dCBR: A Global View on Network Coordinates for More Efficient Peer-to-Peer Systems

Bernhard Heep
Institute of Telematics
Karlsruhe Institute of Technology (KIT)
76128 Karlsruhe, Germany
Email: heep@kit.edu

***Abstract*—Minimizing routing latencies in structured peer-to-peer systems that provide the key-based routing service (KBR) is an important field of research as many novel applications are based on these systems. These applications and their users would benefit from faster overlay routing procedures. In this paper we present *decentralized Coordinate-based Routing* (dCBR), a method to achieve low routing latencies in structured peer-to-peer systems by exploiting decentralized network coordinates like Vivaldi and KBR monitoring systems. A global view on network coordinates is distributed among all participating overlay nodes and utilized to generate either topology-based and uniform distributed node identifiers for overlay nodes. This way, a better routing performance is achieved and load-balancing is maintained. dCBR is the successor of the previously proposed *Coordinate-based Routing* (CBR) system, which—combined with Proximity Neighbor Selection (PNS)—significantly decreases routing latencies, due to the fact that it allows for target-oriented routing. dCBR enables the usage of decentralized network coordinate systems in a CBR-based network, making the overlay independent of a landmark node infrastructure.**

I. Introduction

Structured peer-to-peer systems offer a base for a broad range of applications like distributed storage, mobility support and application layer multicast (ALM). Current research efforts concentrate on multi-media P2P applications like e.g. decentralized Voice-over-IP (VoIP)[1] and video streaming applications. Structured P2P systems usually form an overlay network on top of a given infrastructure like the Internet and provide a simple service for upper network layers: *Key-based Routing* (KBR). Here, identifiers (*nodeIds*) are assigned to all participating nodes and messages are sent to keys instead of network addresses. The message is always delivered to the node that is currently responsible for the destination key. In today's KBR protocols, routing usually needs $\mathcal{O}(\log N)$ hops, where N is the total number of overlay nodes.

For minimizing KBR latencies, different topology adaptation mechanisms have been proposed and deployed with several of today's state-of-the-art protocols. The utilized mechanisms can be categorized as *Proximity Routing* (PR), where the routing decisions are affected, *Proximity Neighbor Selection* (PNS), where a node's choice of peers (i.e. the entries in the local overlay routing tables) is modified, and *Topology-based NodeId Assignment* (TbNA), where the overlay identifiers are determined depending on the underlaying network topology. Due to the fact that TbNA manipulates the overlay nodes' identifiers—usually according to network coordinates assigned by a network coordinate system (NCS)—it leads to a non-uniform distribution of nodeIds, which has negative impacts on load-balancing and routing performance. For this reason, no further research efforts have been made in this area so far.

nodeId: 10233102	0	1	2	3
-	**0**2212102	(local node)	**2**2301203	**3**1203203
1	(local node)	1**1**301233	1**2**230203	1**3**021022
10	10**0**31203	10**1**32102	(local node)	10**3**23302
102	102**0**0230	102**1**1302	102**2**2302	(local node)
1023	1023**0**322	1023**1**000	1023**2**121	(local node)
10233	10233**0**01	(local node)	10233**2**32	
102331	(local node)		102331**2**0	
1023310			(local node)	

Fig. 1. Pastry's Routing Table, $b = 2$

The contribution of this paper is as follows: We introduce the TbNA-system *decentralized Coordinate-based Routing* (dCBR), which assigns uniform distributed but topology-based nodeIds to nodes in structured peer-to-peer systems to decrease KBR latencies. This is achieved by target-oriented routing and the combined usage with PNS. The assigned nodeIds are based on decentralized network coordinates where it is assumed, that physically close nodes also have close coordinates in the coordinate space.

The rest of this paper is organized as follows: In Section II the combination of the topology adaptation mechanisms PNS and TbNA in prefix-based KBR-protocols is discussed. In Section III the original *Coordinate-based Routing* (CBR) system is described. Section IV illustrates the dCBR system and its components. After an overview of current and an outlook on future evaluation in Section V, the related work is discussed in Section VI. The paper ends with the conclusion in Section VII.

