candidate has been found. Meanwhile the

not yet balanced SR is used. If there is a low degree of NTC in a network, the probability is high to find a suitable candidate in short time. Figure 3 shows an example of applying the suggested load-balancing heuristic. Node A has routing-table entries [A::B], [A::C], [A::D], [A:B:E], and [A:J:F]. The SRs [A:B:E] and [A:J:F] are considered for load balancing. [A:B:E] is already load balanced, since it fulfills all preconditions. The SR [A:J:F] on the other hand, is not load balanced, because the IN J is not an overlay neighbor of A. A will attempt to substitute J with one of the directly reachable overlay neighbors B, C or D, in random order.

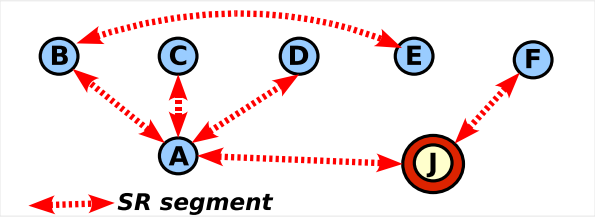

Figure 3: Node A attempts to substitute intermediate node J with nodes B, C or D.

DHTs usually distribute the overlay neighbors of a node fairly among all overlay nodes, leading to (ideally) distinct routing tables at every node. Thus, the suggested IN load-balancing technique nicely fits in existing DHT load-distribution mechanisms.

Available load-balancing properties of DHT hash functions have been used before to balance the load on INs. FreePastry [1] uses a similar mechanism, for instance, but in a limited way (cf. Section 5). In this work this approach is generalized and in Section 4 a comprehensive performance evaluation is provided.

Note that without the presence of NTC, both route-shortening and load-balancing approaches do not affect the DHT. An enhanced DHT operates as if it had not been modified at all in this case. No overhead is produced in terms of additional packets or increased packet size. The SRs then only consist of the source node and the destination node address, which are usually contained in a packet anyway. Solely direct routes are used without presence of NTC.

The two suggested heuristics operate the better, the lower the degree of NTC in a network is. As already mentioned, the degree of NTC in fixed networks is at a relatively low level. But even parts of the network experiencing higher degrees of NTC are still supported by source routing, as shown in Section 4. However, the SRs might get longer for these network parts, since the suggested route-shortening and load-balancing heuristics might perform less efficient in this case.

As a proof of concept, source routing has been integrated into CHORD [13], which is a popular DHT implementation. The resulting architecture is shown in Figure 4 and evaluated in Section 4. In the CHORD routing table the use of network addresses has been replaced by the use of SRs. In addition three new components have been added, *Router*, *RouteAssembler*, and *RouteManager*. The router component is responsible for

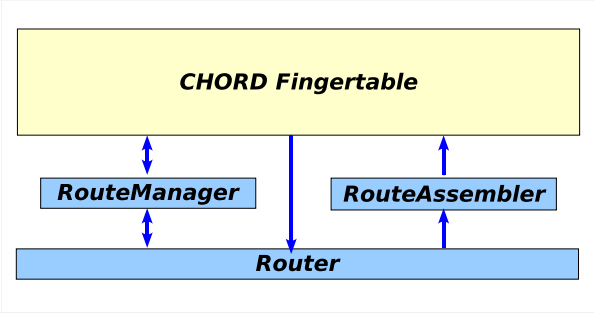

Figure 4: An enhanced CHORD architecture.

packet forwarding according to the SR. Every bypassing packet is processed to decide whether it must be delivered to the local node or forwarded to the next node in the SR. The routeassembler component is in charge of the SR assembly as described in Section 2. Every new route passes the RouteAssembler component before being inserted into the CHORD fingertable. The routemanager component is responsible for the route-shortening and load-balancing tasks. While route shortening is done before the first usage of the route (as described before), the more time consuming load balancing is done asynchronously.

4 Evaluation

This section evaluates the effectiveness and overhead of the suggested heuristics and provides results of a statistical and measurement analysis under different NTC conditions. For this study, the CHORD-based architecture presented in Section 3 has been implemented and evaluated in the PlanetLab environment.

