Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

On the Structure of Graph Edge Designs that Optimize the Algebraic Connectivity

Yan Wan, Sandip Roy, Xu Wang, Ali Saberi, Tao Yang, Mengran Xue, Babak Malek

Abstract— **We take a structural approach to the problem of designing the edge weights in an undirected graph subject to an upper bound on their total, so as to maximize the algebraic connectivity. Specifically, we first characterize the eigenvector(s) associated with the algebraic connectivity at the optimum, using optimization machinery together with eigenvalue sensitivity notions. Using these characterizations, we fully address optimal design in tree graphs that is quadratic in the number of vertices, and also obtain a suite of results concerning the topological and eigen-structure of optimal designs for bipartite and general graphs.**

I. Introduction

The burgeoning importance of large-scale dynamical networks in our day-to-day lives has brought about a keen interest in graph theory in the engineering community. While the interface between graph theory and dynamical-network analysis has been widely studied, the problem of *designing* networks and their controllers to exploit a their topological structure remains challenging. It is our perspective that such network and controller designs are of critical importance, and so we have engaged in a major effort to develop a systematic graph-based methodology for design [1], [2], [3], [5]. Here, we enrich our methodology to address the particular problem of designing the edge weights in a graph, so as to optimize an associated network's dynamics.

Edge-weight design in graphs is needed for a range of network applications, including sensor network algorithm-design, assembly of autonomous vehicle teams, virus-spreading control, and optimization of numerical sampling tools, among others [1], [3], [7]. Of particular interest, Boyd and his co-workers have identified and given a common framework for several important edge-weight design problems (see [7] and references contained theirein), and in turn have used a semi-definite programming (SDP) methodology to find optimal edge-weight designs. While the SDP methodology does provide solutions to the edge-weight design problems given some (non-trivial) regularity conditions, it does not directly yield insight into graphical and dynamical properties of high-performance designs; such insights are critical both because they help to characterize the behaviors of the optimally designed systems, and because they allow us to identify/construct good designs even when precise optimization is not possible. Here, we use a methodology that meshes optimization, spectral graph theory, and eigenvalue sensitivity notions to obtain structural results concerning optimal edge designs. Our approach enriches and informs the existing numerical (SDP-based) characterizations of the optimal edge design.

All authors are with the Department of Electrical Engineering, Washington State University. Correspondence to: {ywan}@eecs.wsu.edu. This work was partially supported by NSF Grants ECS-0528882 and ECS-07258889 and NASA Grant NNA06CN26A.

The methodology for edge-weight design introduced here is closely connected to the techniques for static controller design that we introduced in [1], [2], [3]. In [1], we motivated and addressed an optimal node-design or scaling problem, in particular addressing the design of diagonal matrix K to optimize a performance measure defined from the matrix KG. In [2], [3], we went one step further and showed that the approach can be used to solve some optimal decentralized design problems with constraints, i.e., the design of a diagonal matrix D or K to minimize the dominant eigenvalue of $D + KG$ subject to constraints on D/K. That problem is common in the decentralized control of infrastructure network dynamics such as epidemic spread or air traffic flow. We have also given a complementary methodology for designing *dynamic* decentralized controllers using a multiple-derivative and multiple-delay paradigm, in [5], [6].

Our efforts here also contribute to algebraic graph theory research. While the bulk of the literature in this domain is focused on analyzing particular graphs rather than designing them, Fiedler has addressed an optimal edge-weight design problem. Fiedler's important work provides structural understanding of an optimal edge-weight design (in particular, for an eigenvalue design of tree-graph Laplacian matrices) [8]. Also of interest, a few recent efforts have obtained structural results for other graph design problems (e.g., the fastest mixing Markov chain problem), but for limited classes of graphs such as paths (see the references in [7]). Meanwhile, other efforts in the algebraic graph theory community have sought structural insight into optimal edge-weight designs, starting from the SDP formulation (e.g. [10]). From this algebraic graph-theory perspective, our efforts serve to 1) further the structural analysis given in the literature, 2) achieve design for a much broader class of edge-weight design problems and graphs, and 3) clarify that structural insights into the optimal design in fact yield and/or permit refinement of good algorithms for edge-weight design.

In this article, we study one canonical graph-edge design problem: namely, that of selecting the edge weights subject to an upper bound on their total, so as to maximize the graph's *algebraic connectivity* (i.e., the second-smallest eigenvalue of the Laplacian matrix associated with the graph). To save space, we omit all proofs and examples; the reader is kindly asked to see the extended document [9] on our web page for these aspects.

