P2P-based Routing and Data Management using the Virtual Cord Protocol (VCP)

Abdalkarim Awad, Reinhard German and Falko Dressler
Computer Networks and Communication Systems
Department of Computer Science, University of Erlangen, Germany
{abdalkarim.awad,german,dressler}@informatik.uni-erlangen.de

ABSTRACT

We present Virtual Cord Protocol (VCP), a virtual relative position based routing protocol for sensor networks that also provides methods for data management as known from standard DHT services. Self-organizing and cooperative algorithms are thought to be the optimal solution to overcome the inherent resource limitations in sensor networks. On an abstract level, DHT techniques offer O(1) complexity data lookup. Unfortunately, they usually rely on underlayer routing techniques. The key contributions of VCP are the independence of real location information by relying on relative positions of neighboring nodes, successors and predecessors in the cord are always in their vicinity, and the high scalability because only information about direct neighbors are needed for routing. Furthermore, VCP inherently prevents dead-ends and it is easy to be implemented.

Categories and Subject Descriptors: C.2.2 [Computer-Communication Networks]: Network Protocols

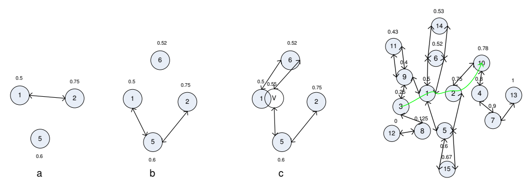
General Terms: Algorithms, Performance

1. INTRODUCTION

Wireless Sensor Networks (WSNs) are demanded to operate well facing strong resource limitations in terms of processing power or energy as well as high dynamics introduced by joining and leaving nodes [1]. In addition to ad hoc routing in the network, efficient data management is essential.. Both, centralized approaches (bottleneck, single point of failure) and flooding techniques (energy constraints) fail to provide the necessary services. Distributed Hash Tables (DHTs) are used to associate data items to node in the network. Usually, DHTs are built on the application layer and rely on an underlying routing protocol that provides connectivity between the nodes. Implementing DHTs in WSNs as an overlay and relying on ad hoc routing protocols has the drawback that these routing protocols already need to maintain valid topology information. Virtual coordinate routing protocols are thought to solve this problem.

Related work includes Virtual Ring Routing (VRR) [2], which uses a location independent integer to identify nodes. All nodes are organized into a virtual ring in order of increasing keys. Each node maintains a set of virtual neighbors of cardinality r that are nearest to node identifier in the virtual ring. In the hop id routing scheme [3], each node maintains a hop id, which is a multidimensional coordinate based on the distance to some landmark nodes. Fundamentally, landmarks can be randomly selected in the network.

MobiHoc'08, May 26–30, 2008, Hong Kong SAR, China.
ACM 978-1-60558-073-9/08/05.



