Building Robust Spanning Trees in Free Space Optical Networks

In Keun Son, Sunjai Kim, and Shiwen Mao
Department of Electrical and Computer Engineering
Auburn University, Auburn, AL 36849-5201
Email: soninkeun@auburn.edu, szk0041@auburn.edu, smao@ieee.org

***Abstract*—Free space optical (FSO) technology has been considered as a viable solution for broadband wireless networks. It has high potential for military communications for the next decade. In this paper, we investigate the problem of building robust spanning trees for FSO networks. We adopt *algebraic connectivity* as robustness measure, and formulate a 0-1 integer linear programming (ILP) problem. We present a fragment selection and merging (FSM) algorithm, which is executed in a distributed and asynchronous fashion to iteratively merge fragments until a spanning tree is formed. Our simulation study shows that FSM can achieve significantly improved connectivity at the cost of slightly degraded average link weight performance, compared to prior approaches. FSM can be used not only for FSO network bootstrapping, but also for auto-reconfiguration in response to network dynamics during operation.**

I. Introduction

Network centric warfare (NCW) is a new theory to share information and enhance collaboration in operations through robust networks. Robustly networked forces can be aware of current combat situation and manipulate shared information. Ultimately, mission operation can be dramatically improved. The global information grid (GIG) is a communication project of the United States Department of Defense (DOD) to globally interconnect military forces [1]. There is a compelling demand for broadband wireless communications for military applications.

Free space optics (FSO) has been a viable technology for broadband communications. FSO transmits optical data signals through free space at high bit rates. It has high potential for military applications [1], [2]. Besides broadband capacity, FSO links are immune to electromagnetic interference (EMI). They are secure with low probability of interception and low probability of detection (LPI/LPD) properties [3]. However, FSO link performance is degraded by turbulent atmospheric conditions. There has been considerable research on mitigating the impact of turbulent atmosphere [2]–[4]. It is important to design and optimize FSO network topology to achieve rich connectivity, and to make it auto-reconfigurable to adapt to bad weather conditions.

In this paper, we investigate the problem of building robust spanning trees in FSO networks. The problem is usually associated with the *bootstrapping* process: when isolated FSO nodes are powered on, an effective algorithm is needed to form a connected topology. The problem may also involve *reconfiguration* of FSO networks: when nodes or links fail, the network may be partitioned; an effective algorithm is needed to adapt to such dynamics to maintain a connected network during operation. In both cases, a distributed algorithm is needed since there is no network connectivity in the first place and there is no global information that is available for centralized control.

In a recent work [3], a distributed bottom-up (BU) bootstrapping algorithm was presented to form a spanning tree from isolated FSO nodes. BU can construct a spanning tree with maximal node degree at most one larger than that in the optimal solution. However, BU assumes that all participating nodes work synchronously and does not consider link quality when building the tree. Such synchronous execution may be hard to achieve due to the lack of network connectivity. Furthermore, strong links should be preferred when constructing the spanning tree to obtain robust networks.

In this paper, we adopt *algebraic connectivity* from spectral graph theory as measure of network robustness [5]. Motivated by the observation that networks with the same set of nodes and same amount of links can have quite different algebraic connectivity, we aim to build spanning trees with maximized algebraic connectivity. The formulated problem considers FSO link quality, node degree constraints, and algebraic connectivity, and is NP-hard. For competitive solutions, we develop a distributed *fragment selection and merging* (FSM) algorithm that keeps on merging network fragments, starting from isolated nodes, until a spanning tree is formed. Each fragment executes FSM independently to seek the "best" neighbor to merge to with no need of synchronized execution, where "best" means the largest algebraic connectivity after merging.

The proposed FSM algorithm is evaluated with simulations. Compared with the minimum weight spanning tree (MST) algorithms [6], the proposed algorithm can achieve significantly improved connectivity at the cost of slightly degraded average link weight performance. FSM also outperforms BU [3] on both connectivity and average link weight performance. The FSM algorithm can be used not only for bootstrapping an FSO network, but also for auto-reconfiguring an FSO in response to network dynamics during operation.

The remainder of this paper is organized as follows. Related work are reviewed in Section II. We describe the system model and problem statement in Section III. The FSM algorithm is presented in Section IV. Our simulation results are presented in Section V. Section VI concludes this paper.