The structural approach to graph-edge design studied here

978-1-4244-3124-3/08/$25.00 ©2008 IEEE

can be adapted to numerous other design problems of interest. We are in the process of applying the graph-edge design methodology for two other canonical problems—namely, 1) optimization of the dominant eigenvalue of (both symmetric and asymmetric) positive matrices defined on a graph, and 2) optimization of the mixing rate of a Markov chain (i.e., optimization of the subdominant eigenvalue of a stochastic matrix defined on a graph). In the interest of space, these results will be presented in future work.

II. Review and Problem Formulation

We consider the problem of designing the edge weights in a graph, to maximize the **algebraic connectivity** or **Fiedler eigenvalue** of the graph, i.e. the second-smallest eigenvalue of the associated Laplacian matrix.

Formally, let us consider a non-negatively weighted, undirected graph G with vertex set $V = \{1, \dots, n\}$, edge set E (where edges are specified as unordered pairs of vertices), and nonnegative weight $k_{ij} = k_{ji}$ associated with each edge $\{i, j\} \in E$. We recall that the **Laplacian matrix** L of the graph G is defined as follows: $L = D - K$, where $k_{ij} \triangleq 0$ for $\{i, j\} \notin E$, $K \triangleq [k_{ij}]$, and D is a diagonal matrix with ith diagonal entry given by $\sum_{j=1}^{n} k_{ij}$. We note that the Laplacian matrix L is a symmetric positive semidefinite matrix, and so has real eigenvalues with the minimum one equal to 0. The second-smallest eigenvalue of the Laplacian matrix, known as the **algebraic connectivity** (or, more informally, the Fiedler eigenvalue in honor of the extensive study of it by M. Fiedler), is of wide interest to both graph theorists and dynamical-network analysts.

Here, we aim to assign the nonnegative edge weights k_{ij} for $\{i, j\} \in E$ so as to maximize the algebraic connectivity λ_2, subject to an upper bound Γ on the total edge weight: $\sum_{\{i,j\} \in E} k_{ij} \leq \Gamma$. We refer to this edge-weight design problem as the **Laplacian edge design problem** and refer to E as the designable edge set. We call a design that achieves the maximum an **optimal edge design**, and use the notation k^*_{ij} for the edge weights in such a design; analogously, we use the notation λ^*_2 for the **optimal algebraic connectivity** (i.e., the algebraic connectivity for the optimal edge design). We refer to eigenvectors associated with the algebraic connectivity at the optimum as **optimized eigenvectors**. In the case where the optimal eigenvalue is non-repeated, we use the notation $\mathbf{x}^*$ for the unique (to within a scale factor) eigenvector. When the optimal eigenvalue is repeated m times, we denote a basis for the corresponding eigenspace as $\mathbf{x}^*(1), \dots, \mathbf{x}^*(m)$, and use the notation $\mathbf{x}^*$ for vectors in the span of $\mathbf{x}^*(1), \dots, \mathbf{x}^*(m)$ We also find it convenient to refer to the set of edges in a design that have strictly positive weights as the **non-zero edge set**, and to use the notation E_d (respectively, E^*_d for an optimal design) for this set. We notice that $E^*_d \in E$.

The Laplacian edge design problem posed above was first studied by Fiedler in [8], wherein he coined the term **absolute algebraic connectivity**[1] for the optimal λ_2. In particular, the article [8] gives a structural characterization of the eigenvector of the Laplacian L associated with the optimal algebraic connectivity when it is nonrepeated, and in turn finds the optimal edge design for tree graphs (including ones with repeated algebraic connectivity) in terms of the *variance* of the graph. More recently, Boyd and his co-workers have used semi-definite programming methods to obtain the absolute algebraic connectivity for general graphs, in the process obtaining a set of optimal edge weights. Also of interest, the article [8] exposes that the absolute algebraic connectivity can be found as a solution to an embedding problem, and in turn relates absolute algebraic connectivity to the graph separators.