II. Combining Topology Adaptation Methods

In this section, a combination of PNS and TbNA is discussed in detail. PNS considers proximity while constructing the routing tables of an overlay node. If a node X is found by A, that fits in the routing table of A, an existing entry Y is displaced if X is physically closer than Y to A. If there is space for k entries for this position in the routing tables, the closest k nodes are kept. When routing a message, only

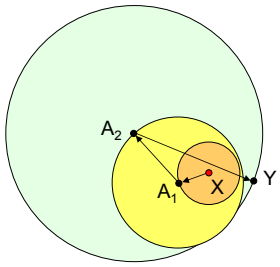
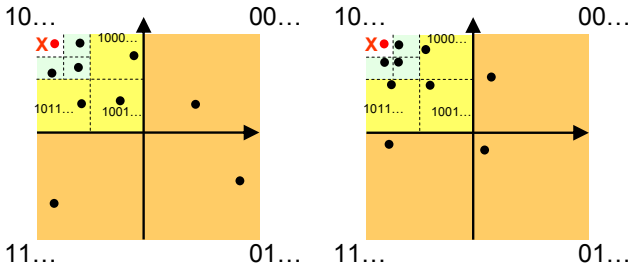
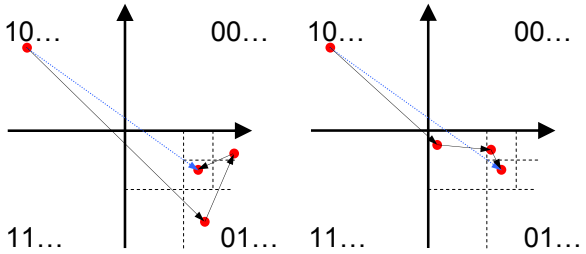


Fig. 2. Effect of PNS on KBR routing hops



(a) TbNA without PNS (b) TbNA with PNS activated

Fig. 3. Routing tables when using TbNA and PNS



(a) TbNA routing without PNS (b) TbNA routing with PNS

Fig. 4. Routing paths when using TbNA and PNS

progress in id-space is considered. The latency of all peer candidates must be determined before they can be put into the routing tables.

Pastry[2] and Bamboo[3] are *prefix-based* KBR-protocols since overlay nodes in a Pastry/Bamboo network maintain routing tables holding other nodes whose nodeIds share a common prefix with the local node's identifier (s. Fig. 1). The nodes' identifiers $X = d_1, d_2, ..., d_n = x_1, x_2, ..., x_m$ are partitioned into digits (d_j) consisting of b bits (where m is the identifier's length—usually 128 or 160 Bits—and $n = m/b$). The routing table holds nodes with a common nodeId prefix. It is organized as a matrix R_{ij} with $2^b - 1$ columns. The row index n indicates the number of common digits of the present and the foreign node's id, starting in the first row with zero common digits. So nodes in row n share exactly $n - 1$ common digits. For routing decisions, i.e. to chose the next hop to forward a message, all KBR protocols need a routing metric $d_{\text{route}}()$. A node X chooses $A_i \in P_X$ (where P_X is the set of peers of X) as next hop to route a message to key z if $d_{\text{route}}(A_i, z) = \min$. Prefix-based KBR protocols use $d_{\text{prefix}}()$ as routing metric:

$$d_{\text{prefix}}(X, Y) = \begin{cases} 0 & d_i = e_i, 0 \leq i < n \\ n - k & d_i = e_i, d_{k+1} \neq e_{k+1} \end{cases}$$

with $X = d_1 d_2 ... d_n$, $Y = e_1 e_2 ... e_n$, $0 \leq i \leq k < n$, and $d_{\text{prefix}}(X, Y) \in [0 : n] \subset \mathbb{N}$.

Due to the fact that there are several possible peers available for one position in the routing table, both protocols are able to perform PNS, i.e. for each position in the routing table, the node is selected that is physically closest[1]. The shorter the common prefix, the more overlay nodes fit into the considered position of the routing table. The size of the set of possible nodes that have a common prefix of i is N/b^{2^i} when nodeIds are uniform distributed. Therefore, when using PNS, the first routing hop—for that usually the first row of the routing table is used—is physically shorter than all the following hops. The last hop has the latency of an average one-way latency in the underlaying network since the node is found in the local *LeafSet* structure, where proximity cannot be considered as it holds the overlay nodes that have the directly preceding and succeeding nodeIds to the local node.