978-1-4244-8180-4/10/$26.00 ©2010 IEEE

II. Related Work

Building a spanning tree is one of the well-known problems in graph theory. A spanning tree is a connected and undirected graph with no loop and multiplicity over one. For graphs with weighted edges, MST algorithms have been well studied. The Prim's algorithm and Kruskal's algorithm are well-known centralized NST algorithms. In [6], Gallager, Humblet and Spira proposed a distributed MST algorithm assuming that each node knows the link weights of its neighbors. Aiming to minimize or maximize the weight sum of connected edges, each fragment seeks neighbor independently to connect to. Using link reliability as weights, it seems that a spanning tree with the maximum weight sum may also have rich connectivity. However, in this paper we will show that this is not always true. The weight sum does not seem tightly correlated with network robustness.

Recently, there have been several papers on the design [7]–[9] and (re)configuration [3], [10] of FSO networks. In these papers, integer linear programming (ILP) problems were formulated and various heuristic algorithms were proposed to provide sub-optimal solutions. In [11], Wang and Abouzeid investigated a hybrid radio-frequency (RF) and FSO multi-hop network and derived an upper bound on the per node capacity. In [3], the authors proposed a bottom-up (BU) algorithm for bootstrapping an FSO network. The objective is to form a spanning tree as quickly as possible. BU is a distributed and synchronous heuristic, where the executions are synchronized among the fragments and fragment selection is solely based on node IDs (without considering link quality). As mentioned in [3], the time synchronization problem has not been solved.

Algebraic connectivity is a useful tool from *spectral graph theory* [5], [12]. It has been used in several papers as measure of connectivity [7], [13]. In [13], a semidefinite program (SDP) formulation was presented for the relaxed problem. The authors proposed a greedy heuristic based on the Fiedler vector to grow a well-connected network. In our prior work [7], we study the problem of building a richly connected mesh topology for FSO networks, and a Greedy Edge-Appending (GEA) algorithm was used to iteratively inserts edges into a connected graph. Such greedy edge-insertion approach cannot be applied here since it starts from a spanning tree. It is also a centralized algorithm that may not be suitable for bootstrapping an FSO network with initially isolated nodes.

III. Problem Statement

A. Model Description

We consider an FSO network consisting of n base stations. Each base station works as a backbone router, aggregating traffic though its RF transceiver to/from mobile users, and relaying the aggregate traffic through wireless optical links. Due to cost and space concerns, each base station has a limited number of FSO transceivers. We assume that autonomous Pointing, Acquisition, and Tracking (PAT) technologies are incorporated, in order to seek available neighbors and maintain current connections. Thus location and channel information can be shared between two neighboring nodes.

We adopt the log-normal model to characterize fading under turbulent atmosphere [4]. Under such model, link reliability γ_{ij} is the probability that the intensity of received signal (I) is over than certain threshold (I_{th}). Letting I_0 be the average received intensity, γ_{ij} can be computed as

$$\gamma_{ij} = P(I \geq I_{th}) = \frac{1}{2} - \frac{1}{2}\ \text{erf}\left(\frac{\ln(I_{th}/I_0)}{2\sigma_X\sqrt{2}}\right). \quad (1)$$

The ratio I_{th}/I_0 is determined by distance and absorption coefficient. γ_{ij} also depends on the standard deviation σ_X, which is strongly influenced by weather condition and transmission distance. An edge satisfying $\gamma_{ij} \geq \gamma_{th}$ becomes a *candidate link* for constructing the FSO network topology.

An FSO network can be modeled as a *simple* graph $G(V, E)$, where V is the set of vertices and E is the set of edges. Each vertex $v \in V$ represents an FSO base station and each edge $e \in E$ represents a wireless optical link between two distinct nodes. As in prior work [7], we assume that each FSO channel is full duplex. Due to line-of-sight transmissions with narrow beam divergence we assume symmetric link reliability, i.e., $\gamma_{ij} = \gamma_{ji}$ for $i, j \in V$. The link reliability is used as edge weight. The simple graph G is undirected and weighted.

B. Algebraic Connectivity Preliminaries

There are several graph connectivity measures, such as k-vertex/edge connectivity and bisection connectivity. We consider *algebraic connectivity* from spectral graph theory, which is a useful measure of network robustness [5]. The algebraic connectivity of a graph G is the second smallest eigenvalue of the Laplacian matrix $\mathbf{L}$ of the graph, denoted as $\lambda_2(\mathbf{L})$. It is a quantitative measure of graph connectivity: it is positive when the network has exactly one connected component, and remains zero when the network is partitioned. A higher $\lambda_2(\mathbf{L})$ value indicates a better connected topology.