For the statistical analysis a new measure for the degree of NTC is necessary. The triple-based measure, suggested by [5], was derived from the definition of NTC. However, this measure has some disadvantages. On one hand, the missing connectivity of a node pair is counted more than once, if the node pair is involved in several NTC triples, e.g. (ABC), (ADC), (AEC)... . On the other hand, a missing connectivity between two nodes is not counted if it is not involved in a node triple with NTC. This means, the triple-based measure considers the connectivity of some overlay node pairs to be more important than the connectivity of others. But, this does not always reflect the impact of NTC to DHTs. An example for this is the broken return path problem [3] which may be caused by any overlay node pair without

connectivity, independent of being part of a triple with NTC. Therefore, this paper suggests an NTC measure based on the connectivity of overlay node pairs in the network. The coherence of a node pair based measure and the triple based measure is as follows: If a non-partitioned overlay network contains a triple of nodes with NTC it also contains a node pair without connectivity (proof is trivial). If a non-partitioned overlay with more than two nodes contains a node pair without connectivity, it also contains a node triple with NTC (proof skipped, due to lack of space). This allows a node-pair based definition of NTC: *Two nodes cause NTC in a non-partitioned overlay network containing more than two nodes, if they don't have connectivity.* The degree of NTC is then defined as $\frac{NP_{nc}}{NP_{total}}$ where NP_{nc} is the number of all overlay node pairs without connectivity and NP_{total} is the total number of overlay node pairs.

Further properties concerning single nodes can be derived from this measure: Let X be a node of a DHT overlay network, establishing connections to its predefined overlay neighbors (e.g., fingers in CHORD). In a physical network with a certain degree of NTC, the *node transitivity* q of a node X is defined as $\frac{L_c}{L_{total}-1}$. L_c is the number of all nodes in the network having connectivity to X and L_{total} is the total number of nodes. This means q is the probability of a node X to have connectivity to node Y, which is randomly chosen from $L_{total} - 1$ nodes. In any DHT every node establishes connections to a prescribed set of overlay neighbors with a certain ID. The ID is determined by the DHT's hash function, e.g., SHA or MD5, to ensure a load balancing among the nodes. In CHORD, e.g., a node with ID id attempts to establish connections to the nodes with IDs $id + 2^0, id + 2^1 \cdot \cdot, id + 2^{log(H)}$, where H is the highest value in the ID space. Determining the IDs of the nodes using a (good) hash function leads to a set of neighbors of a node X, which are randomly chosen from all $L_{total} - 1$ nodes. The analysis in this paper assumes, that there is no stochastic dependency between the connectivity of any two node pairs. In this case a DHT node X has connectivity to any of its overlay neighbors with probability q. A high degree of stochastic independence is given in networks which provide many alternative overlay routes e.g. the PlanetLab or the Internet. This assumption has been confirmed by measurements in PlanetLab.

Commonly, q is not equal for all nodes X. A node hidden behind a firewall, for instance, might have much less connectivity, than other nodes in the network have. Thus, for the analysis in this paper, the value of q is randomly distributed among all nodes with an unknown density function $P_x(q)$.

To evaluate the effectiveness of the route-shortening heuristic, a probability upper bound for the emerging of an SR with a length greater or equal to l is given ($l \geq 2$). Let q_{min} be the lowest node transitivity of a node in an SR $[X_1 : X_2 X_3 \cdots X_{l-1} : X_l]$. If and only if every node X_i with $0 < i < l - 1$ does not have connectivity to the nodes $X_{i+1}, X_{i+2} \cdots X_l$, the SR cannot be shortened. Thus, the probability upper bound for the emerging of an SR of a length greater or equal to l is

$$RS_{\geq l}(q_{min}) = (1 - q_{min})^{\sum_{i=1}^{l-2} i}. \quad (1)$$

Consequently an upper bound for the probability of the emerging of an SR of length l is

$$RS_{=l}(q_{min}) = RS_{\geq l}(q_{min}) - RS_{\geq l+1}(q_{min}). \quad (2)$$

Equations 1 and 2 show, that the probability for a long SR to emerge decreases faster than exponential, with $(1 - q_{min})$ as basis. In reality, the probability of the emerging of an SR of length l is much lower, e.g., because not all nodes X_i have node transitivity q_{min}. This is confirmed by measurements in PlanetLab, for instance.

To evaluate the overhead of the route-shortening heuristic, an upper bound for the expected value of the additionally produced messages is given. For an SR of length l the exact number of sent probing messages is $\sum_{j=1}^{l-2} j$. Let R be the total number of SRs in the overlay. R can be calculated by multiplying the L_{total} by the average number of neighbors per node. This, combined with Equation 2 results in an upper bound of

$$R \cdot \sum_{l=2}^{L_{total}} RS_{=l}(q_{min}) \cdot \sum_{j=0}^{l-2} j \quad (3)$$

for the overall route-shortening overhead.