Fig. 2 illustrates the effect of key-based routing with PNS enabled: Node X forwards a message with the destination key x to A_1, which is from the 1st row of X's routing table, thus having the longest common prefix (1 digit) with x. As there is a wide range of node candidates, i.e. $\frac{N}{b^2}$ fitting into positions of the first row, most probably the message is forwarded to a very close node. At X_1, the the routing table's second row is searched though. Since here, the set of candidates is $\frac{N}{b^4}$, a higher latency may be expected for routing the message to A_2. By each routing hop, the radius of expected latencies gets bigger (illustrated by circles) as the number of possible candidates for lower rows decreases.

PNS is a effective way to decrease KBR routing latencies. However, in the worst case, all hops lead to antipodal directions and the next to last node on the routing path is far away from the destination node.

When using TbNA, nodes with close nodeIds according to the routing metric $d_{\text{route}}()$ (e.g. the prefix metric $d_{\text{prefix}}()$) are also physically close. This way, every single hop leads towards the destination node, as every hop gets closer to the destination key according to the routing metric $d_{\text{route}}()$. When applying TbNA to Pastry or Bamboo, each position in the routing table represents an area in physical space as shown in Fig. 3, where the physical space is represented by a 2-dimensional euclidean space. Both figures show the routing table of node X, rows are shown in different colors. Fig. 3(a) illustrates the effect of TbNA: The first row of node X's routing table holds nodes from all $b^2 - 1$ main areas the node does not belong to (here with the prefixes 00, 01, and 11). The 2nd row hold nodes from all $b^2 - 1$ sub-areas from the main area the node is located in (here with the prefixes 1011, 1001, and 1000), and so on. Fig. 3(b) shows the selected nodes for the first three rows of the routing table with TbNA and PNS enabled. Here, each node selects the physically closest nodes out of each area.

[1]Pastry and Bamboo only store a single node at each position in the routing table, i.e. $k = 1$

Fig. 4 shows the resulting routing paths: When PNS is deactivated as illustrated in Fig. 4(a), it can be observed that routing paths do not lead target-oriented to the destination node since single hops lead anywhere into the areas. But when using a combination of PNS and TbNA (s. Fig. 4(b)), this leads to almost direct routing paths. For comparison, the direct connection or ideal routing path to the destination node is shown additionally in both figures. This way—theoretically—the lowest possible KBR latencies can be achieved, assuming the following constraints:

- The underlaying network meets (in most cases) the triangle inequality, i.e. the triangle inequality violation (TIV) occurs rarely.
- The employed network coordinate system which is used as representation model of the physical space provides an accurate picture of the underlaying network, i.e. physically close nodes have close identifiers and nodes that are physically far away from each other are also far away in identifier space according to $d_{\text{route}}()$.
- The employed KBR-protocol needs a minimal number ($\mathcal{O}(\log\ N)$) of routing hops to deliver a message to a destination node (except for CAN[4], all well-established protocols do so).

In this paper we combine PNS and TbNA using a prefix-based KBR-protocol with the goal of minimal KBR latencies for KBR-based applications. Here, the main problem we have to face is the non-uniform distribution of coordinates which leads to areas with different node densities. When assigning nodeId prefixes to these areas this obviously results in non-uniform distributed nodeIds. However, most KBR-protocols rely on uniform distributed nodeIds to provide routing paths of $\mathcal{O}(\log\ N)$ hops. Otherwise, even with smaller delays between nodes on the routing path, a higher number of routing hops has negative impact on the total latency of the routing procedure $l_{\text{KBR}} = k\Delta$, where Δ is the average delay between overlay nodes and k the number of routing hops. Increasing the number of routing hops will compensate the positive effect of lower per-hop latencies.

Another problem that occurs in overlay networks with non-uniform distributed nodeIds is an unfair load-balancing among the participating nodes: When the KBR service is used by an application for e.g. distributed storage, one node has to store more data than others thus needing more network resources, memory, and computational power. The CBR/dCBR approaches described in the next sections tackle this problem.