The structural approach that we pursue here is very closely connected with the eigenvector structure-based approach of Fiedler. However, our focus here is not only on characterizing the absolute algebraic connectivity but also explicitly constructing and characterizing the optimal edge design (or designs). To this end, we clarify that a polynomial-time (in fact, $\mathcal{O}(n^2)$) algorithm can be used to find the optimal edge design for trees, as an alternative to the design strategy given in Fiedler. We also characterize the topological structure of the optimal design in this case and show how the design can be obtained with $\mathcal{O}(n)$ effort upon addition of a new vertex. Further, we obtain several characterizations of the family of optimal edge designs, as well as bounds on the absolute algebraic connectivity, for bipartite and general graphs.

III. The Structure of the Optimized Eigenvector

Let us review and extend the structural characterization of the optimized eigenvectors given in [8]; Specifically, we first review the result from Fiedler's work on the optimized eigenvector, in the case where the optimal algebraic connectivity is assumed non-repeated.

Theorem 1: Consider the Laplacian edge design problem. For an optimal edge design such that the algebraic connectivity is a non-repeated, the following condition holds: for each $\{i, j\} \in E$, either 1) $k^*_{ij} = 0$ or 2) $|x^*_i - x^*_j| = u$ and $k^*_{ij} > 0$, where u is a positive constant.

In words, the theorem states that the absolute difference in eigenvector components along each edge in E used in the optimal design is identical, as long as the algebraic connectivity is non-repeated. We have proved the result using the standard Lagrange multiplier machinery together with eigenvalue sensitivity ideas, see [9], as an alternative to the majorization proof in [8]. The proof methodology highlights the analagy of our graph-design methodology with our previous results on optimal scaling and static decentralized controller design [1], [2], [3].

The above result fully characterizes the optimized eigenvector, in the case where the optimal algebraic connectivity is a nonrepeated eigenvalue of the Laplacian matrix. We note that this optimized eigenvector has numerous other structural

[1]To be precise, Fiedler uses the term for the case where $\Sigma_{i,j \in E} k_{ij} \leq n$, so we use different terminology here; however, notice that the more general problem trivially reduces to this one through scaling.

properties that are common to all eigenvectors associated with the algebraic connectivity of a Laplacian, see e.g. [11] for details. We also note (as was also noted in [8]) that the above structural result implies that the non-zero edge set of the optimum forms a *bipartite graph*, if indeed the algebraic connectivity is not repeated. We will show in the following subsections that the above structural result facilitates design and provides a variety of insights into the optimal design.

As our later discussion will clarify, the optimal solution commonly has repeated algebraic connectivity, and so characterizations of the optimized eigenvectors associated with a repeated algebraic connectivity are needed. Broadly, the repeated-eigenvalue case is complicated because of the difficulty in finding sensitivities of repeated eigenvalues to perturbations, see e.g. [12]. Here, we review the explicit characterization of the eigenvectors given by Fiedler for the case that the designable edge set forms a tree [8]. We then develop a check for whether a repeated-eigenvalue solution is an optimal one, for general graphs.

Let us begin with the tree case.

Theorem 2: Consider the optimal edge design problem for a connected tree graph (i.e., for the case that the designable edge set E specifies a connected tree). The unique optimal edge design satisfies one of the following three conditions:

1) the optimal algebraic connectivity is non-repeated, the optimized eigenvector $\mathbf{x}^*$ has no zero components, $|x_i^* - x_j^*|$ is identical for each $\{i,j\} \in E$, and the optimized eigenvector's components are strictly increasing along the path from any vertex s to any vertex t in the designable edge graph such that $x_s^* < 0$ and $x_t^* > 0$.

2) the optimal algebraic connectivity is non-repeated, the optimized eigenvector $\mathbf{x}^*$ has a single zero component at a vertex of degree 2 in the designable edge graph, $|x_i^* - x_j^*|$ is identical for each $\{i,j\} \in E$, and the optimized eigenvector's components are strictly increasing along the path from any vertex s to any vertex t in the designable edge graph such that $x_s^* \leq 0$ and $x_t^* > 0$.

3) the optimal algebraic connectivity is repeated z times. Also, all (eigen)vectors $\mathbf{x}^* \in Span(\mathbf{x}^*(1), \cdots, \mathbf{x}^*(z))$ have a zero component at a particular vertex i with degree $z+1$. Further, for a path from vertex i to any vertex t, the sequences $x_i^*, \cdots, x_t^*$ is monotonic and the differences of the eigenvector components across each edge in the path are identical.

Next, let us give a general condition for checking whether a repeated-eigenvalue solution is optimal.