Algebraic connectivity has close relation with various graph invariants. Consider a connected graph G that is not *complete*. There are n vertices, i.e., $|V(G)| = n$. Let $K_v(G)$ denote the vertex connectivity, $K_e(G)$ the edge connectivity, $\delta(G)$ the minimum degree of the graph, and D the graph diameter. The following inequality condition holds true [14]:

$$\frac{4}{D \cdot |V(G)|} \leq \lambda_2(\mathbf{L}) \leq K_v(G) \leq K_e(G) \leq \delta(G). \quad (2)$$

C. Algebraic Connectivity-based Topology Design

In this section, we formulate the spanning tree topology optimization problem as a 0-1 ILP problem. We assume that each base station can have at most κ transceivers due to cost or space constraints. During bootstrapping, each node first searches its neighbors using the PAT mechanism as described in [3], [10]. Each node then shares location and channel information with its identified neighbors. We now have a set of potential edges, denoted as E_{pot}. The spanning tree design problem is to choose an $(n-1)$-edge set E from E_{pot} such that (i) the unweighted degree of each node is no higher than κ and (ii) the algebraic connectivity $\lambda_2(\mathbf{L})$ of the resulting spanning tree is maximized.

We follow the Laplacian matrix definition in [12]. Let edge $l = \{i, j\} \in E$ be a link between nodes i and j with weight γ_l. Let $\mathbf{m}_l$ be an $\mathcal{R}^n$ vector, where $[\mathbf{m}_l]_i = 1$, $[\mathbf{m}_l]_j = -1$, and $[\mathbf{m}_l]_k = 0$ for all $k \neq i, j$. Then, the $n \times n$ Laplacian matrix $\mathbf{L}$ of a graph G is $\mathbf{L}(G) = \sum_{l=1}^{n-1} \gamma_l \cdot \mathbf{m}_l \cdot \mathbf{m}_l^T = \mathbf{M} \cdot \text{diag}(\vec{\gamma}) \cdot \mathbf{M}^T$, where $\text{diag}(\vec{\gamma}) \in \mathcal{R}^{(n-1)\times(n-1)}$ is a diagonal matrix with diagonal elements γ_l, $l = 1, 2, ..., n-1$. Note that $\mathbf{L}$ is a *positive semi-definite* matrix, which has n eigenvalues satisfying $\lambda_1(\mathbf{L}) \leq \lambda_2(\mathbf{L}) \leq ... \leq \lambda_n(\mathbf{L})$. Its smallest eigenvalue, $\lambda_1(\mathbf{L})$, is always zero, corresponding to eigenvector $\mathbf{1} = [1, 1, ..., 1]^T$. The second smallest eigenvalue $\lambda_2(\mathbf{L})$ is algebraic connectivity that we aim to maximize.

The problem of finding the spanning tree that maximizes algebraic connectivity can be formulated as follows.

$$\textbf{maximize} \quad \lambda_2(\mathbf{L}) \tag{3}$$

$$\textbf{subject to:} \quad d(v) \leq \kappa, \text{ for all } v \in V \tag{4}$$

$$E \subseteq E_{pot}, \ \ |E| = n - 1, \tag{5}$$

where $d(v)$ is the unweighted degree of node v.

For edge $l \in E_{pot}$, define an index variable x_l as:

$$x_l = \begin{cases} 1, & \text{if edge } l \in E \\ 0, & \text{otherwise,} \end{cases} \quad \text{for all } l \in E_{pot}. \tag{6}$$

We have a 0-1 vector $\mathbf{x} \in \{1, 0\}^{|E_{pot}|}$. The FSO spanning tree design problem can be reformulated into a 0-1 ILP problem as follows.

$$\textbf{maximize} \quad \lambda_2(\textstyle\sum_{l=1}^{|E_{pot}|} x_l \cdot \gamma_l \cdot \mathbf{m}_l \cdot \mathbf{m}_l^T) \tag{7}$$

$$\textbf{subject to:} \quad d(v) \leq \kappa, \text{ for all } v \in V \tag{8}$$

$$\mathbf{1}^T \cdot \mathbf{x} = n - 1 \tag{9}$$

$$\mathbf{x} \in \{1, 0\}^{|E_{pot}|}. \tag{10}$$

The optimal solution consists of a boolean vector $\mathbf{x}$ that achieves the maximum algebraic connectivity.

