October 2019

DISTRIBUTED DESTINATION ADVERTISEMENT OBJECT (DAO) PROJECTION FOR PEER-TO-PEER ROUTING IN LOW POWER AND LOSSY NETWORKS (LLNS)

Huimin She

Pascal Thubert

Harbor Dong

Li Zhao

DISTRIBUTED DESTINATION ADVERTISEMENT OBJECT (DAO) PROJECTION FOR PEER-TO-PEER ROUTING IN LOW POWER AND LOSSY NETWORKS (LLNS)

AUTHORS:

Huimin She
Pascal Thubert
Harbor Dong
Li Zhao

ABSTRACT

Presented herein is a recursive method to find a common ancestor to a source node (source), a destination node (destination), and intermediate nodes in a constrained network, as well as to compute shortcut route(s) in order to offload the root of mundane routing computations. The techniques presented herein provide methods to filter neighbor information that is provided to a common ancestor node in order to enable these operation in a constrained network.

DETAILED DESCRIPTION

Increasingly, networking customers desire optimized Peer-to-Peer (P2P) routing in connection with, for example, smart utility and smart city applications. An optimal solution is obtained with route projection, which involves PCE that is either collocated with a root node (root) or behind the root in the Internet. This entails load near the root for control messages involved in route establishment, as well as high latency if many routes are to be computed at the same time and only the root performs the computation.

The techniques presented herein build on the observation that, in storing mode, a node (a subroot in non-storing) has as much knowledge about its subDAG as the root has about the entire network. This means that, if the root can compute a projected route between any pair of nodes in the DODAG, then the Routing Protocol for Low-Power and Lossy Networks (RPL) could also be used to compute any route within its subDAG. In general, P2P routes are usually local so that a parent that has enough knowledge (e.g., all hops in the subDAG) to build a path that is usually close (i.e., common ancestor such as common parent, grand parent or grand-grand parent).

That common ancestor is located close to the nodes, thus it is more aware of the topology changes than the root. The techniques presented herein select a parent to perform the projected DAO computation and address the problem of parent selection, as well as problems associated with obtaining the information that is made available to that parent, noting that it is probably a constrained node. The techniques presented herein are implemented in the RPL storing mode environment where the parent nodes are aware of their subDAG.

Certain variations can operate in non-storing mode leveraging subDAGs and some embodiments use relay nodes in a connected dominating set. Part of the concept of route projection is that the nodes advertise information about their neighbors (e.g., advertise neighbor information). However, conventional techniques do not address the selection of the appropriate neighbors.

As such, presented herein are techniques to compute and install, in a distributed manner, projected routes to the target nodes from within a DODAG tree. The goal is to offload the root that is usually congested, while benefiting from route projection that is simpler to implement and based on a more predictable model than conventional arrangements. The techniques presented herein leverage the concept of subDAGs and subroots and enable subroots to compute projected routes within their subDAGs. To achieve this, the techniques presented herein consider nested subDAGs and locate the common ancestor of the source and destination that has enough information to build the projected route.

The techniques presented are implemented in a RPL storing mode environment where the parents are aware of their subDAG. A variation can operate in non-storing mode leveraging subDAGs. In this variation, the nodes send DAO information to the subroot of the subDAG to which they belong (as opposed to the main root). The subroots can then recursively send all that information to the subroot above all the way to the root.

The techniques presented herein may be implemented as several steps. First, during the RPL tree building phase, a node places neighbor information, as well as parent information, into DAO messages. In order to avoid too many neighbors, it is possible to define rules to constrain the number of neighbors (e.g., limit to the peers that are members of a connected dominating set). In certain examples, only a maximum number of neighbors can be presented (e.g., set an RSSI threshold where only neighbors above the threshold are included in the DAO).