Theorem 3: Consider an edge weight assignment with $\sum_{\{i,j\}\in E} k_{ij} = \Gamma$, which has algebraic connectivity repeated z times, and coresponding eigenvectors $\mathbf{x}^*(1), \cdots, \mathbf{x}^*(z)$. This assignment is an optimal edge design, if and only if the following condition holds: for any set of numbers Δ_{ij}, $\{i,j\} \in E$ such that $\sum_{\{i,j\}\in E} \Delta_{ij} = 0$, there exists $\mathbf{x}^* \neq 0 \in Span(\mathbf{x}^*(1), \cdots, \mathbf{x}^*(z))$ such that $\sum_{\{i,j\}\in E} \Delta_{ij}(x_i^* - x_j^*)^2 \leq 0$.

We note that Theorem 3 gives a full structural characterization of the optimal design, albeit in an implicit form. Also, Theorem 3 reduces to Theorem 1 when $z = 1$, i.e. the algebraic connectivity is non-repeated.

IV. An Explicit Design for Tree Graphs

The structural characterizations of the optimized eigenvector(s) from Section 3 immediately permit us to develop finite-dimensional search algorithms for finding the optimal edge design. For tree graphs, it turns out that we can obtain the optimal design exactly with $\mathcal{O}(n^2)$ operations. In this section, we give the algorithm for finding the optimal edge design for tree graphs, present some simple qualitative insights into the pattern of the optimal edge weights, and show how the design can be updated upon addition of a new vertex in $\mathcal{O}(n)$ time.

We note that the article [8] already has obtained the optimal edge designs for tree graphs, with the results phrased in terms of the *variance* of the graph. Our algorithm for the optimum is an alternative to the one of Fiedler, that facilitates distributed computation of the design and leads to the further presented results.

Before presenting the algorithm, we define some terminology. First, consider an edge $\{i,j\}$ in the tree. We refer to the two partitions formed upon removal of the edge as the **partitions induced by edge** $\{i,j\}$, and use the notation $S_i(\{i,j\})$ and $S_j(\{i,j\})$ for the partition including i and the partition including j, respectively. Similarly, we refer to the partitions formed upon removal of a vertex i in the tree as the **vertex-partitions induced by the vertex** i. We use the notation $S_1(i), \ldots, S_m(i)$ for the (in general m) partitions formed. Finally, we use the notation $D(i,j)$ for the distance (number of edges) between vertex i and vertex j.

We are now ready to present the algorithm. The algorithm has two steps. The first concerns finding the **critical edge** or **critical vertex**, i.e. the edge such that the optimized eigenvector has components of different signs at the two ends, or else the vertex for which the eigenvector component is null. The second step is the computation of the optimal design.

Algorithm, Step 1: Finding the Critical Edge or Vertex. Search through the edges in the graph, until a critical edge or vertex is found. In particular, for each edge $\{i,j\} \in E$, find $C(\{i,j\}) = \frac{1}{n}(\sum_{k\in S_j(\{i,j\})} D(i,k) - \sum_{k\in S_i(\{i,j\})} D(i,k))$. If $0 < C(\{i,j\}) < 1$, then $\{i,j\}$ is the critical edge. If none of the edges is critical, find $C(\{i\}) = \frac{1}{n}\sum_{k=1,\cdots,n} D^2(i,k)$. Then the vertex i for which $C(\{i\})$ is minimized is the critical one (and in fact $C(\{i\}) \leq C(\{j\})$-1 for all j in this case). We notice that finding the distances and hence $C(\{i,j\})$ or $C(\{i\})$ requires $\mathcal{O}(n)$ additions/multiplications, and so Step 1 has maximum computational complexity $\mathcal{O}(n^2)$.

Algorithm, Step 2: Finding the Optimal Edge Design. Let us consider two cases, namely the case where we have a critical edge and that where we have a critical vertex.

Critical-Edge Case: Say that the edge $\{i,j\}$ is the critical one. Then construct the n-component vector $\widehat{\mathbf{x}}$ that has kth entry given by $D(i,k) - C(\{i,j\})$. Then $\mathbf{x}^* = \frac{\widehat{\mathbf{x}}}{||\widehat{\mathbf{x}}||_2}$ is the optimized eigenvector (normalized to unit length). Also, the optimal algebraic connectivity is $\lambda^* = \Gamma(x_i^* - x_j^*)^2$.