The formulated problem is NP-hard [15]. For small-sized networks, a centralized exhaustive search may be applied. Further, an edge-deleting greedy heuristic can also be applied to delete edges from a richly connected graph until a spanning tree is reached [7]. The challenge of using algebraic connectivity in building a tree stems from the fact that, as discussed, $\lambda_2(\mathbf{L})$ remains zero until a tree is formed. Furthermore, a centralized algorithm requires sharing link and topology information. Although this is possible when a spanning tree is already formed, it is not an easy task now since we aim to build a tree in the first place [3]. In the following section, we develop a distributed heuristic algorithm for the formulated problem, which achieves significantly improved algebraic connectivity without centralized control and global information.

IV. Fragment Selection and Merging Algorithm

The FSM algorithm is a distributed and asynchronous algorithm where starting from isolated nodes, each fragment seeks the "best" neighbor node or fragment and merge with it, until a spanning tree is formed. Since the objective is to form a tree with maximized robustness, each fragment identifies the "best" neighbor or fragment with respect to algebraic connectivity.

TABLE I
Fragment Selection and Merging (FSM) Algorithm

```
1:  Read current member information;
2:  Read neighbor information;
3:  IF (there exists a neighbor to be connected)
4:    Try to connect to a single-node neighbor:
5:      Search candidate edges;
6:      Rule out edges not satisfying degree constraint κ;
7:      Select the edge with the highest weight;
8:      IF (corresponding neighbor information not changed)
9:        Insert the selected edge;
10:     ELSE
11:       Update neighbor information;
12:       Go to Step 3.
13:     END
14:     Update member and neighbor information;
15:     Recalculate λ2 for the merged fragment;
16:     go to Step 3.
17:   Try to connect to a fragment neighbor:
18:     Search candidate edges;
19:     Rule out edges not satisfying degree constraint κ;
20:     Select the edge connecting to the fragment neighbor with
          the largest λ2;
21:     IF (corresponding neighbor information not changed)
22:       Insert the selected edge;
23:     ELSE
24:       Update neighbor information;
25:       Go to Step 3;
26:     END
27:     Update member and neighbor information;
28:     Recalculate λ2 for the merged fragment;
29:     Go to Step 3;
30:   IF (current member is not changed)
31:     Go to Step 33;
32:   END
33: ELSE
34:   Terminate with a spanning tree formed;
35: END
```

The FSM algorithm is presented in Table I. Each node first adopts a precise PAT mechanism to search for potential neighbors and exchanges location and channel information with identified neighbors. When two fragments (or nodes) are connected by inserting an FSO link to form a larger fragment, we assume all the information are shared among nodes within the same fragment [3]. The unweighted node degree is upper bounded by κ. The two key components, Fragment Selection and Merging, are discussed in the following.

A. Fragment Selection

Consider a tagged fragment that selects one of its neighboring fragment to merge to. Since the objective is to maximize algebraic connectivity, the fragment selects a neighbor that can achieve the highest connectivity for the merged subgraph. The following three cases could occur: (i) node-to-node merging, (ii) node-to-fragment merging, and (iii) fragment-to-fragment merging. In the following, we examine the algebraic connectivity of these cases and describe the fragment selection rules.

First, assume that two single nodes i and j are to be connected. They must be neighbor of each other and the edge weight is γ_{ij}. The resulting subgraph, denoted as T, consists of nodes i and j, and the weighted edge. The algebraic connectivity of T can be derived. Let $d'(i)$ be the weighted degree of node $i \in V$, defined as $d'(i) = \sum_i \gamma_{ij}$ for all $j \neq i$. We have $\sum_{i=1}^n \lambda_i = \sum_{i=1}^n d'(i)$ from spectral graph

theory [14]. That is, the sum of all the eigenvalues is equal to the sum of weighted degree of all nodes. Since λ_1 is zero for the connected subgraph, it follows that

$$\lambda_2(T) = \sum_{i=1}^{2} \lambda_i(T) = \sum_{i=1}^{2} d'(i) = 2\gamma_{ij}. \quad (11)$$

Second, consider the case that a fragment T_1 is to be merged with a node i. Let T' be the resulting new fragment. This is equivalent to adding a new pendant vertex with an edge to a tree. According to Corollary 1.1 in [16], $\lambda_2(T')$ is bounded as

$$\lambda_2(T') \leq \lambda_2(T_1). \quad (12)$$

Third, consider the case of merging two fragments T_1 and T_2 by adding an weighted edge to connect them. Let T'' denote the resulting new fragment. In [17], it is shown that the algebraic connectivity of T'' is bounded as