To evaluate the effectiveness of the load-balancing heuristic, the total number of non-balanceable SRs is taken as a measure. For a non-balanceable SR, the load-balancing heuristic fails, due to a lack of alternative nodes. Let node X have n overlay neighbors. The probability of k neighbors ($k \leq n$) having connectivity to X is binomially distributed with a density function $B_q(n, k)$. The probability of the $n - k$ remaining neighbors of having no connectivity to one of the other k neighbors, is $(1 - q)^k$. Thus, the expected number of non-balanced neighbors is:

$$\int_0^1 P_x(q) \cdot \sum_{k=0}^{n} B_q(n, k) \cdot \sum_{j=0}^{n-k} B_{(1-q)^k}(j) \cdot j \, \mathbf{d}q \quad (4)$$

at node X.

To evaluate the overhead of the load-balancing heuristic, upper bounds for the number of sent messages

are given. With increasing node transitivity the number of sent messages also increases. If a node X has $n-k$ indirect SRs, which are in a worst case all non-balanced,

$$O_q(n,k) = (n-k)\cdot[(1-q)^k \cdot k + \sum_{j=0}^{k-1}(1-q)^j \cdot q \cdot j)] \quad (5)$$

is the expected value for the overhead produced at X for a certain q. Consequently, with $P_x(q)$ and Equation 4,

$$\int_0^1 P_x(q) \cdot \sum_{k=0}^{n} B_q(n,k) \cdot O_q(n,k)\, \mathbf{d}q \quad (6)$$

is the overall load-balancing overhead.

To illustrate the effect of Equations 4 and 6 an example calculation of the load-balancing effectiveness and overhead for a CHORD DHT with $L_t = 10^6$ nodes is shown in Figure 5 under various node transitivity conditions. The node transitivity q is set identical for all nodes, because $P_x(q)$ is unknown. Every CHORD node has $n = 20$ fingers (neighbors). The graph shows that the load-balancing algorithm balances all SRs for a q of 0.5, which is far below the minimum node transitivity measured in PlanetLab (see below). Also the load-balancing overhead is relatively small for a q of more than 0.5. The figure shows that for $q = 0.5$ about 10^7 load-balancing overhead messages are sent. $q = 0.5$ means, that half of all $2 \cdot 10^7$ SRs are indirect. Thus, only a single load-balancing overhead message per indirect SR is produced in this case.

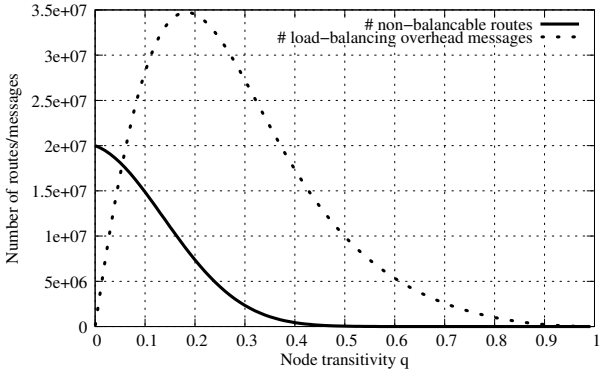

Figure 5: Load-balancing effectiveness and overhead, depending on the node transitivity.

Note, that route-shortening and load-balancing strategies have been analyzed separately. In practice however the two strategies interfere with each other and increase their effects. The load-balancing procedure applied after the route shortening, for instance, might further shorten an SR. It is not trivial to estimate these interference effects, thus the real behavior of the system has been measured in PlanetLab.

There are several measurement studies proving the existence of NTC in PlanetLab [3, 5]. The experienced degree of NTC in these measurements has been about 9.9% (triple based measure) in average. The studies are based on the data provided by [14] where the connectivity (node pairs which are able to ping each other) of currently about 300 nodes is collected daily. In this contribution, further node connectivity experiments have been done. On one hand, a more current view on the PlanetLab's situation has been gathered. This is important, because the PlanetLab network grows constantly. In our experiments a total number of 390 usable nodes were identified. On the other hand, the node transitivity q has been calculated from the gathered data, because q and especially the distribution $P_x(q)$ can not directly be derived from the NTC percentage. The set of 390 PlanetLab nodes used in the experiment had connectivity to one node at the University of Passau (UP). This node was also used as a bootstrap node in the later CHORD experiments.

Every PlanetLab node attempted to establish a connection to all other nodes via the *secure shell* (ssh) application. According to PlanetLab conventions, ssh ports are always open at every node. Thus, if a connection to the ssh port times out, it is either due to NTC or to congestion. To minimize the probability of a failure due to congestion, a high timeout value of about 60 seconds has been chosen.