III. Coordinate-based Routing

dCBR is based on a landmark-based approach called CBR, which has been proposed in [5]. The main idea of CBR is to map the network coordinates $X_A = (x_1, x_2, ..., x_n)$ of a node A to a nodeId prefix $a_1a_2...a_m$ using a global picture of prefix areas:

$$f_{\text{CBR}} : (x_1, x_2, ..., x_n) \longmapsto a_1a_2a_3...a_m$$

Then A's nodeId I_A is calculated as follows, where c_j are random bits for the nodeId's suffix:

$$I_A \longleftarrow a_1a_2...a_mc_1c_2...c_{i-m}$$

In prefix-based KBR protocols the common prefix of the nodeId and the destination key grows with each hop. When combined with CBR, the first hop leads to the whereabouts of the destination node (in the overlay's id-space as well as physically in the underlay), since the longer the common nodeId prefix of two nodes is, the physically closer they are. The following (smaller) hops lead all target-orientedly to the destination as well, resulting in significantly decreased routing latencies even under churn.

For the deployment of CBR, two requirements have to be met: The usage of a recursive, prefix-based KBR protocol like Pastry and Bamboo and the availability of a landmark-based network coordinate system like GNP[6] and NPS[7]. These systems offer an estimation of latencies between nodes A and B by calculating the euclidean distance of their coordinates $||X_A - X_B||$. The landmark nodes form the base of the coordinate system by providing the reference coordinates to ordinary nodes. These nodes probe several landmarks, receive their coordinates—the base coordinates thus spanning the coordinate space—and calculate their own coordinates using the measured RTTs.

CBR partitions the underlay—represented by the network coordinates—into areas, where a specific nodeId prefix is assigned to each of them. They all have a lower and an upper bound in each dimension. Because of the non-uniform distribution of nodes in the coordinates' topology, the area sizes depend on the density of nodes: The sparser a topology region is populated, the larger the prefix area is (s. Fig. 5). This leads to a uniform distribution of nodeId prefixes in a non-uniformly populated space. In order to determine the correct positions of the borders, a global picture of coordinate spreading—here called the *Global Knowledge* (GK)—is needed. The network coordinates of a representative number of participating overlay nodes are calculated and collected. This is accomplished by latency monitoring nodes, preferably the landmark nodes. As soon as sufficient coordinates are collected the GK can be created by drawing the borders accordingly and attaching prefixes to the resulting regions. Then, the GK must be distributed among all nodes joining the overlay. As long as the GK's coordinate basis or the landmark nodes do not change, new nodes can look up their appropriate nodeId prefix in the distributed GK using their network coordinates.

As overlay nodes are aware of their positions and the positions of other nodes and keys in a CBR-based network, CBR allows for a location-based replication strategy in DHTs: A key-value pair is stored on n nodes instead of only one. These n nodes are responsible for one of the keys $K_n \in C$, with the given set of hashes $C = \{K|K = H^i(V), 0 < i \leq m\}$ using a given hash function H and a value V. Here, m is called the number of *spreaded replicas*. The destination key for *get()*-requests in the DHT is chosen out of all these n candidates. Only a single message is sent to the closest key K_n with respect to the proximity information taken from the global knowledge, i.e. the closest node is chosen that holds the value, leading to a low delay for delivering the value to the initiator.

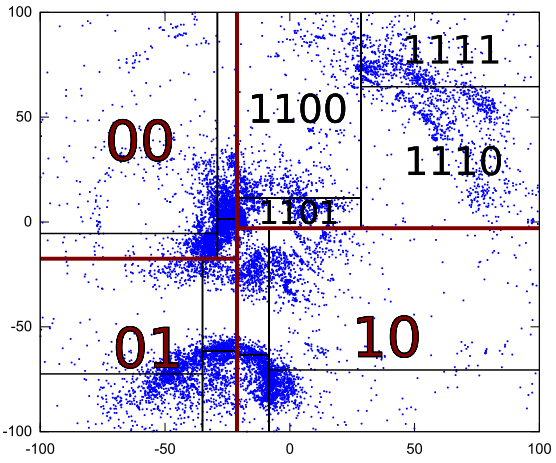


Fig. 5. Coordinate spreading and corresponding CBR prefix areas (here coordinates calculated from the Skitter data set)

IV. Decentralized CBR

CBR achieves low routing latencies in KBR overlays but it has several drawbacks, that make it a solution only for a few network scenarios:

- A centralized landmark infrastructure must be provided.
- Fixed coordinates must be assigned to the overlay nodes
- No adaptation of coordinates is possible, which may lead to inaccuracy and hence poor routing performance
- The GK has to be gained and distributed a priori

The dCBR approach relies on decentralized network coordinate systems, thus landmark nodes are unneeded. The GK is replaced by the *Current Coordinate Distribution* (CCD), which represents the current coordinate spreading in the overlay network and the *Current Area Partitioning* (CAP), which is the current partitioning scheme, i.e. information about the borders of the currently used prefix areas. Both the CCD is gained and the CAP is distributed while the overlay protocol is in service, i.e. not a priori.