(a) Creating virtual nodes (b) Routing

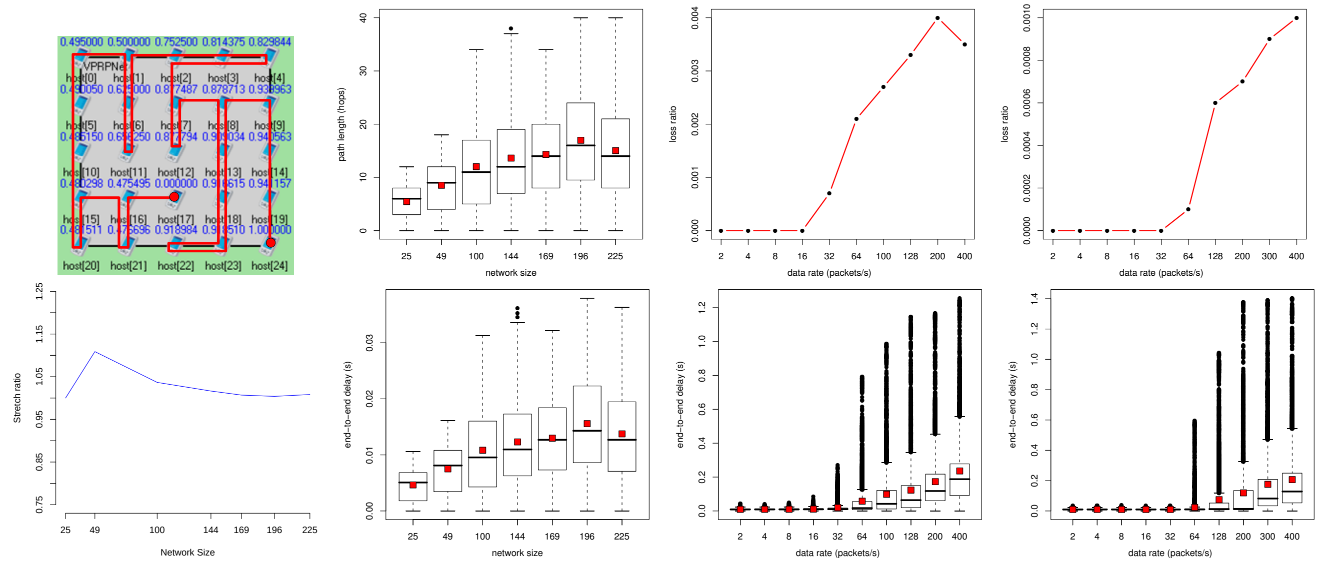
Figure 1: Basic VCP operations

2. VIRTUAL CORD PROTOCOL

Virtual Cord Protocol (VCP) is a DHT-like protocol that offers in addition to standard DHT functions an efficient routing mechanism. The key characteristics of VCP are the geographical vicinity of virtual neighbors, which reduces the communication load, VCP only needs information about direct neighbors for routing, and it cannot stuck with dead-ends. Finally, the protocol is easy to be implemented on the top of a typical MAC layer. All data items are associated with numbers in a pre-defined range $[S, E]$ that is captured by the available nodes. The basic join procedure is shown in Algorithm 1. Each joining node discovers all neighboring nodes by local broadcasting. In this step, it collects the virtual position of physical neighbors as well as two hop neighborhood information. By definition, the first node has

Algorithm 1 VCP join algorithm

if $Neighbour = S$ (or E) **then**
  $MyPosition \leftarrow S$ (or E)
  $Successor$ (or $Predecessor$)$\leftarrow MyNeighbour$
  SendUpdate($MyNeighbour$)
  return
end if
if $MyNeighbour1$ is predecessor to $MyNeighbour2$ **then**
  $MyPosition \leftarrow$ (a number between the two positions)
  $Successor \leftarrow MyNeighbour1$
  $Predecessor \leftarrow MyNeighbour2$
  SendUpdate($MyNeighbour1, MyNeighbour2$)
  return
end if
SendCreatVirtualNode($MyNeighbour$)



(a) Setup (top) and stretch ratio for different network sizes

(b) Number of hops (top) and end-to-end delay (bottom) for varying network size

(c) Loss ratio (top) and end-to-end delay for bursty traffic behavior

(d) Loss ratio (top) and end-to-end delay for constant traffic rate

Figure 2: Simulation setup and preliminary results. Most figures are drawn as boxplots indicating the mean value and the quartiles. The small box shows the average value

the smallest value S of the entire range and the second node gets the largest number E. All further joining nodes get a virtual position between two adjacent nodes. The new node becomes successor of the old node with the lower position and predecessor to the node that has the greater position. If the new node can communicate with only one node, then it asks that node to create a virtual node.

Figure 1(a) depicts the joining process. It is clear that the joining of a new node only affects a small number of nodes in the vicinity and it is independent of the total number of nodes in the network. In fact, the insertion of a new node only affects O(m), where m is the number of successors. Routing is based on knowledge of virtual and physical neighbors. A greedy algorithm is employed to send packets to the neighbor that has the closest position to the destination until there is no more progress and the value lies between the positions of the predecessor and successor. We assume that node failures can be detected if a node does no longer communicate. If an end node fails (i.e., either S or E), then the successor or predecessor gets the end position. If the successor and predecessor of the failed node can communicate directly (including its virtual nodes), the cord can easily be repaired. In the other case, the network needs to wait until either a new node can be used to rebuild the cord.

3. PRELIMINARY RESULTS

For a first analysis of the VCP protocol, we implemented a simulation model in OMNeT++. Figure 2(a) (top) shows the simulation setup for 25 nodes including virtual relative positions and the virtual cord connecting all the nodes. For the experiments, we varied the network size from 25 to 225 hosts and adapted the plane size to keep the density constant. Moreover, we studied two different traffic pattern. First, we evaluated bursty traffic, i.e. all messages were sent at a random time within 100 s. Secondly, a constant packet stream was analyzed to compare the protocol behavior under artificial traffic conditions. Figure 2(a) shows the stretch ratio as for varying network size. There is no stretch in case of 25 nodes. Using 49 nodes, it increased a bit but stays below 15%. For other network sizes, the path was almost optimal, which indicates that VCP uses optimal paths. Figure 2(b) shows the effect of network size on the performance of VCP. It is clear the path length increases with the network size in a logarithmic manner. However, there are few node which used a path length larger than the shortest path to reach the destination. More than 75% of the nodes have a path length smaller than l_{max}. In all experiments, the delay was proportional to the path length and the success ratio was 100%. To study the behavior of VCP under different traffic load, we kept the number of nodes constant (100). Each node sends 100 packets to the same destination. As shown in Figure 2(c), packets can be delayed at the MAC layer due to congestion. As a result, the end-to-end delay is increased. Nevertheless, the delay is still in an acceptable range. Below a rate of 16 packets/s, the effect of congestion is negligible. The effect of increasing traffic was not so big on the packet delivery rate. As can be seen from Figure 2(c), the loss ratio was below 0.4%. We repeated the same experiment using a constant packet rate. As shown in Figure 2(d), the results are a little bit better.

4. REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Elsevier Computer Networks*, vol. 38, pp. 393–422, 2002.

[2] M. Caesar, M. Castro, E. B. Nightingale, G. O'Shea, and A. Rowstron, "Virtual Ring Routing: Network routing inspired by DHTs," in *SIGCOMM 2006*, Pisa, Italy, September 2006.

[3] Y. Zhao, Y. Chen, B. Li, and Q. Zhang, "Hop ID: A Virtual Coordinate-Based Routing for Sparse Mobile Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 9, pp. 1075–1089, September 2007.