Algebraic Connectivity Maximization of Air Transportation Network On Flight Routes Addition/Deletion Problem

P. Wei[a,*], L. Chen[b], D. Sun[a]

[a] *School of Aeronautics and Astronautics, Purdue University, West Lafayette, Indiana, 47907, U.S.A.*
[b] *Department of Industrial Engineering, University of Louisville, Louisville, Kentucky, 40292, U.S.A.*

Abstract

A common metric to measure the robustness of a network is its algebraic connectivity. This paper introduces the flight routes addition/deletion problem and compares three different methods to analyze and optimize the algebraic connectivity of air transportation networks. The Modified Greedy Perturbation Algorithm (MGP) provides a local optimum in an efficient iterative manner. The Weighted Tabu Search (WTS) is designed for the flight routes addition/deletion problem to offer a better optimal solution with longer computation time. The relaxed semidefinite programming (SDP) is used to set a performance upper bound and three corresponding rounding techniques are applied to obtain the feasible solution. The simulation results present the trade-off among the Modified Greedy Perturbation, Weighted Tabu Search and relaxed SDP, with which we can decide the appropriate algorithm to adopt for maximizing the algebraic connectivity of an air transportation network under different network sizes. Finally a real air transportation network of Virgin America is analyzed.

Keywords: Air transportation network, Algebraic connectivity, Optimization

1. Introduction

The air transportation network is made of the nodes that represent airports and the edges that represent the flight routes which directly link two airports [29, 17, 28]. In an air transportation network either a node failure or a link failure may happen due to weather

*Corresponding author
Email addresses: weip@purdue.edu (P. Wei), lijian.chen@louisville.edu (L. Chen), dsun@purdue.edu (D. Sun)

hazard and other emergencies. How to build a robust or well connected network, which has the ability to transport passengers between any two airports via one link or through multiple links under the unpredictable node or link failures, is a practical problem that has significant economic impact. In this paper we measure and optimize the robustness of air transportation networks by computing its *algebraic connectivity*, which is one of the network metrics from graph theory research. The algebraic connectivity is defined by Fiedler as the second smallest Laplacian eigenvalue of a graph [12]. Compared to the betweenness, degree and clustering coefficient that are defined on each node [4], the algebraic connectivity is selected as the network robustness metric in this work because the researchers have shown that it has the tightest bound to the network robustness in terms of node and link connectivities and it is the most computational efficient network robustness metric [20, 19, 6].

Traditionally, the *node connectivity* and the *edge connectivity* are two metrics to evaluate a graph's robustness [14]. The node (edge) connectivity of a graph G is the minimum number of node (edge) deletions sufficient to disconnect G.

In order to show the limitation of node (edge) connectivity metric, two different topologies are shown, where Figure 1a is a N-node line topology and Figure 1b is a N-node star topology. The node connectivities for both topology formations are 1 and so are the edge connectivities. However, the star topology should be more robust than the line topology because in Figure 1b the network will be disconnected only when the central node fails, while in Figure 1a any node failure can cause the network disconneted except the two end nodes. The robustness features of the two topologies are intuitively different. But neither the node connectivity nor the edge connectivity can observe the difference between these two topologies.

According to the definition in [12], when $N = 4$, the algebraic connectivities of Figure 1a and Figure 1b are 0.586 and 1 respectively, which show that the star topology is more robust than the line topology.

Researchers from graph theory and networks have proved that the algebraic connectivity provides better resolution on how well a graph or network is connected and is a fair measurement of the network robustness [20, 6].

The air traffic demand is expected to continue its rapid growth in the future. The Federal Aviation Administration (FAA) estimated that the number of passengers is projected to increase by an average of 3% every year until 2025 [11], so the expanding load on the current air transportation network will cause more and more flight delays and cancellations with the limited airspace and sector capacities. As a result, a robust air transportation network design scheme is necessary to sustain the increasing traffic demand. This is the major motivation of this work.

An air transportation network can be represented as a graph G with n nodes and m edges. With the fact that if a direct flight route exists between airport a_i and airport a_j, normally the direct return flight route from a_j to a_i also exists [18], G is constructed as an undirected simple graph, where the airports are indexed as $\{a_i | i = 1, 2, \ldots, n\}$ and the link between airports a_i and a_j is named as e_{ij}.

In reality imposing weights on edges is necessary because the weights bring more information to an air transportation network. An edge weight may be used to represent the link strength for random failure, the number of daily flight operations on this edge, the monthly passenger flow between two airports, etc [23]. In this study we set the edge weight as the link strength with non-negative integers, which is used to describe how likely a link is going to fail. A stronger link under random failure is assigned with a bigger edge weight value while smaller edge weights are assigned to those links that are easy to fail (in practice, some flights are more often to be cancelled or delayed, and some air space is more likely to be affected by severe weather).

