A Recursive, Distributed Minimum Spanning Tree Algorithm for Mobile Ad Hoc Networks

Omur Arslan
Department of Electrical and Systems Engineering, University of Pennsylvania
Philadelphia, PA 19104
Email: omur@seas.upenn.edu

Daniel E. Koditschek
Department of Electrical and Systems Engineering, University of Pennsylvania
Philadelphia, PA 19104
Email: kod@seas.upenn.edu

***Abstract*—We introduce a recursive ("anytime") distributed algorithm that iteratively restructures any initial spanning tree of a weighted graph towards a minimum spanning tree while guaranteeing at each successive step a spanning tree shared by all nodes that is of lower weight than the previous. Each recursive step is computed by a different active node at a computational cost at most quadratic in the total number of nodes and at a communications cost incurred by subsequent broadcast of the new edge set over the new spanning tree. We show that a polynomial cubic in the number of nodes bounds the worst case number of such steps required to reach a minimum spanning tree and, hence, the number of broadcasts along the way. We conjecture that the distributed, anytime nature of this algorithm is particularly suited to tracking minimum spanning trees in (sufficiently slowly changing) mobile ad hoc networks.**

I. Introduction

A mobile ad hoc network (MANET) is a wireless, self-organizing, distributed communications infrastructure consisting of mobile nodes required to maintain active channels supporting arbitrary pairwise messages at all times. Because of their decentralized and dynamic nature, routing MANET communication channels presents a longstanding and challenging problem that has often been addressed by hierarchical protocols in view of their scalability and efficiency for large-scale networks [1].

In this preliminary work, we address the problem of maintaining such a hierarchical routing protocol, assuming a very specific choice of *clusterhead* (the communications hub assigned to a nested group of nodes) and resulting *backbone* (the connected graph across which these clusterheads communicate with each other): the minimum spanning tree (MST) associated with a single-linkage clusterhead hierarchy [1]. Because our approach derives from a family of anytime hierarchy-restructuring algorithms encompassing a broader variety of commonly applied cluster linkage methods [2], we believe that the present results should generalize similarly. This extended abstract continues in Section II with a statement of the specific problem addressed and its relation to the existing literature. Section III summarizes the principal technical results, and Section IV offers a brief summary and appraisal of next steps.

II. Problem Statement, Solution, and Prior Work

Let $G = (V, E, W)$ be a weighted, undirected connected graph.[1] Suppose that each node $v \in V$ is equipped with: (i) a data structure representing the (same, initial) spanning tree $S_0 = (V, E_{S_0}, W_{S_0})$ of G; (ii) a list of its immediate neighbors and the associated edge weights in G. Then we seek a distributed recursive minimum spanning tree policy, initiated at an arbitrary, a priori designated "active" node $a_0 \in V$, whose next, $k+1$ step, follows from step $k \in \mathbb{N}$ with the following properties:

- *Local Optimization:* at each step, k, node a_k computes a locally optimal restructuring of S_k towards another spanning tree S_{k+1} that has a lower edge weight sum;
- *Broadcast:* at each step, k, agent a_k broadcasts to all nodes the local edge insertions and deletions that transform S_k into S_{k+1} (along this new spanning tree) and then activates an adjacent node, $a_{k+1} \in V$.

A. Our Problem Solution

We present in Table II a solution to the problem just posed. It relies on the locally optimal restructuring policy of Table I, entailing insertions and deletions of edges incident only upon the active agent, a_k who can thus implement its broadcast along the new tree, S_{k+1}. The computation of the next agent is specified by means of a token circulation protocol discussed below. Clearly, equipped with such a policy at each step, k, there is one and only one active node, $a_k \in V$. We show in Theorem 1 that the recursion halts at a MST exactly when every node has been activated at least once, incurring $\mathrm{O}(|V|\,|E|)$ worst case computational and communications costs.[2]

B. Our Contribution Relative to Prior Literature

The Local Optimization step of our problem is related to a local version of the minimum spanning tree verification problem which can be solved in linear time, $\mathrm{O}(|V|)$ [4]. The node activation process in the Broadcasting step can be solved

[1] In the intended application this graph will typically be time varying, as discussed in Section IV. However, our formal convergence results and associated communications and computational costs are stated with respect to a putative fixed graph as described here.

[2] Here, $|X|$ denotes the cardinality of set X

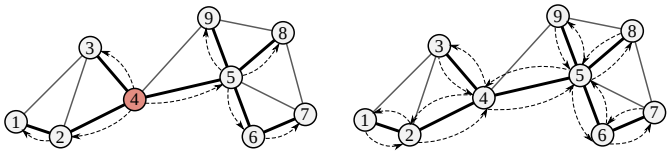

Fig. 1. (left) Broadcasting (dashed) using a spanning tree (thickened) of a connected graph from a node, 4. (right) Token circulation using Euler Tour (dashed) of a spanning tree (thickened), which follows nodes 1-2-4-5-6-7-6-5-8-5-9-5-4-3-2 in circular order.

using a standard token circulation strategy among nodes that guarantees mutual exclusion, i.e. only one node can be active at a time. We employ a token circulation protocol along a spanning tree of G by embedding a ring using its Euler tour [3], as illustrated in Figure 1. Let $\mathtt{TC}_S : E_S \to V$ denote the token circulation order along an Euler tour of a spanning tree $S = (V, E_S, W_S)$ of G.