In the techniques presented herein, a neighbor "N" can be omitted in the DAO message to the parent (or subroot) if another neighbor "N1" or a set of other neighbors "N1... Np" provide connectivity to the all of N's neighbors (IOW if that neighbor does not bring additional connectivity). In order to make this determination, the techniques presented herein leverage the RPL multicast DAO (sent by MAC layer broadcast to all neighbors) to expose its neighbors to other neighbors. As a result, all nodes are aware of the complete neighbor information for a node (i.e., determine the "neighbors' neighbors").

The parent, ancestors (conversely the subroots towards the root in non-storing), and the root all store the neighbor information for computing the P2P path. The root can decide to enable neighbors in DAO based on the P2P requirement. If there is no P2P requirement, then the node will not include neighbor information in its DAO.

When a source node queries for a P2P path, the query is handled hierarchically. In particular, the node sends a P2P query to the parent. If the parent can compute a P2P path, it installs the P-DAO to the source. If the parent cannot compute a P2P path, it forwards the query upwards.

In a non-storing mode, the above description applies except that the term "parent" can be changed to "subroot."

In accordance with the techniques presented herein, even if a parent node computes the P2P path, it continues to forward the query upwards. If there is a better P2P path computed by an ancestor node, the P2P path is updated. When the root computes a P2P path for the source-destination (src-dst) pair, it pushes the P-DAO to the source and its parent and ancestor. Next time, the P-DAO can be obtained from its parent or ancestor.

Figure 1, below, illustrates an example implementation of the techniques presented herein. The operations of Figure 1 can be defined in four (4) steps:

Step 1: Source node first send P2P query to its parent node "I."

Step 2: Node I cannot compute the path based on its sub-DAG. So node I forwards the query to node "H."

Step 3: Node H computes the P2P path: source -> I -> J -> K -> destination. It pushes the route to the source node.

Step 4: Node H continues to forward the query upwards until reaching the root. Some of its parent, or the root, might compute a better path: source -> J -> K -> destination. Then the P2P path is updated accordingly.

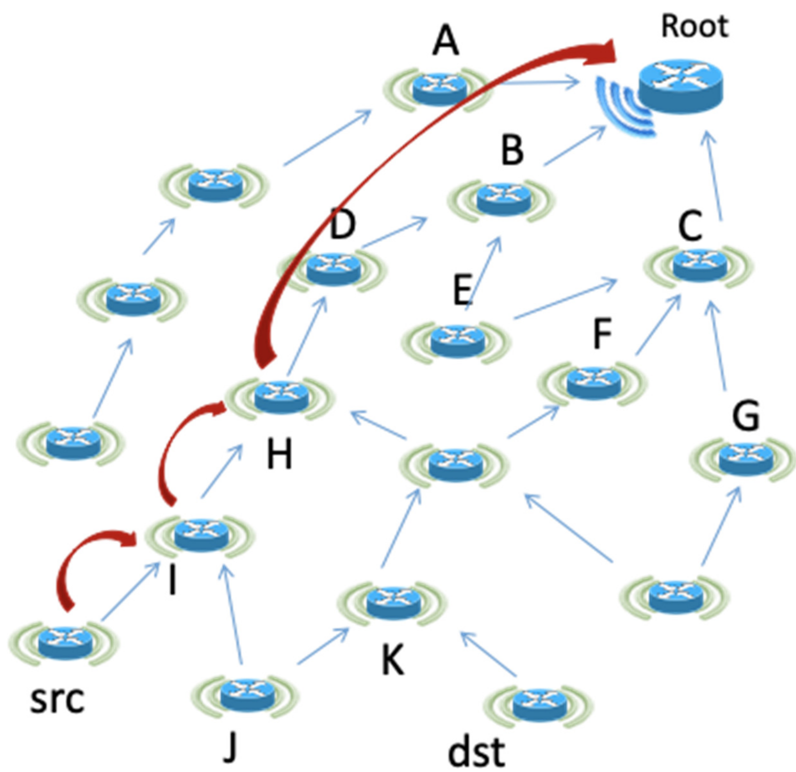

Figure 1