Self-Stabilizing Protocol for Shortest Path Tree for Multi-cast Routing in Mobile Networks*

Sandeep K.S. Gupta[1], Abdelmadjid Bouabdallah[2], and Pradip K Srimani[1,2]

[1] Department of Computer Science, Colorado State University, Ft. Collins, CO 80523 USA
[2] Universite de Technologie de Compiegne, Lab. Heudiasic, UMR CNRS 6599, Dep. Genie Informatique, BP 20529, 60205 Compiegne Cedex, France

Abstract. Our objective is to view the topology change as a change in the node adjacency information at one or more nodes and utilize the tools of self-stabilization to converge to a stable global state in the new network graph. We illustrate the concept by designing a new efficient distributed algorithm for multi-cast in a mobile network that can accommodate any change in the network topology due to node mobility.

Keywords: Self-stabilizing Protocol, Distributed System, Multi-cast Protocol, Fault Tolerance, Convergence, System Graph.

1 Introduction

Most of the protocols for designing near optimal multi-cast trees for given multi-cast groups in mobile ad hoc networks and analyzing their performance [AB96, CB94]. assume that the underlying network topology does not change. Recently we have proposed a self-stabilizing protocol for maintaining multi-cast tree in mobile ad hoc network which is based on pruning a minimum weight spanning tree [GS99]. This protocol minimizes the bandwidth requirement for multi-casting a message. In order to minimize the multi-cast latency a shortest-path tree can be employed. A shortest path tree rooted at node r is spanning tree such that for any vertex v, the distance between r and v in the tree is the same as the shortest-path distance in the entire graph. Our purpose in this short note is to show how a self-stabilizing algorithm for shortest path tree generation can be simply adapted to solve the problem of maintaining a hortest-path multi-cast tree in a radio network for a given multi-cast group. We analyze the time complexity of the algorithm in terms of number of rounds needed for the algorithm to stabilize after a topology change, where a round is defined as a period of time in which each node in the system receives beacon messages from all its neighbors.

* Address for Correspondence: Pradip K Srimani, Department of Computer Science, Colorado State University, Ft. Collins, CO 80523, Tel: (970) 491-7097, Fax: (970) 491-2466, Email: srimani@CS.ColoState.Edu

A. Bode et al. (Eds.): Euro-Par 2000, LNCS 1900, pp. 600–604, 2000.

2 Shortest Path Tree Protocol

We make the following assumptions about the system. (1) A data link layer protocol at each node i maintains the identities of its neighbors in some list $neighbors(i)$. This data link protocol also resolves any contention for the shared medium by supporting logical links between neighbors and ensures that a message sent over a correct (or functioning) logical link is correctly received by the node at the other end of that link. (2) Each node periodically (at intervals of t_b) broadcasts a *beacon* message. This forms the basis of *neighbor discovery protocol*. When a node i receives the beacon signal from a node j which is not in its neighbors list $neighbors(i)$, it adds j to its neighbors list (data structure $neighbors_i$ at node i), thus establishing link (i, j). For each link (i, j), node i maintains a timer t_{ij} for each of its neighbors j. If node i does not receive a beacon signal from neighbor j in time t_b, it assumes that link (i, j) is no longer available and removes j from its neighbor set. Upon receiving a beacon signal from neighbor j, node i resets its appropriate timer. (3) The topology of the ad-hoc network is modeled by a (undirected) graph $G = (V, E)$, where V is the set of nodes and E is the set of links between neighboring nodes. We assume that the links between two adjacent nodes are always bidirectional. Since the nodes are mobile, the network topology changes with time. We assume that no node leaves the system and no new node joins the system. (4) There is an underlying unicast routing protocol to send unicast messages between two arbitrary nodes in the network.

Each node $i \in V$ maintains a local variable $D_i(r)$; $D_i(r)$ is the current estimate of $S_i(r)$ known at node i and it determines the *local state* of node i. In addition, each node i also maintains a predecessor pointer P_i pointing to one of the nodes in $Adj(i)$; P_i points to the node adjacent to node i in the currently estimated shortest path from node i to node r. The set $\mathcal{N}(i)$ contains neighboring nodes of i that are on currently estimated shortest paths from node i to r. Each node i executes the following code:

$$\textbf{if } (i = r \wedge (D_i \neq 0 \vee P_i \neq NULL)) \textbf{ then } D_i = 0 \;\&\; P_i = NULL$$

$$\textbf{else if } (i \neq r \wedge (D_i(r) \neq \min_{\forall j \in Adj(i)} (D_j(r) + w_{ij}) \vee P_i \notin \mathcal{N}(i))$$

$$\textbf{then } D_i(r) = \min_{\forall j \in Adj(i)} (D_j(r) + w_{ij}) \;\&\; P_i = k, k \in \mathcal{N}(i)$$

2.1 Complexity Analysis

In case of a mobile ad hoc network, where the SPST protocol is used to maintain a multi-cast routing tree, it is required that the protocol converges as quickly as possible and it is also true that the participating mobile clients (nodes) do not act in a adversarial way i.e., they make their moves according to some known uniform protocol (i.e., each node sends its state to its neighbors at regular intervals). The purpose of this section is to provide an analysis of the convergence time of the proposed protocol under the

assumptions of the ad hoc network model. Each node periodically broadcasts a beacon message to its neighbors and this period is same for each node in the system. Let us define a *round* of computation as the time between two consecutive beacon message broadcast (i.e. the period of beacon message broadcast). Thus, in each round, every node that is privileged due to the actions taken by the nodes during the immediate past round, will make a move to make the node locally stable. Let $\mathcal{D}$ denote the diameter of the underlying network with uniform edge weights (i.e. when each edge is assigned an uniform weight of 1); and m_1 and m_2 denote respectively the minimum and the maximum edge weight in the system graph.