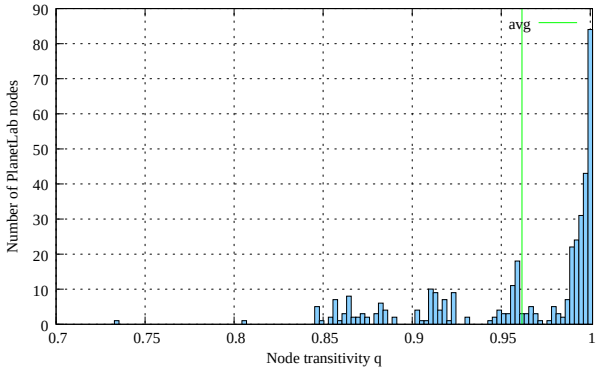

Figure 6: Node transitivity in PlanetLab.

The measured node transitivity q for every PlanetLab node is shown in Figure 6. No PlanetLab node had a node transitivity lower than 0.7, whereas the average node transitivity was about 0.96. According to this, a 4% degree of NTC (node pair based measure) has been calculated from the collected data. Applying the 0.7 lowest node transitivity to the graph from Figure 5 yields an upper bound of 0 non-balanceable SRs and about 0.41 load-balancing overhead messages per indirect SR.

In Figure 6 four different clusters can be seen, containing nodes with nearly similar node transitivity. To-

gether with results of previous measurements this indicates that there is no trivial (closed formula) probability distribution approximating $P_x(q)$ in general. As a consequence NTC modeling of a particular network has to be supported by measurements.

In a second experiment the CHORD implementation described in Section 3 was tested in PlanetLab. After the bootstrapping and stabilizing of the DHT, a snapshot of all SRs in the system has been made. First, the effectiveness of the proposed solution has been confirmed. All nodes established successfully connections to their fingers. Approximately 3% of the direct logical neighbors (successors and predecessors in CHORD) had indirect routes. These connections would not have been operable without the suggested modifications and have caused severe problems. Generally, about 97% of all routes in the system were direct, complying with 96% measured average node transitivity. About 98% of all indirect SRs had only one IN. Only two indirect SRs had two INs and no indirect SR had more than two INs, proving the effectiveness of the suggested route-shortening and load-balancing mechanisms. The effectiveness of the suggested load-balancing mechanism in PlanetLab is shown in Figure 7.

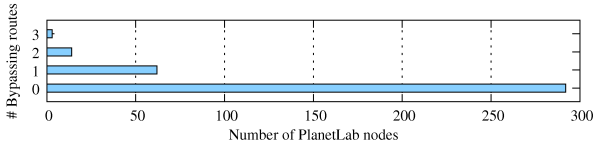

Figure 7: Load distribution in PlanetLab.

It can be observed, that only 3 nodes carry the load of three bypassing SRs. 14 nodes carry two SRs, 62 nodes are involved in a single SR and the rest is never used as IN. Only about 4.3% of the nodes carry load that could be further balanced, but there is no node involved in more than three SRs.

5 Related Work

Structured P2P overlays such as DHTs [7, 10, 13] implicitly rely on the assumption of connectivity between all pairs of overlay nodes. However, evaluations among the PlanetLab hosts, for instance, revealed about 9,9% of node triples with NTC [5]. As a reason for that level of NTC in PlanetLab, a division into three classes of nodes was identified: Nodes of the first two classes have only connectivity to nodes within their class and to nodes in the third class, whereas the nodes of the third class have connectivity to nodes of all classes [5]. Based on the data provided in [14], transient routing problems where also found to be a main source for NTC in PlanetLab [3].

Problems arising from NTC for DHTs are extensively discussed in [3]. DHTs suffer from invisible nodes, routing loops, broken return paths, inconsistent roots, etc. In [3], the problems for the DHT systems OpenDHT [10] (based on Bamboo [9]), i3 [12] (based on CHORD [13]), and Coral [2] (based on Kademlia [7]) were analyzed. Isolated solutions for each of the mentioned problems were discussed. The effects of invisible nodes, for instance, can be decreased by the use of virtual coordinate systems, sending several lookup threads in parallel, or caching unreachable nodes. The inconsistent root problem can be addressed by consensus algorithms, which however are not explored well in the DHT context, and so on. The disadvantage of such separate solutions is that they do not eliminate the source of these problems, the NTC. No comprehensive solution was given to generally solve the impact of NTC to DHTs.