A. Overview

dCBR makes use of two well-known techniques to avoid the drawbacks of CBR: *Decentralized NCS* and *Monitoring in Structured P2P Systems* with SOMO[8] or similar systems[9].

In dCBR no fixed coordinate base—and hence no landmark nodes—are needed since the coordinate spreading is not gained a priori but generated from the coordinates of the currently participating overlay nodes. As long as the CCD is updated in reasonable intervals, it is improbable that the coordinate system has rotated or shifted and nodes are placed into the wrong area and so assigned the wrong nodeId prefix.

The main idea of dCBR is therefore to collect the CCD periodically, calculate the CAP (using the partitioning scheme from Sect. III) at a single node, and distribute the CAP to all participating nodes, which forwards the CAP to all new overlay nodes that try to join the overlay. This way, joining nodes can assign their nodeIds immediately after calculating initial network coordinates.

B. Decentralized Network Coordinate Systems

In contrast to GNP and NPS decentralized NCS like Vivaldi[10] or one of its successors [11][12][13][14][15] do not rely on dedicated landmark nodes. Coordinates are calculated from latency measurements and coordinates of other nodes. Additionally, most decentralized NCS complete the coordinates with an error value that expresses their accuracy. Using this data, coordinates are continuously adapted and converge against a position in the coordinate space with a minimal error of latency estimation. For modeling the access network, most decentralized NCS provide a *height vector*. Its value is added to the coordinate distance for estimating latencies. For dCBR the height vector can be used but is ignored when assigning a node's identifier prefix as it cannot be mapped to the CAP.

For some NCS using spherical coordinates (e.g. Htrae), geographic coordinates, i.e. the latitude and longitude are needed to calculate accurate initial coordinates. For this, the *GeoIP*[16] system is used to map the node's IP address to geographic coordinates using the local GeoIP data-set.

As CBR requires stable coordinates—to avoid a permanent change of area belongings and therefor nodeId alteration—initial coordinates as accurate as possible are needed for dCBR. This can be achieved by either using geolocation, and/or an exhaustive discovery mode, where many other overlay nodes are probed and those coordinates are used for calculating the own coordinates. Some NCS like SVivaldi offer a mechanism, where the adaption of coordinates is limited by a decreasing factor, hence all overlay nodes finally take a stable position. This factor can be adjusted to the number of nodes that are probed during the discovery mode to achieve stable coordinates before joining the overlay network.

C. Monitoring of Network Coordinates

The collection of coordinate data and the distribution of the CAP is done by the monitoring unit which builds up a second overlay—usually a tree structure—on top of the established KBR overlay. All nodes find their positions in the tree with respect to their current nodeId: One node becomes the root node as it is responsible for the middle of the id-space. The id-space is partitioned into 2^d domains, where other nodes are responsible for, as the keys in the middle of these domains fall into their range of responsibility. The domains are partitioned the same way recursively until all nodes have found their domain. This way, each overlay node has a parent node one layer above (expect for the root node) and 2^d children (except for the leaf nodes). This structure is stable under churn if the utilized KBR protocol is, as all communication in the tree is based on KBR.