$$\lambda_2(T'') \leq \min\{\lambda_2(T_1), \lambda_2(T_2)\}. \quad (13)$$

Equations (11), (12), and (13) provide useful insights for fragment selection. Recall that all potential links satisfy $\gamma_{ij} \geq \gamma_{th}$ (see Section III-A). Usually $\gamma_{th} > 0.5$ to exclude weak links. Therefore, we have $\lambda_2(T) = 2\gamma_{ij} > 1$. On the other hand, the algebraic connectivity of a graph is less than the minimum node degree *if the graph is not complete* (i.e., not fully connected) [14]. Any tree with $n \geq 3$ is not complete and has at least one leaf node. Thus the corresponding $\lambda_2(T')$ or $\lambda_2(T'')$ should be less than the minimum node degree 1. The single-edge subgraph T always has higher connectivity than the other two subgraphs T' and T''. Consequently, when a node seeks a neighbor, an isolated neighbor with the *highest edge weight* is preferred over neighboring fragments.

When all the cases of node-to-node merging are done, then we consider the cases of node-to-fragment and fragment-to-fragment merging. Assume that the tagged fragment T_1 searches for a neighbor. From (12) and (13), the resulting new subgraph has an algebraic connectivity no larger than that of the original fragments. That is, the algebraic connectivity decreases as the tree grows. The strategy is to let each fragment find a neighbor that achieves the *smallest reduction* in algebraic connectivity. The tagged fragment T_1 will first search for isolated neighbor nodes and merge to the one with the highest edge weight. When there is no such isolated neighbor, the tagged fragment T_1 will choose a neighboring fragment with the highest algebraic connectivity to merge to.

B. Asynchronous Merging

During bootstrapping, each node or fragment executes the FSM algorithm independently to search for a neighbor node or fragment to merge to. Unlike prior work [3], there is no need to synchronize the executions at the FSO nodes.

As shown in Table I, each fragment executes FSM independently. When the "best" neighboring node or fragment is identified, the fragment requests to connect to the neighbor fragment. Before inserting a new link to connect the two fragments, FSM checks if the network state has been changed (e.g., whether the target fragment or node has been merged to another fragment in the meantime). If that is the case, the two fragments will not be merged and FSM is executed to identify the "best" fragment under the updated neighbor information. Otherwise, a link will be inserted to merge the the two fragments. The synchronization requirement with respect to algorithm operation can be greatly relaxed [3].

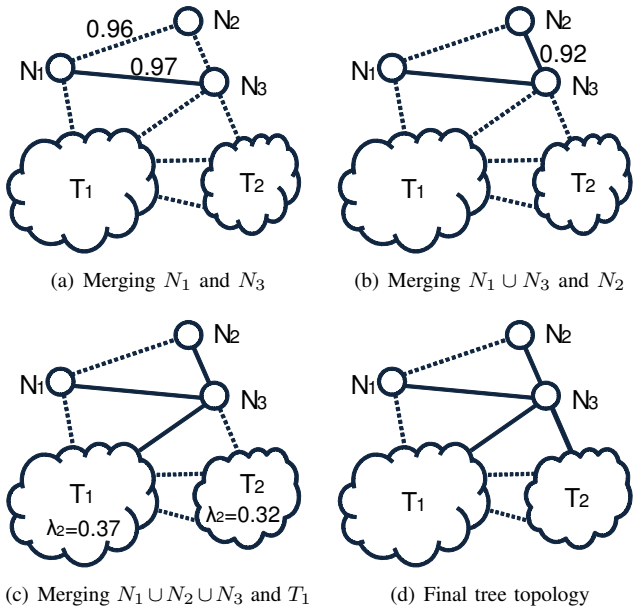

Fig. 1. An example illustrating the operation of the FSM algorithm.

C. Example

An example of FSM operation is given in Fig. 1. Assume that there are three node fragments, N_1, N_2, and N_3, and two tree fragments, T_1 and T_2, each having more than one nodes. We assume that all the merging procedures are executed in the fragment that containing N_1. In the beginning, N_1 has three candidate edges with two single-node neighbor, N_2 and N_3, and one fragment neighbor, T_1. Since FSM first seeks an isolated neighbor according to edge weight, N_1 first selects the edge with the highest weight 0.97 (i.e., the solid line between N_1 and N_3 in Fig. 1(a)). Next, the fragment consisting of N_1 and N_3 selects the edge between N_2 and N_3, since an isolated node neighbor is preferred than a fragment neighbor, as shown in Fig. 1(b). In the third iteration, there are two fragment neighbors to choose. Fragment T_1 is chosen since it has a higher algebraic connectivity value than fragment T_2, as shown in Fig. 1(c). Finally, the edge between N_3 and T_2 is chosen since T_2 is the last fragment to merge to. The algorithm stops and a spanning tree is formed.