Finally, the optimal edge weights can be found recursively, as follows. For edges $\{a, b\}$ such that a is a leaf, $k^*_{a,b} = \frac{\lambda^* x^*_a}{x^*_a - x^*_b}$; these optimal edge weights can immediately be calculated. Once they have been calculated, notice that there exists at least one non-leaf edge $\{a, b\}$ whose optimal weights has not been computed, and for which all the optimal edge weights in the partition $S_a(\{a, b\})$ have been computed. For this edge, $k^*_{a,b}$ can be computed as $k^*_{a,b} = \frac{\lambda^* x^*_a - \sum_{q \in \mathcal{N}(a)} k^*_{q,a}(x^*_q - x^*_a)}{x^*_a - x^*_b}$, where $\mathcal{N}(a)$ contains the neighbors of vertex a except for b. After computing this optimal edge weight, we see that again one of the edges whose optimal weight remains to be computed has an associated partition for which all optimal weights have been computed; hence, recursively, all the optimal edge weights can be computed. It is easy to check that this computation is $\mathcal{O}(n)$.

Critical Vertex Case: Say that the vertex i is the critical one. Construct a vector $\mathbf{y}$ with kth entry given by $D(i, k)$. For each edge $\{a, b\}$ in the graph, find the **scaled weight** $\widehat{k}_{a,b}$ of the edge. Do this as follows: first, for each edge $\{a, b\}$ such that a is a leaf of the tree, find $\widehat{k}_{a,b} = \frac{y_a}{y_a - y_b}$. After the weights for the leaves have been found, notice that in each vertex-partition induced by i, at least one of the edges whose weight remains to be found has associated (edge) partition which 1) is within the vertex partition and 2) has all scaled weights determined. For any such edge $\{a, b\}$ (with a connected to the partition with known weights), the scaled weight is computed as $\widehat{k}_{a,b} = \frac{y_a - \sum_{q \in \mathcal{N}(a)} \widehat{k}_{q,a}(y_q - y_a)}{y_a - y_b}$. In this way, scaled weights can recursively be computed for all edges. Next, the optimal edge weight for each edge $\{a, b\}$ can be computed as $k^*_{a,b} = \frac{\Gamma \widehat{k}_{a,b}}{T}$, where $T = \sum_{\{a,b\} \in E} \widehat{k}_{a,b}$. Also, the optimal eigenvalue is given by $\lambda^* = \frac{\Gamma}{T}$. Finally, the optimized eigenvectors form a vector space of dimension $m - 1$, where m is the number of vertex-partitions induced by i. In particular, any vector with kth entry given by $c_j y_k$, where j is the induced partition which contains vertex k, that has zero sum is an optimized eigenvector. It is easy to check that this computation is $\mathcal{O}(n)$. $\square$

We note that the computation of the critical edge or vertex is identical to the one given in [8]. Meanwhile, the edge-weight computation that we use contrasts from that in [8], in that individual weights are found recursively from neighbors' weights. We notice that this approach make clear that only a limited set of global information (e.g. the critical edge location) need be transmitted to permit local computation of the optimal weights.

The algorithm is computationally $\mathcal{O}(n^2)$. We notice that when we add new vertices to an existing tree, we do not need to recalculate the critical edge or vertex from scratch. Specifically, when a single node is added to an existing tree, Step 1 can be simplified to have only constant computational time. This simplification is made possible by the fact that upon tree expansion, the critical edge/vertex $\{i, j\}$ moves in a special pattern. We show this result in Theorem 4. For convenience, let us denote the tree constructed from tree $T = T(V, E)$ by connecting new vertex q to vertex $p \in V$ as $\tilde{T} = T(\tilde{V}, \tilde{E})$, where $\tilde{V} = \{V, q\}$ and $\tilde{E} = \{V, \{p, q\}\}$.

Theorem 4: Consider a tree graph $T = T(V, E)$ and say that a vertex q is added to the graph through connecting to vertex $p \in V$. The critical edge or vertex of tree $\tilde{T}$ can be determined as follows:

1) Suppose tree T has a critical edge $\{i, j\}$, and $q \in S_i(\{i, j\})$, then tree $\tilde{T}$ either has a critical edge $\{i, j\}$ or has a critical vertex i.

2) Suppose tree T has a critical vertex i, and $q \in S_k(i)$, then tree $\tilde{T}$ either has a critical vertex i or has a critical edge $\{i, j\}$, where $j \in S_k(i)$ and $\{i, j\} \in E$.