III. Anytime Distributed MST Algorithm

Before proceeding with our distributed algorithm, we need to introduce some notation. Recall that a spanning tree $S = (V, E_S, W_S)$ defines a unique path between any pair of nodes. Let $P_S(u, v)$ denote the path joining node u and v in a spanning tree S, and $d_S(u, v)$ denote the hop distance between u and v defined as the maximum hop length along $P_S(u, v)$,

$$d_S(u, v) := \max_{(i,j)\in P_S(u,v)} W_S(i, j)\,, \tag{1}$$

and let $e_S(u, v)$ be an edge in $P_S(u, v)$ with the maximum hop length,

$$e_S(u, v) := \arg\max_{(i,j)\in P_S(u,v)} W_S(i, j)\,. \tag{2}$$

Note that we only need the hop distance d_S of an active node to all other nodes, which can be computed in linear time, $\mathrm{O}(|V|)$, using any tree traversal method or Dijsktra's algorithm. It is also convenient to have $N_G(u)$ be the ordered set of adjacent nodes of node u in $G = (V, E, W)$ in ascending order according to the associated edge costs, i.e.

$$N_G(u) := \big(v \in V \big| (u, v) \in E\big) = \big(v_1, v_2, \ldots, v_{\deg_G(u)}\big)\,, \tag{3}$$

$$W(u, v_i) \le W(u, v_j) \iff i \le j, \tag{4}$$

where $\deg_G(u)$ denotes the degree of node u in G. Finally, let $G\big|_F := (V, F, W_F)$ denote the subgraph of graph $G = (V, E, W)$ with edge set $F \subseteq E$ and edge weights $W_F(u, v) = W(u, v)$ for all $(u, v) \in F$.

Given a spanning tree $S = (V, E_S, W_S)$ of a connected graph $G = (V, E, W)$, Table I describes how to compute the locally optimal restructuring of S at a given node $u \in V$ only using its local connectivity in G.

TABLE I
Locally Optimal Spanning Tree Restructuring Policy
$(D, I) = \mathrm{LORE}(S, u)$

Let $S = (V, E_S, W_S)$ be a spanning tree of a connected graph $G = (V, E, W)$, and $u \in V$ be the active node. Then, the set of locally optimal edge deletions D and insertions I of S are found as follows:

- Begin with $D = \emptyset$, $I = \emptyset$, and $S_1 = S$.
- For every $k \in [1, \deg_G(u)]$ and $v = N_G(u)_k$, if $W(u, v) < d_{S_k}(u, v)$, then

$$D = D \cup \big\{e_{S_k}(u, v)\big\}\,, \tag{5}$$

$$I = I \cup \{(u, v)\}\,, \tag{6}$$

$$S_{k+1} = G\big|_{(E_S \cup I)\setminus D}. \tag{7}$$

Lemma 1. *The locally optimal restructuring of any spanning tree $S = (V, E_S, W_S)$ of a connected graph $G = (V, E, W)$ at a node $u \in V$ in Table I can be computed at most in $\mathrm{O}(|V| \deg_G(u))$ time.*

With the methods of this local restructuring and the token circulation along the Euler tour now in place, Table II formalizes our anytime distributed MST algorithm that guarantees the maintenance of a shared spanning tree.

TABLE II
Anytime Distributed MST Algorithm

Let $S = (V, E_S, W_S)$ be a spanning tree of graph $G = (V, E, W)$ shared by all nodes in V, and $a_0 \in V$ be the initial active node that is assumed to receive a token from $a_{-1} \in N_G(a_0)$.

- Begin with $S_0 = S$.
- For every $k \in \mathbb{N}$ until the activation counter[3] reaches $|V|$
  - Find locally optimal restructuring of S_k at node a_k, $(D_k, I_k) = \mathrm{LORE}(S_k, a_k)$ (Table I).
  - Update $S_{k+1} = G\big|_{\left(E_{S_k} \cup I_k\right)\setminus D_k}$.
  - Broadcast the found updates (D_k, I_k), with associated edge weights, through S_{k+1}.
  - Wait until every node receives the updates.[4]
  - Pass the token to the next node, $a_{k+1} = \mathtt{TC}_{S_{k+1}}(a_{k-1}, a_k)$.

Theorem 1. *The anytime distributed MST algorithm in Table II initiated from any spanning tree of a graph $G = (V, E, W)$ terminates after each agent has been activated at least once with a MST of G using at most $\mathrm{O}(|V|\,|E|)$ messages in $\mathrm{O}(|V|\,|E|)$ time.*

IV. Conclusion

In this paper we introduce a simple anytime distributed MST algorithm for a fixed network with the intent to apply it to online tracking of MSTs in dynamic networks. The recursive nature of this algorithm lends itself to realtime dynamic settings by permitting nodes to stop routing management at any time and resume arbitrarily later. Work presently in progress includes the performance evaluation of the proposed algorithm in both simulations and experiments. In the longer term, we are planning to generalize this proposed algorithm to a broader class of adaptive hierarchical routing protocols.

Acknowledgments

This work was funded in part by the Air Force Office of Science Research under the MURI FA9550-10-1-0567.

References

[1] Mehran Abolhasan, Tadeusz Wysocki, and Eryk Dutkiewicz. A review of routing protocols for mobile ad hoc networks. *Ad Hoc Networks*, 2(1):1 – 22, 2004.

[2] O. Arslan and D. E. Koditschek. Anytime hierarchical clustering. Technical report, University Of Pennsylvania, 2014. URL http://arxiv.org/abs/1404.3439.

[3] Mario Baldi and Yoram Ofek. A comparison of ring and tree embedding for real-time group multicast. *IEEE/ACM Transactions on Networking (TON)*, 11(3):451–464, 2003.

[4] V. King. A simpler minimum spanning tree verification algorithm. *Algorithmica*, 18(2):263–270, 1997.

[3] Here, the "activation counter" is a register passed from node to node which is incremented by each node only upon its first activation.

[4] This is required to guarantee the maintenance of a shared spanning tree and is achieved by passing an acknowledgement message towards the active node along the spanning tree.