The leaf nodes propagate their current coordinates to the node one layer above. These nodes collects the coordinates from all its children, add their own coordinates and forwards them to the layer above towards the root node of the tree. This way, the root node gets a global picture of all currently assigned coordinates sent from its children. Then, the root node calculates the CAP the same way like it is done in CBR a priori, and sends it back to its children, which also forward

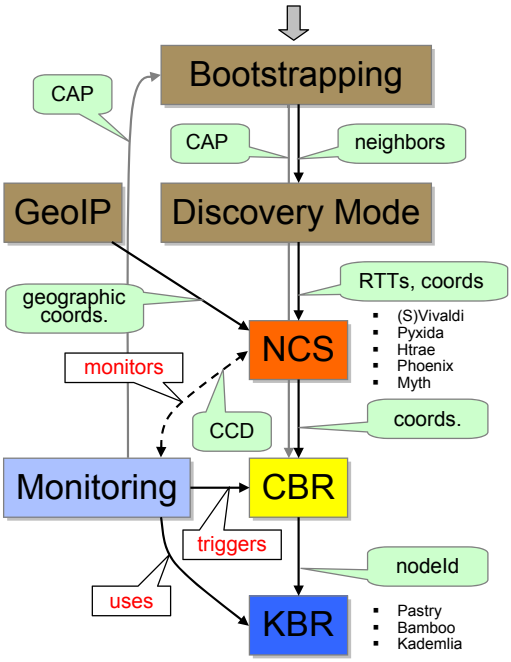


Fig. 6. Overview of the operations in dCBR

the CAP to their children and so to all overlay nodes in the tree. Joining overlay nodes receive the CAP from its bootstrap node to determine their nodeId.

D. Aggregation & Compression of Coordinate Data

When collecting and forwarding the coordinates, a certain problem has to be considered: The bandwidth consumption needed for the transmission of coordinate information must not exceed a reasonable limit in order to keep the system scalable. Several solutions facing this problem are feasible:

- The pool of coordinates can be thinned out at each node towards the root node by removing random coordinates if the total number of coordinates received by this node exceeds a predefined number. This way, the size of coordinate data sent up in the tree is limited while retaining a representative picture of global coordinate spreading.
- The coordinate values can be rounded or quantized to avoid sending too accurate coordinate data, needlessly to generate the CAP.
- The coordinate space can be partitioned in small, a priori determined and numbered areas of equal size, where all nodes are assigned to. Due to the numeration, the information about a pool of coordinates can forwarded without explicitly sending coordinate values but only the number of nodes belonging to these areas.

All these approaches can be combined to keep the monitoring traffic low, a trade-off between bandwidth consumption and accuracy of the CAP must always be considered, though.

E. Example & Summary

To summarize, Fig. 6 illustrates all operations of dCBR. The following steps a node has to take to join a CBR-based overlay network:

1) **Bootstrapping**: A node that wants to join the dCBR-based overlay network initially contacts its *Bootstrap Node*[2] and receives the current CAP, a list of other overlay nodes, and the bootstrap node's coordinates.
2) **Discovery Mode**: All the nodes from the list are probed to get their latencies and asked for their current coordinates. The more nodes are probed, the more accurate the node's initial coordinates get.
3) **GeoIP**: Optionally, geographic coordinates are calculated from the node's IP address using the local GeoIP database.
4) **NCS**: The gained coordinate information is combined with the optional geolocation information, if this is supported by the NCS (e.g. Htrae). Then the NCS calculates the initial coordinates using all information gained from the Discovery Mode and GeoIP and passes them to the CBR module.
5) **CBR**: The CBR module calculates the node's nodeId using the CAP and the initial coordinates. The appropriate area from the CAP is selected and a new nodeId is generated using the associated prefix and a random suffix.
6) **KBR**: With the calculated nodeId, finally the node joins the overlay network by sending a join-request to its own nodeId.
7) **Monitoring**: The *Monitoring* unit
   - continuously *monitors* the coordinates calculated by the local NCS,
   - uses the *KBR* overlay to periodically gain the CCD and distribute the CAP to all overlay nodes,
   - and—if it is unavoidable—triggers the *CBR* module to recalculate the node's identifier, which leads to a re-join of the overlay node using its new nodeId.

V. Future Work: Evaluation

In [5] CBR was evaluated in simulated network scenarios with 4,500 overlay nodes using Pastry and Bamboo. The results show that CBR decreases KBR latencies in scenarios without churn up to 13% (Pastry) or 20% (Bamboo), respectively. In scenarios with moderate churn a decrease of latencies by 37% and 15% can be observed. Using the CBR-based replication strategy, get()-delays in a DHT can be decreased up to 60%, depending on the number of spreaded replicas. Since dCBR is based on the same mechanism as CBR, routing latencies in dCBR are expected to be similar to those in CBR, as long as the CCD is accurate enough compared to the GK gained in CBR. The GK is based on a pool of coordinates, where only a subset occur in the network. In contrast, the CCD in dCBR represents all or a subset of the current overlay nodes and their network coordinates.