The theorem informs that when adding one node to an existing tree, the critical vertex/edge of the expanded tree can be easily found. The critical vertex/edge in the expanded tree either stays the same, or moves along the edge in the direction of the added node without crossing the nearest vertex. The precise location of the critical vertex/edge can be obtained in constant time through modifying $C(\{i, j\})$ of tree T.

Also, the edge weights in the optimal solution have some interesting dependences on the visual graph structure. We present these simple results in Theorems 5 and 6.

Theorem 5: Consider any path from the critical edge/vertex to a leaf in a connected tree graph. The optimal edge weights along the path decrease monotonically.

Theorem 6: Consider the set of edges that connect to a leaf in a connected tree graph. The magnitudes of optimal edge weights in the set are ordered according to, and in fact are linear proportional to, the corresponding leaves' distances to the critical edge/vertex.

For a connected tree graph, the above two theorems give us necessary conditions for an edge weight design to be the optimum. In the circumstance that the optimal edge weights are hard to obtain, we can resort to the theorems to obtain a suboptimal but good solutions.

V. Some Structural Results for Bipartite and General Graphs

From the structural results in Section 3, we also can obtain a finite-search algorithm for finding the optimal design, in the case where the optimal algebraic connectivity is not repeated. Briefly, the finite-dimensional algorithm works as follows: for each possible cutset of the graph, it turns out that one can assign a unique *potential optimal eigenvector* such that the cutset separates eigenvector components with different signs. In turn, an edge weight assignment that achieves this eigenvector can be identified, if one exists, and the optimality of the potential solution can be checked. A direct implementation of such a finite-search algorithm (which is deeply related to our algorithms for scaling design, see [1], [2], [3]) is computationally intensive as compared to the standard numerical methods, and so we omit the details.

A more important consequence of our direct approach to the Laplacian edge design problem is that it yields significant structural insight into the optimal design for general (non-tree) graphs. Here, we summarize some interesting insights into the optimal designs for more general classes of graphs.

In presenting results for non-tree designable edge sets, we shall often find it convenient to distinguish between bipartite and non-bipartite graphs. Let us thus recall that a **bipartite graph** is one in which the vertices can be divided into two sets, such that every edge connects a vertex in one set with a vertex in the other. We shall also find it convenient at times to classify graphs in terms of whether they admit an optimal design without a repeated algebraic connectivity.

A. Edge-Utilization Structure and Eigenstructure

Let us begin with some analysis of the edge-utilization structure and eigen-structure for the optimal. A critical observation that underlies the results in this subsection is that the Laplacian edge design problem (almost) always admits multiple optimal edge designs. Let us formalize this concept, in the following theorem:

Theorem 7: Consider a Laplacian edge-design problem that has at least one optimal design with non-repeated algebraic connectivity. The optimal edge design for the problem is unique if and only if the designable edge set forms a tree graph.

This result can be proved straightforwardly by using the eigenvalue/eigenvector equation at the optimum together with perturbation arguments. Thus, we see that for non-tree designable edge sets, the Laplacian edge-design problem admits a family of optimal edge-weight selections. The method of proof shows that, in fact, multiple edge designs exist even though the optimizing eigenvector may be unique.

Observing that many optimal edge designs are possible, we may be motivated to seek for designs that have particular edge-utilization characteristics. The next result clarifies that, at least for bipartite graphs, optimal designs can be obtained that are identical in structure to the designable edge graph. Designs with this characteristic may be preferable e.g. because of their desirable fault-tolerance properties.

Theorem 8: Consider a Laplacian edge-design problem with designable edge set that forms a bipartite graph. If there is at least one optimal design for which the algebraic connectivity is non-repeated, then there is an optimal design for which all the designable edges are assigned non-zero weights.

One can also obtain a more restricted result on edge utilization for general graphs:

Theorem 9: Consider a Laplacian edge-design problem. If there is at least one optimal design for which the algebraic connectivity is non-repeated, then there is an optimal design that 1) is bipartite and 2) would become non-bipartite if any other edge from the designable edge set E were made non-zero.

Also, the locations of the eigenvalues in the optimal solution—and in particular the possibility for repeated algebraic connectivity—is important because it informs on e.g. the sign patterns of eigenvector components (and hence associated dynamics) and impacts use of numerical tools for design.

In the following theorem, we characterize the presence of solutions to the Laplacian edge design problem with repeated eigenvalues:

Theorem 10: Each Laplacian edge design problem either has an optimal edge design such that the non-zero edge weight set E_d forms a tree, or has an optimal edge design such that the algebraic connectivity is repeated.