A technique related to source routing is applied in Gnutella [11]. Reachability of nodes is ensured there by flooding the overlay network, thus finding a indirect route to a target. However, flooding does not scale well,so this approach can not be utilized for DHTs.

The degree of NTC in mobile environments is usually much larger than in fixed networks. The feasibility of source routing for DHTs in MANETs is demonstrated in [4]. However, the presented approach is not applicable to fixed networks as, in order to find SRs, all nodes within transmission range of a node are probed. Only in MANETs two nodes having connectivity are also physical neighbors. The number of neighbors of a node is typically quite small. In a fixed network the number of nodes a node has connectivity to is usually larger in orders of magnitude.

Another DHT implementation for fixed networks that addresses NTC issues is FreePastry [1]. It performs NTC detection by exchanging link state information among leaf set nodes. In case of detected NTC, a node randomly picks an IN from its own leaf set. Besides the disadvantage of being probabilistic, this approach is restricted to the local leaf set, all other nodes in the fingertable are not considered. The mechanism is implemented in such a way, that indirect routes with more than one IN cannot be constructed.

6 Conclusion

Running DHTs in fixed networks with NTC combined with dynamic properties like churn is a difficult task, several problems have to be solved. In current DHT implementations including OpenDHT, i3, or Coral, each of the evolving problems is fixed with particular isolated solutions. In contrast to these separated algorithms, this paper has presented a general comprehensive solution to run DHTs in networks with NTC. To achieve this, an integrated set of algorithms and heuristics have been suggested to combine the well known source routing mech-

anism with DHTs in fixed networks. The proposed solution can easily be applied to existing and future DHT protocols without loosing or changing their advanced properties. The suggested modifications of the DHTs impose no additional overhead in the absence of NTC, and operate effectively and efficiently in networks with NTC.

As a proof of concept, the standard CHORD DHT protocol has been modified to support source routing and has been evaluated in PlanetLab. Additionally, a new NTC measure and a statistical NTC model have been developed to analyze the effectiveness and overhead of the suggested solution. Both, statistical analysis and measurement have proved a very good performance and limited overhead. The statistical analysis has indicated that much higher degrees of NTC than existing in PlanetLab are manageable with the suggested mechanisms.

Further heuristics will be evaluated in future work to improve the operation of DHTs in networks under harder NTC conditions. Apart of that, the integration of unidirectional connectivity into the suggested solution will be researched.

References

[1] FreePastry release notes. http://freepastry.org/FreePastry/README-2.0b.html. (last visited: 11/23/2006).

[2] M. J. Freedman, E. Freudenthal, and D. Mazières. Democratizing content publication with Coral. In *Proceedings of the 1st USENIX Symposium on Networked Systems Design and Implementation (NSDI '04)*, pages 239–252, 2004.

[3] M. J. Freedman, K. Lakshminarayanan, S. Rhea, and I. Stoica. Non-transitive connectivity and dhts. In *Proceedings of USENIX WORLDS 2005*, pages 55–60, 2005.

[4] T. Fuhrmann. Combining virtual and physical structures for self-organized routing. In *International Workshop on Self-Organizing Systems*, pages 49–61, 2006.

[5] S. Gerding and J. Stribling. Examining the tradeoffs of structured overlays in a dynamic non-transitive network. 6.829 fall 2003 class project. http://pdos.csail.mit.edu/~strib/docs/projects/networking_fall2003.pdf, MIT, 2003.

[6] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[7] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. In *Peer-to-Peer Systems: First InternationalWorkshop, IPTPS 2002. Revised Papers*, volume 2429/2002, pages 53–65, 2002.

[8] L. Peterson, T. Anderson, D. Culler, and T. Roscoe. A blueprint for introducing disruptive technology into the Internet. *SIGCOMM Comput. Commun. Rev.*, 33(1):59–64, 2003.

[9] S. Rhea and D. Geels. Handling churn in a DHT. In *USENIX 2004 Annual Technical Conference*, pages 127–140, 2004.

[10] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu. OpenDHT: a public DHT service and its uses. In *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 73–84. ACM Press, 2005.

[11] M. Ripeanu. Peer-to-peer architecture case study: Gnutella network. In *Proceedings of the First International Conference on Peer-to-Peer Computing*, pages 99–100, 2001.

[12] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet indirection infrastructure. In *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 73–86. ACM Press, 2002.

[13] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149–160. ACM Press, 2001.

[14] C. Yoshikawa. PlanetLab All-Pairs Pings. http://ping.ececs.uc.edu/ping. (last visited: 11/29/2006).