Nonetheless, a comprehensive evaluation of dCBR is essential. At the moment we are working on a implementation of dCBR using the overlay framework OverSim[17][18] to simulate dCBR in overlay networks with thousands of nodes

[2]Each node that wants to join the overlay network must know the IP address and the UDP port of at least one node that is already part of the overlay network.

and to validate dCBR in testbeds like PlanetLab[19] and G-Lab[20]. Additionally, it is planned to evaluate CBR/dCBR in combination with the widely-used Kademlia protocol[21].

VI. Related Work

In the last years some attempts have been made to realize TbNA in structured peer-to-peer systems: In [22] the authors present a system to assign location-based nodeId prefixes to overlay nodes in Chord and prefix-based overlay networks like Pastry and Bamboo. Here, geographic regions (e.g the north-west of the USA) are mapped to different prefixes. To achieve a uniform distribution of nodeIds the authors only rely on assumptions about the node density of these regions. In [23] the authors propose a system where the IPv6 address prefixes are used for the generation of corresponding nodeId prefixes. In [24] prefixes are assigned according to the autonomous system (AS) the node belongs to. Both approaches do not consider the resulting non-uniform distribution of nodeIds. The developers of Canary[25] utilize the Vivaldi coordinate system for directly assigning nodeIds in a CAN overlay network. Again, the problem of non-uniform distributed nodeIds is not considered.

VII. Conclusion

In this paper we presented dCBR, a system to exploit a global view on network coordinates to assign uniform distributed but topology-based nodeIds in structured P2P systems. This way, better routing performance i.e. lower routing latencies in KBR overlay networks can be achieved while maintaining load-balancing in the overlay network, which was a major drawback of earlier proposed approaches that manipulate the overlay nodes' identifiers. dCBR does not change the main behavior of the employed protocols. It is based on the recently proposed CBR system but relies on decentralized NCS and KBR monitoring systems, thus no centralized and expensive infrastructure is needed. In the future, dCBR can be the base of large-scale applications which make high demands on routing latencies, like a decentralized *Domain Name System* (DNS) or gaming overlays.

References

[1] I. Baumgart, B. Heep, and S. Krause, "A P2PSIP Demonstrator Powered by OverSim," in *Proceedings of 7th IEEE International Conference on Peer-to-Peer Computing (P2P2007), Galway, Ireland*, Sep. 2007, pp. 243–244.

[2] A. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems," in *Middleware 2001 : Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg, Germany*, vol. 2218/2001, Nov. 2001, pp. 329+.

[3] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz, "Handling Churn in a DHT," in *ATEC '04: Proceedings of the annual conference on USENIX Annual Technical Conference*, Boston, MA, USA, Jun. 2004, pp. 127–140.

[4] S. Ratnasamy, P. Francis, S. Shenker, and M. Handley, "A Scalable Content-Addressable Network," in *In Proceedings of ACM SIGCOMM*, 2001, pp. 161–172.

[5] F. Hartmann and B. Heep, "Coordinate-based Routing: Refining NodeIds in Structured Peer-to-Peer Systems," in *Proceedings of the International Conference on Ultra Modern Telecommunications & Workshops (ICUMT'09)*, St. Petersburg, Russia, Oct. 2009.

[6] T. S. E. Ng and H. Zhang, "Predicting Internet Network Distance with Coordinates-Based Approaches," in *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, 2001, pp. 170–179.

[7] T. S. E. Ng and H. Zhang, "A Network Positioning System for the Internet," in *ATEC '04: Proceedings of the annual conference on USENIX Annual Technical Conference*. Berkeley, CA, USA: USENIX Association, 2004.

[8] Z. Zhang, S. Shi, and J. Zhu, "SOMO: Self-Organized Metadata Overlay for Resource Management in P2P DHT," in *Peer-to-Peer Systems II, Second International Workshop, IPTPS 2003, Revised Papers*, Feb. 2003, pp. 170–182.