This theorem, which also is proved by studying the family of possible designs achieving the optimum, clarifies the very common presence of repeated-eigenvalue optima. This common occurrence of repeated-eigenvalue solutions is important from the perspective of using numerical optimization tools (such as SDP-based methods, see e.g. [7]): at feasible solutions with repeated eigenvalues, the derivative of the Lagrangian with respect to the design parameters becomes ill-defined; the fact that the optima themselves have such a structure suggests that appropriate regularization may be needed in using the numerical techniques.

In concluding the discussion on the edge-utilization structure and eigenstructure of optimal designs, we recall that the optimizing eigenvector has a very special structure, in the case where the algebraic connectivity is nonrepeated, see also [8]. In particular, as we stated in positing the finite-search algorithm, we can uniquely determine the optimizing eigenvector (obviously, to within a scaling factor) once the cutset that separates positive-valued and negative-valued components in the eigenvector is known. This conclusion is a natural generalization of the eigenvector construction for trees (see Theorem 2), but does not provide low-order algorithms for finding the optimum, and so we omit the details. What is interesting is that this insight into eigenvector structure (or, alternatively, the check for optimality given in [8]) immediately yields a conclusion about whether or not an the optimal edge design can be a tree:

Theorem 11: Consider a Laplacian edge-design problem, where the minimum edge- and vertex- cutsets of the designable edge graph are both at least 2. Then the non-zero edge set for any optimal edge design does not form a tree.

We notice that this result gives a more precise characterization of the edge-utilization structure, in particular one where the structures of *all* optimal designs are characterized for a broad class of designable edge graphs.

B. Graph-Theoretic Bounds on Performance

Our solution methodology for the Laplacian edge-design problem also permits development of graph-theoretic bounds on the optimal algebraic connectivity. Such bounds lend insight into the design methodology, because they permit comparison of the optimal design with uniform-weight designs and allow evaluation of performance for particular graph classes (e.g., regular meshes, small-world networks, etc). Here, as an example, we introduce a lower-bound on the optimal algebraic connectivity, and discuss application of this bound.

Lower bounding the algebraic connectivity is often of particular interest, because such a bound implies an upper bound on the *settling time* of an associated dynamics, e.g., a distributed agreement protocol in a sensor network or a *formation* dynamics in an autonomous-vehicle team (e.g., [13],

[14]). Here, we are interested in lower-bounding the algebraic connectivity, in the case where the optimal edge design is used. We can simply bound the algebraic connectivity in terms of the diameter of graph specified by the designable edge-set, as follows:

Theorem 12: Consider a Laplacian edge design problem that has resource bound Γ. Then the optimal Fiedler eigenvalue is lower-bounded by $\frac{4\Gamma}{nD^2}$, where D is the diameter of the designable edge graph (i.e. of the unweighted graph in n- vertices with edge set E).

The proof of this lower-bound follows from considering a modified Laplacian edge design problem for a spanning tree of the original designable graph, and using the eigenvector-structure result given in Theorem 2. A particularly interesting instance of the above theorem is in the case where the upper bound Γ is set equal to the number of vertices n; this instance permits us to study the scaling of the optimal eigenvalue with respect to the number of vertices in the designable graph, when the total available resource (Γ) is proportional to the number of vertices. In particular, we find that optimal Fiedler eigenvalue is lower-bounded by $\frac{4}{D^2}$ in this case. Thus, we see that the optimal eigenvalue remains relatively large as the number of vertices in the graph increases, as long as the diameter remains small. This result immediately implies that the optimal edge design for small-world graphs, which have small diameter (see [15], [16], [17]), achieve fast settling.

It is worth noting that an even tighter lower bound, phrased in terms of the variance of spanning trees of the graph, follows immediately from [8]. However, application of the variance-based result is more complex, because of the difficulty both in its computation and in choosing the appropriate spanning tree.

C. Meshing our Design with Numerical Methods

Our direct methodology for addressing the Laplacian edge-design problem can also inform existing numerical solution techniques. To review, Boyd and co-workers have used *semi-definite programming* (SDP) techniques to solve the Laplacian edge-design problem and other similar design problems (e.g., [7]). In our complementary work on network scaling problems [1], we have also noted that simple gradient descent-type algorithms can be used to obtain optima. These numerical methods typically yield fast solutions to the design problems, which complement the structural insights that can be obtained through our direct approach. Here, let us briefly discuss one way in which our approach can inform, and be meshed with, the numerical solution strategies.