D. Bootstrapping and Auto-reconfiguration

FSM can be used to bootstrap an FSO network and for auto-reconfiguration of an operating FSO network. As in [3], FSM can form a spanning tree when isolated FSO nodes are started. Since link quality and connectivity are explicitly considered, the resulting spanning tree has a richer connectivity compared to alternative approaches (see Section V).

During operation, there are three types of network dynamics: (i) one or more new FSO nodes are started, (ii) some

operating FSO nodes fail, (iii) some operating FSO links are broken, and (iv) the quality of some operating FSO links are changed (e.g., due to weather changes). The FSM algorithm can be executed to handle all these cases to to automatically reconfigure the network and to restore a robust spanning tree.

E. Computational Complexity

It is worth noting that the objective of BU is to obtain a connected graph as quickly as possible without caring about the weight or quality of the links. It has a low computational complexity $O(n \cdot |E|_{pot})$, where n is the number of FSO nodes and $|E|_{pot}$ is the number of potential edges [3]. The proposed FSM algorithm considers link quality when building spanning trees with better connectivity, but at the cost of comparatively higher computational complexity.

The proposed FSM algorithm works in a distributed and asynchronous fashion, so we consider its worst case complexity. The worst case occurs when the nodes are connected one at a time. The worst case complexity is $O(n^3 \cdot |E|_{pot})$. Specifically, finding the algebraic connectivity (i.e., finding eigenvalues of an Laplacian matrice) incurs most of the computations. There should be $(n-1)$ iterations in the worst case to build a tree. To determine the "best" neighbor at each iteration, FSM compares at most $|E|_{pot}$ algebraic connectivity values, each computed for an $n \times n$ matrix. According to [18], it takes $O(n^2)$ to compute eigenvalues of a real symmetric $n \times n$ matrix (i.e., Laplacian matrix in our case).

V. Simulation Studies

A. Simulation Setting

In this section, we present our simulation results to evaluate the performance of FSM and to provide a comparison with alternative schemes. In the simulations, we use two types of FSO networks: a 7-node network with a regular topology and randomly generated networks with various number of nodes. Specifically, we first use the 7-node FSO network to visually demonstrate the resulting spanning trees and link weights, and then study the more general case of random networks with sizes ranging from 20 to 50 in a rectangular region. To maintain a constant density, we enlarge the region as the number of nodes is increased.

We assume that each base station is stationary and can have at most five transceivers. Link connectivity between nodes is determined by link reliability, which is derived using the FSO channel model described in Section III-A. We set the reliability threshold as $\gamma_{th} = 0.9$. The potential edge set E_{pot} includes all the edges with a reliability value greater than the threshold.

We consider three algorithms in our simulations: (i) FSM, (ii) Prim's minimum spanning tree (MST) algorithm [6], and (iii) the bottom-up algorithm (denoted as BU) [3]. For comparison purpose, we use the algebraic connectivity of the resulting spanning tree graph as a measure of robustness. We also consider the average weight of the chosen links, which is an indication of the capability of choosing strong links. Since $\gamma_{th} = 0.9$, each link in E_{pot} has reliability ranging from 0.9 to 1, and thus the average weight is also within this range.

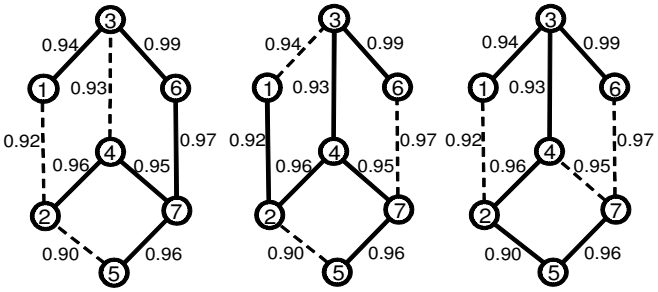

Fig. 2. Spanning trees formed by FSM, MST, and BU, respectively, for a 7-node FSO network.

B. Simulation Results

In Fig. 2, we plot the spanning trees formed by FSM, MST, and BU for the 7-node FSO network. For the 7-node network, six edges are chosen to build a spanning tree. As shown in Fig. 2, the maximum weight graph could not be well-connected with respect to algebraic connectivity. The algebraic connectivity values achieved by FSM, MST and BU are 0.3617, 0.2519, and 0.2112, respectively; the average edge weight achieved by FSM, MST and BU are 0.9517, 0.9617, and 0.9467, respectively. The spanning tree formed by FSM yields the largest algebraic connectivity (43.59% normalized improvement), at the cost of slightly reduced sum weight (a decrease of 0.01) as compared to MST.

