ABSTRACT
Lookup operations in CAN might be long, as requests are passed only to direct neighbors. Realities ease this problem, but their random nature does not ensure fairness between participating nodes. In this paper we propose multiple levels of long range neighbors, called Reference Points (RPs), placed in a deterministic manner. Their location is calculated for each node so as to ensure reaching any point in the coordinate space in a comparably small number of steps. Multi-level RPs also bring fairness to CAN.

1. INTRODUCTION
Peer-to-peer overlay networks are very much in focus of researchers nowadays. In such networks, scalability and robustness can be achieved through the use of Distributed Hash Tables (DHTs). The hypercube-based CAN (Content-Addressable Network) [1] and the ring-based Chord [2] are well-known DHT solutions, they offer good average path length for lookup operations. The notion of realities in CAN and the finger tables in Chord further improve the performance of these systems compared to the base algorithms. Nevertheless, since zones and neighbors are chosen randomly, CAN realities do not equilibrate paths.

A further enhancement for CAN is proposed in [3]. This paper considers routing tradeoffs between stored states and average path length. It introduces Long Range Nodes (LRNs), i.e., non-direct neighbors of a certain node, allowing for longer jumps in the coordinate space, hence reducing the path length. Neither this solution deals with fairness nor considers routing tradeoffs between stored states and average path length. The notion of realities in CAN realities do not equilibrate paths.

We choose the Reference Points of node $N$ “around” $P_N$. $RP_1(P_N), RP_2(P_N), \ldots, RP_i(P_N)$ are chosen so as to be at equal distance from $P_N$ and to divide the coordinate space in $i+1$ equal chunks, one for $P_N$ and one for each RP. Each point $X$ in the coordinate space belongs to the zone whose responsible vertex (i.e., the RP) is closest to it in Cartesian distance. If node $N$ searches a file that hashes in $X$, it is worth sending the query to the vertex responsible for the chunk that includes $X$. Reference Points are themselves included in zones of CAN nodes and queries, in fact, are sent to the CAN node responsible for the given RP. By using Reference Points, one can jump rapidly “close” to any point of the coordinate space.

2. GEOMETRICAL CONSIDERATIONS
When a node $N$ joins the CAN, it receives an $n$-dimensional rectangular zone to be responsible for. Let the leftmost vertex $P_N$ on the lowest dimension of the zone be the “anchor point” of the node. According to the zone splitting rules, $P_N$ will always belong to node $N$.

The idea for calculating $i$ RPs for node $N$ in a 2-dimensional space is the following: $P_N$ is responsible for an $i$-regular polygon $(POLY_i)$, such as the covered chunk for each RP equals to the area of $POLY_i$. By applying $A_{POLY_i} = \frac{1}{2} \tan \left( \frac{i \pi}{i+1} \right)$, where $a$ is the side of $POLY_i$, and the area of the whole CAN equals to 1 (CAN is represented by a square). From this, a and the circumradius of a big regular i-polygon $(POLY_i, P_N)$ is the center, RPs are located on the vertices) can be derived and so the polygon and the coordinates of $RP_1(P_N), RP_2(P_N), \ldots, RP_i(P_N)$ themselves. In this construction, $P_N$ is equidistant from all the RPs, and the covered chunks are equal in size. By convention, $POLY_i$, is constructed such a way that at least one of its sides is orthogonal to the symmetry diagonal of the CAN square.

Naturally, this method can be extended to higher dimensions. Note, that if $i$ is an odd number the areas covered by certain RPs will not be continuous in hyper-cubic representation because of the cyclic nature of the CAN torus. If $i$ is an even number, this phenomenon does not exist. In Figure 1, an example of a 2-dimensional coordinate space is shown with 3 and 4 Reference Points respectively.

Since RPs are calculated separately by each node, RPs of different nodes are distributed uniformly in the coordinate space. This means that—if there are a certain number of nodes in the CAN—no nodes will be overloaded because
of being responsible for RPs of several nodes. Henceforth, favorable robustness properties of CAN are preserved.

3. MULTI-LEVEL REFERENCE POINTS
Having RPs uniformly distributed in the entire coordinate space is useful: one can move close to any point in space in a single, long step. However, if we would like to use the favorable effect of RPs in further routing steps as well, multi-level Reference Points should be introduced. By having only a single-level of RPs for every given node, after the first, large routing step the basic CAN routing mechanism has to be resumed, since the request may have come “too close” to the destination to use the benefits of the next node’s RPs. To avoid that, we propose to use multi-level RPs: a given node has \( j \) set of RPs, each set consists of \( i \) RPs. The set of RPs are at exponentially decreasing distances from \( P_N \), the anchor point of the given node \( N \) (see Figure 2). The first level of RPs are the original Reference Points mentioned in Section 2.

By using multi-level RPs, we can achieve similar gains in terms of average path length as through the use of finger tables in Chord. An other favorable effect is that multi-level RPs bring fairness to CAN routing, as the variance of the path length as a variate will decrease compared to the original CAN algorithm. Hence, path lengths will be roughly equal regardless of the source and the destination of the lookup operation.

4. MODIFIED OPERATION
Usual CAN operations need to be modified in order to apply the optimized algorithm. When a new node \( N \) joins the CAN, it starts with the usual CAN \textit{Join} mechanism. Once the new node \( N \) received its zone, it determines its “anchor point” \( P_N \) and calculates its RPs, depending on the required number and levels of long range neighbors. If \( N \)'s zone is split, so that \( RP_i \) does not belong to it anymore, it informs the newly joined node \( M \) about it. Then, node \( M \) updates the neighbor list of the corresponding \( L_i \).

When routing towards a point \( X \) in space, node \( N \) compares the distance between the destination and all of its usual and long range neighbors (of all levels). The query is forwarded to the neighbor, which is closest to \( X \).

5. CONCLUSION AND FUTURE WORK
By using the Reference Point method, lookup paths can be significantly shortened, since with a small number of steps we can get very close to the target. Furthermore, fairness is achieved with regard to lookup path lengths: paths are of comparable length regardless of source and destination. The excellent scalability and robustness properties of the CAN concept are not corrupted, since even multi-level RPs bring only a small constant number of states per node to the system.

This paper describes a work in progress, hence there are a number of tasks ahead. Implementation of the optimized algorithm is in progress. Simulations are necessary to evaluate the performance of the proposed method with regard to the average path length of a lookup operation, the tradeoff of using multiple RPs at multiple levels and the fairness achieved. All the numbers should be compared to the original CAN algorithm, with and without realities. An analytic study of the RP concept is also planned involving n-dimensional coordinate spaces.

6. REFERENCES