In particular, we stress again that our methodology shows that many optimal designs can be obtained for non-tree graphs, not only the single optimum provided by the numerical methods. Thus, our direct methodology can naturally be interfaced with the numerical techniques to identify a family of optimal edge designs, as follows:

1) The numerical technique can be used to obtain with high fidelity one optimal edge design and corresponding optimal algebraic connectivity and optimized eigenvector. We notice that our direct analysis explicitly specifies the optimized eigenvector once the cutset separating vertices with positive- and negative- valued components is identified; thus, the exact optimized eigenvector can be obtained after a finite number of iterations of the numerical algorithm.

2) Once the optimized eigenvector $\mathbf{x}^*$ and eigenvalue λ^* have been obtained, the eigenvector equation can be used to find the space of edge designs for which $\mathbf{x}^*$ and λ^* are an eigenvector and eigenvalue, respectively. We notice that only some of these designs have λ^* as the algebraic connectivity (rather than as one of the higher eigenvalues); it is straightforward to check whether λ^* is the optimal eigenvalue from the eigen-analysis.

We have developed an example illustrating how a family of valid designs can be found, as well as a larger example for a flow-network-design application. We ask the reader to see the extended paper [9] for these examples.

REFERENCES

[1] S. Roy and A. Saberi, "Scaling: a canonical design problem for networks," *International Journal of Control*, vol. 80, no. 8, pp. 1342-1353, 2007.

[2] Y. Wan, S. Roy, and A. Saberi, "Designing spatially-heterogeneous strategies for control of virus spread," to appear in *IET Systems Biology*.

[3] Y. Wan, S. Roy, and A. Saberi, "A new focus in the science of networks: towards methods for design," *Proceedings of the Royal Society A*, vol. 464, pp. 513-535, March 2008.

[4] S. Roy, J. Krueger, D. Rector, and Y. Wan, "A network model for activity-dependent sleep regulation," submitted to *Journal of Theoretical Biology*.

[5] Y. Wan, S. Roy, A. Saberi, and A. Stoorvogel, "A multiple-derivative and multiple-delay paradigm for decentralized controller design: uniform-rank systems," Submitted to *Automatica*, 2007.

[6] Y. Wan, S. Roy, A. Saberi, and A. Stoorvogel, "A multiple-derivative and multiple-delay paradigm for decentralized controller design: introduction using the canonical double-integrator network," Submitted to *Automatica*, 2007.

[7] S. Boyd, "Convex optimization of graph laplacian eigenvalues," *Proceedings International Congress of Mathematicians*, vol. 3, pp. 1311-1319, 2006.

[8] M. Fiedler, "Absolute algebraic connectivity of trees," *Linear and Multilinear Algebra*, no. 26, pp. 85-106, 1990.

[9] Y. Wan, S. Roy, X. Wang, A. Saberi, T. Yang, M. Xue, and B. Malek, "On the Structure of graph edge designs that optimize the algebraic connectivity," full version available at www.eecs.wsu.edu/~ywan.

[10] F. Göring, C. Helmberg, and M. Wappler, "Embedded in the Shadow of the Separator," submitted to *SIAM Journal on Optimization*, September 2, 2005.

[11] F. R. K. Chung, *Spectral Graph Theory*, American Mathematical Society Press: Providence, RI, 1997.

[12] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Clarendon Press: Oxford, 1965.

[13] S. Roy, A. Saberi, and K. Herlugson, "Formation and alignment of distributed sensing agents with double-integrator dynamics," IEEE Press Monograph on *Sensor Network Operations*, April 2006.

[14] W. Ren, R. W. Beard, and E. M. Atkins, "Information consensus in multivehicle cooperative control," *IEEE Control Systems Magazine*, vol. 27, no. 2, pp. 71-82, 2007.

[15] D. Watts and S. Strogatz, "Collective dynamics of small-world networks," *Nature*, vol. 393, pp. 440-442, 1998.

[16] J. Kleinberg, "The small-world phenomenon: an algorithmic perspective," in *Proceedings of the 32nd ACM Symposium on Theory of Computing*, Portland, OR, May 2000.

[17] B. Bollobas and F. R. K. Chung, "The diameter of a cycle plus a random matching," *SIAM Journal on Discrete Mathematics*, vol. 1, pp. 328-333, 1988.