In Fig. 3, we plot the algebraic connectivities for random networks with various sizes (n is increased from 20 to 50). Each point on the curves is the average of the results for 10 random topologies, with 95% confidence intervals plotted as error bars. It can be seen that networks with larger sizes have smaller algebraic connectivity, which is consistent with the analysis in Section IV-A. The average edge weights achieved by the three algorithms are plotted in Fig. 4, where each point is also the average of 10 random topologies and the 95% confidence intervals are plotted as well. We find that FSM can produce a robust spanning with greatly enhanced algebraic connectivity in all the cases examined. Furthermore, such improvement is achieved at the cost of slightly reduced average edge weight. For the random networks, on average, FSM achieves 71.26% normalized improvement on algebraic connectivity over MST. BU achieves 28.14% normalized improvement on algebraic connectivity over MST. The average edge weight over all cases achieved by FSM is 1.42% lower than that of MST, and the average edge weight over all cases achieved by BU is 4.31% lower than that of MST.

We find BU achieves better algebraic connectivity over MST. This is due to the fact that smaller graph diameter indicates better connectivity [14]. MST only considers the maximum weight edges among neighbors and expands the connected network using such edges, while BU can achieve smaller diameters than MST by merging neighboring fragments. Since BU considers neither link quality nor connectivity, it may choose edges with low weights close to γ_{th}. In Fig. 4, we find the BU average edge weight is approximately 0.95, which is considerably lower than that of the other two

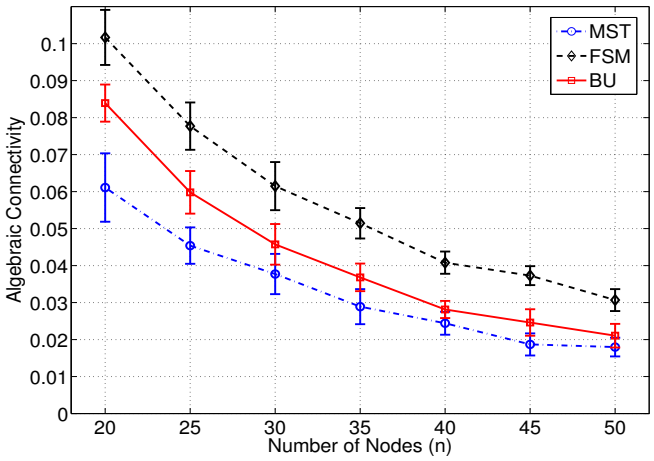

Fig. 3. Algebraic connectivity for random networks with various sizes.

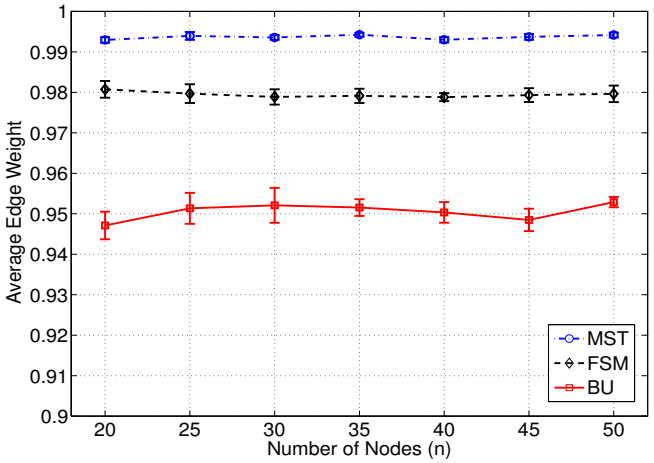

Fig. 4. Average edge weight for random networks with various sizes.

algorithms. Note that such gap will increase when γ_{th} is reduced (i.e., when edge weights are more diverse).

Finally, we present the execution times for the algorithms in Table II. The algorithms are implemented in MATLAB and executed in a desktop computer with an Intel core duo CPU 2.2 GHz and 2GB RAM. We assume that both FSM and BU algorithms work in the worst case when the nodes are connected one at a time. Compared with BU and MST, it takes more time to execute the FSM algorithm due to the need of solving eigenvalues. However, FSM is reasonably fast for practical systems: in the worst case, it takes about 1 sec to building a tree for a 50-node FSO network.