Remark 1. The ratio $\lceil \frac{m_2}{m_1} \rceil$ plays a very important role in determining the convergence time of the protocol. In an adversarial oracle this ratio can be very large (ratio of the largest real number to the smallest positive non zero real number that can be stored), while in the context of an ad hoc network, the ratio (range of link costs) would be small. For example in "revised ARPANET routing metric" the most expensive link is only seven times the cost of the least expensive link. The reason being that there is a relationship link cost and link utilization. A link which has a very low cost gets overly utilized since it is included in many shortest path whereas a link with a very high cost has an extremely low utilization since it hardly gets included in any path. Analysis of Internet packet traces show that, if the range of link cost is very wide, say 1 to 127, a high percentage of network traffic is destined for a small number of network links. This results is overall very poor utilization of the network. In mobile ad hoc network this can aggravate the problem of bandwidth scarcity even further. Hence, in mobile ad hoc networks the ratio of link of most expensive to least expensive link is expected to be very small.

Lemma 1. *Starting from a given illegitimate state consider the system state after p rounds; each of the nodes that are yet to be stabilized has $D_i(r) \geq pm_1$.*

Lemma 2. *Consider a node i which is p hops away from root r (the shortest path from i to r may involve more edges); node i will be stabilized in at most $p\lceil \frac{m_2}{m_1} \rceil$ rounds, after the node r is stabilized.*

Lemma 3. *The upper bound on the number of rounds needed by the entire network to stabilize starting from an arbitrary illegitimate state is given by $\mathcal{D} * \lceil \frac{m_2}{m_1} \rceil + 1$.*

3 Multi-cast Protocol

Our protocol for fault tolerant maintenance of the multi-cast tree for a given source node (we call it root node r) and its multi-cast group consists of 2 logical steps: (1) construction of the shortest path spanning tree of the mobile network graph in presence

of the topology change due to node mobility (establishing unique parent pointer for each node in the SPST); (2) pruning from the SPST the nodes that are not needed to send the message to the multi-cast group members. The protocol described in the previous section maintains the shortest path spanning tree in a fault tolerant way (that accommodates the topology change due to node mobility) as well as maintains the knowledge of the tree in a distributed way; each node knows its unique parent pointer). In this section we describe the protocol to prune the SPST and build the multi-cast tree.

The multi-cast source node r needs to send the message to the members of the arbitrary multi-cast group. Each node in the network knows whether it is a member of the multi-cast group (IS_Member_i is true). Note that even if a node is not a member of the multi-cast group, it will need to transmit the message to its successors iff any of its successors belong to the multi-cast group. In the rooted SPST, each node i can determine if it is a leaf node in the SPST (it has no neighbor node j such that $P_j = i$; in this case, node i will set $Flag_i$ variable to 1 if IS_Member_i is true and to 0 otherwise. Any other node i (i is not a leaf node in the SPST) will look at all its successors in the SPST and will set its $Flag_i$ to 1 iff at least one of its successors either has a Flag of 1 or is a member of the multi-cast group or node i is a member of the the multi-cast group. After this process stabilizes, each node i, when it receives the multi-cast message from its parent in the tree, knows that it needs to forward the message to its successors if $Flag_i$ is 1. Note that the nodes with $Flag_i$ value 1 constitute the multi-cast tree (although not all the nodes in the multi-cast tree are necessarily members of the multi-cast group). Now we can state the complete protocol to maintain the multi-cast tree:

$$\textbf{SPST} \begin{cases} \textbf{if } (i = r \wedge (D_i \neq 0 \vee P_i \neq NULL)) \textbf{ then } D_i = 0 \ \& \ P_i = NULL \\ \textbf{else if } (i \neq r \wedge (D_i(r) \neq \min_{\forall j \in Adj(i)}(D_j(r) + w_{ij}) \vee P_i \notin \mathcal{N}(i)) \\ \qquad \textbf{then } D_i(r) = \min_{\forall j \in Adj(i)}(D_j(r) + w_{ij}) \ \& \ P_i = k, k \in \mathcal{N}(i) \end{cases}$$

$$\textbf{Multi-cast Tree} \begin{cases} \textbf{if } Flag_i \neq \vee_{k \in Adj(i)}((P_k = i) \wedge (IS\_Member_k \vee Flag_k)) \\ \qquad \textbf{then } Flag_i = \vee_{k \in Adj(i)}((P_k = i) \wedge (IS\_Member_k \vee Flag_k)) \end{cases}$$

Lemma 4. *Starting from any illegitimate state the protocol correctly sets the Flag for each node which is a member of the multi-cast group, in at most $n - 1$ rounds after the SPST protocol has stabilized. Starting from any illegitimate state, the entire protocol stabilizes to a valid multi-cast tree in at most $\mathcal{D} * \lceil \frac{m_2}{m_1} \rceil + n$ rounds.*

References

[AB96] A. Acharya and B. R. Badrinath. A framework for delivering multicast messages in networks with mobile hosts. *ACM/Baltzer Journal of Mobile Networks and Applications*, 1:199–219, 1996.

[CB94] K. Chao and K. P. Birman. A group communication approach for mobile communication. In *Proc. Mobile Computing Workshop*, Santa Cruz, CA, December 1994.

[GS99] S. K. S. Gupta and P. K. Srimani. Using self-stabilization to design adaptive multicast protocol for mobile ad hoc networks. In *Proc. DIMACS Workshop on Mobile Networks and Computing*, Rutgers University, NJ, March 1999.

[Kar72] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*. Plenum Press, New York, 1972.