[9] K. Graffi, D. Stingl, J. Rueckert, A. Kovacevic, and R. Steinmetz, "Monitoring and Management of Structured Peer-to-Peer Systems," in *Proceedings of the IEEE Ninth International Conference on Peer-to-Peer Computing (IEEE P2P '09)*, Sep. 9–11, 2009, pp. 311–320.

[10] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: a decentralized network coordinate system," in *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, Portland, OR, USA, Aug. 2004, pp. 15–26.

[11] C. de Launois, S. Uhlig, and O. Bonaventure, "Scalable Route Selection for IPv6 Multihomed Sites," in *Proceedings of Networking 2005*, May 2005, pp. 1357–1361.

[12] J. Ledlie, P. Gardner, and M. I. Seltzer, "Network coordinates in the wild," in *Proceedings of the 4th Symposium on Networked Systems Design and Implementation (NSDI'07)*, Apr. 2007.

[13] Y. Chen, X. Wang, X. Song, E. K. Lua, C. Shi, X. Zhao, B. Deng, and X. Li, "Phoenix: Towards an Accurate, Practical and Decentralized Network Coordinate System," in *Proceedings of Networking 2009*, ser. Lecture Notes in Computer Science, vol. 5550. Springer, 2009, pp. 313–325.

[14] Y. Chen, G. Zhao, A. Li, B. Deng, and X. Li, "Handling node churn in decentralised network coordinate system," *IET Communications*, vol. 3, no. 10, pp. 1578–1586, Oct. 2009.

[15] S. Agarwal and J. R. Lorch, "Matchmaking for online games and other latency-sensitive P2P systems," in *SIGCOMM '09: Proceedings of the ACM SIGCOMM 2009 conference on Data communication*. New York, NY, USA: ACM, 2009, pp. 315–326.

[16] "GeoIP (MaxMind LLC) – Website," http://www.maxmind.com/geoip/, [online] accessed on 05/07/2010.

[17] I. Baumgart, B. Heep, and S. Krause, "OverSim: A flexible overlay network simulation framework," in *Proceedings of 10th IEEE Global Internet Symposium (GI '07) in conjunction with IEEE INFOCOM 2007, Anchorage, AK, USA*, May 2007, pp. 79–84.

[18] I. Baumgart, B. Heep, and S. Krause, "OverSim: A scalable and flexible overlay framework for simulation and real network applications," in *9th International Conference on Peer-to-Peer Computing (IEEE P2P'09)*, Sep. 2009, pp. 87–88, Invited Talk.

[19] L. Peterson, T. Anderson, D. Culler, and T. Roscoe, "A blueprint for introducing disruptive technology into the internet," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 59–64, 2003.

[20] P. Tran-Gia, A. Feldmann, R. Steinmetz, J. Eberspächer, M. Zitterbart, P. Müller, and H. Schotten, "G-Lab White Paper Phase 1 - Studien und Experimentalplattform für das Internet der Zukunft," https://www.german-lab.de/fileadmin/Press/G-Lab_White_Paper_Phase1.pdf, Jan. 2009.

[21] P. Maymounkov and D. Mazières, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric," in *Peer-to-Peer Systems: First International Workshop, IPTPS 2002 Cambridge, MA, USA, March 7-8, 2002. Revised Papers*, vol. 2429/2002, 2002, pp. 53–65.

[22] S. Zhou, G. R. Ganger, and P. Steenkiste, "Location-based Node IDs: Enabling Explicit Locality in DHTs," School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, Tech. Rep. CMU-CS-03-171, Sep. 2003.

[23] J. Xiong, Y. Zhang, P. Hong, and J. Li, "Chord6: IPv6 Based Topology-Aware Chord," in *Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services (ICAS-ICNS 2005)*, Oct. 2005.

[24] W. Wu, Y. Chen, X. Zhang, X. Shi, LinCong, B. Deng, and X. Li, "LDHT: Locality-aware Distributed Hash Tables," in *International Conference on Information Networking (ICOIN 2008)*, Jan. 2008.

[25] T. Kojima, M. Asahara, K. Kono, and A. Hayakawa, "Embedding network coordinates into the heart of distributed hash tables," in *Peer-to-Peer Computing, 2009. P2P '09. IEEE Ninth International Conference on*, Sep. 2009, pp. 155–158.