VI. Conclusion

In this paper, we studied the problem of building robust spanning trees for FSO networks. We adopted algebraic connectivity as measure of robustness and formulated a 0-1 ILP problem. For the NP-hard ILP problem, we developed a distributed FSM algorithm that can compute highly competitive solutions without synchronized execution. Our simulation studies showed that FSM can achieve significant improvements on algebraic connectivity at slightly reduced average edge weight, comparing to prior approaches. FSM can be used for both bootstrapping and auto-reconfiguration of FSO networks.

TABLE II
Comparison of Execution Times (sec)

Number of Nodes	MST	FSM	BU
20	0.003288	0.108970	0.031655
25	0.005526	0.176748	0.037470
30	0.009415	0.392533	0.040043
35	0.015275	0.566627	0.041535
40	0.024081	0.739200	0.062009
45	0.035776	0.865721	0.070993
50	0.051839	1.084987	0.089595

Acknowledgment

This work is supported in part by the National Science Foundation under Grants ECCS-0802113, IIP-1032002, and CNS-0855251, and through the Wireless Internet Center for Advanced Technology at Auburn University.

References

[1] P. Clark and A. Sengers, "Wireless optical networking challenges and solutions," in *IEEE MILCOM'04*, Monterey, CA, Oct./Nov. 2004, pp. 416–422.

[2] J. Juarez, A. Dwivedi, A. Mammons Jr., S. Jones, V. Weerackody, and R. Nichols, "Free-space optical communications for next-generation military networks," *IEEE Commun. Mag.*, vol. 44, no. 11, pp. 46–51, Nov. 2006.

[3] F. Liu, U. Vishkin, and S. Milner, "Bootstrapping free-space optical networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 12, pp. 13–22, Dec. 2006.

[4] X. Zhu and J. M. Kahn, "Free-space optical communication through atmospheric turbulence channels," *IEEE Trans. Commun.*, vol. 50, no. 8, pp. 1293–1300, Mar. 2003.

[5] F. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.

[6] R. G. Gallager, P. Humblet, and P. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Trans. Programming Languages Syst.*, vol. 5, no. 1, pp. 66–77, 1983.

[7] I. K. Son and S. Mao, "Design and optimization of a tiered wireless access network," in *Proc. IEEE INFOCOM'10*, San Diego, CA, Mar. 2010.

[8] ——, "A reformulation-linearization technique-based approach to joint topology design and load balancing in fso networks," in *Proc. IEEE GLOBECOM 2010*, Miami, FL, Dec. 2010, pp. 1–6.

[9] ——, "Fast heuristic algorithm for joint topology design and load balancing in fso networks," in *Proc. IEEE GLOBECOM 2010*, Miami, FL, Dec. 2010, pp. 1–6.

[10] A. Desai and S. Milner, "Autonomous reconfiguration in free-space optical sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 8, pp. 1556–1563, Aug. 2005.

[11] D. Wang and A. Abouzeid, "Throughput capacity of hybrid radio-frequency and free-space-optical (RF/FSO) multi-hop networks," in *Proc. 2007 ITA Workshop*, La Jolla, CA, Jan./Feb. 2007, pp. 1–8.

[12] S. Boyd, "Convex optimization of graph laplacian eigenvalues," in *Proc. International Congress of Mathematicians*, 2006, pp. 1311–1319.

[13] A. Ghosh and S. Boyd, "Growing well-connected graphs," in *Proc. IEEE CDC'06*, San Diego, CA, Dec. 2006, pp. 6605–6611.

[14] B. Mohar, "Eigenvalues, diameter, and mean distance in graphs," *Graphics and Combinatorics*, vol. 7, no. 1, pp. 53–64, Mar. 1991.

[15] D. Mosk-Aoyama, "Maximum algebraic connectivity augmentation is NP-hard," *Oper. Res. Lett.*, vol. 36, no. 6, pp. 677–679, 2008.

[16] S. Kirkland and M. Neumann, "Algebraic connectivity of weighted trees under perturbation," *Linear and Multilinear Algebra*, vol. 42, no. 3, pp. 187–203, 1996.

[17] J. J. Molitierno and M. Neumann, "The algebraic connectivity of two trees connected by an edge of infinite weight," *The Electronic Journal of Linear Algebra*, vol. 8, no. 3, pp. 1–13, 2001.

[18] I. S. Dhillon and B. N. Parlett, "Orthogonal eigenvectors and relative gaps," *SIAM J. Matrix Anal. Appl.*, vol. 25, no. 3, pp. 858–899, 2003.