The aim of this work is to maximize the algebraic connectivity in a weighted air transportation network under several given constraints. Although the maximized algebraic connectivity value might be abstract, the resulted optimal network design is applicable and beneficial in practice. The methods developed in this work are expected to be implemented to measure and to enhance the robustness of air transportation networks. With these methods the network planners from airline companies can maintain or modify the structure of an existing network and make strategies for the future development of the air transportation network.

The rest of the paper is organized as follows. The related work from literature is pre-

sented in Section 2. In Section 3 we formulate the flight routes addition/deletion problem and show that the weighted problem is NP-hard. In Section 4 the heuristic algorithm from Ghosh and Boyd for unweighted graph is analyzed and extended to solve the weighted problem. The tabu search algorithm is designed for the flight routes addition problem in Section 5. In Section 6 the relaxed semidefinite programming (SDP) method is introduced and three different rounding techniques are discussed. In Section 7 we evaluate the performances of our algorithms via simulations. A real air transportation network of Virgin America is investigated in Section 8. Section 9 concludes this paper.

2. Related Work

Air transportation network and its robustness have been studied over the last several years. Guimera and Amaral [17] first studied the scale-free graphical model of air transportation network. Conway [8] showed that although it was better to describe the national air transportation system or the commercial air carrier transportation network as a complex system, the scale-free network model was satisfactory for network simulation. Bonnefoy [5] showed that the air transportation network was scale-free with aggregating multiple airport nodes into mega nodes. Alexandrov [2] defined that on-demand transportation networks would require robustness in system performance (time of service windows, denial of service rates). The robustness of an on-demand network would depend on tolerance of the network to variability in temporal and spatial dynamics of weather, equipment, facility and crew positioning, etc. Kotegawa et al. [23] surveyed different metrics for air transportation network robustness, including betweenness, degree, centrality, connectivity, etc. He selected clustering coefficient and eigenvector centrality as the network robustness metrics in his machine learning approach. However, according to Bigdeli et al. [4], the betweenness, degree and clustering coefficient are all defined on a single node. In order to evaluate the robustness of the entire network, these metrics need to be calculated on every node, which may result in a non-efficient computation process. Jamakovic and Uhlig [20], Jamakovic and Mieghem [19] found that the algebraic connectivity was a generic metric in the analysis of various robustness problems in several typical network models. Jamakovic and Uhlig [20] studied that the algebraic connectivity and the network robustness in terms

of node and link connectivities on three different complex network models: the random graph of Erdos-Renyi, the small-world graph of Watts-Strogatz and the scale-free graph of Barabasi-Albert. They concluded that the algebraic connectivity can be considered as a fair measure of the robustness in all three complex network models. Jamakovic and Mieghem [19] showed that the larger the value of the algebraic connectivity, the better the graph's robustness to node and link failures. Byrne et al. [6] showed algebraic connectivity can improve the robustness of the network by reducing the characteristic path length. They stated that the algebraic connectivity was the efficient network robustness measure with much less computation time for both small and large size networks. Kim and Mesbahi [21] proposed an iterative algorithm for maximizing the algebraic connectivity with a semidefinite programming solver at each recursive step. Although their algorithm has a local convergence behavior, simulations suggest that it often leads to a global optimum. Vargo et al. [28] introduced the algebraic connectivity to air transportation networks for the first time. They chose the algebraic connectivity as the robustness metric and built the optimization problem solved by the edge swapping based tabu search algorithm.

In this paper, the flight routes addition/deletion problem is formulated based on the weighted air transportation network. Greedy heuristic, tabu search and semidefinite programming techniques are applied to find the maximal algebraic connectivity and the corresponding optimal network design. By comparing these three methods, we hope to provide advice to the network planners from the airline companies on how to select the appropriate method according to the trade-off between algorithm performance and computation time.

3. Problem formulation and its NP-hardness

In real world there are very few chances to create a new air transportation network either in a local region or for a whole country because the air routes coverage in modern days is already wide especially in the United States. Instead, to maintain or to improve the robustness of an existing network, restricted by adding or deleting a few links due to airline budgets, weather conditions, economic policies, etc., is much more imperative and necessary.

3.1. Preliminaries

A graph G is used to represent the air transportation network. The *weighted adjacent matrix* A of graph G has the ith row and jth column entry a_{ij}. The diagonal items are all zeros and the off-diagonal item a_{ij} $(i \neq j)$ is equal to edge weight w_{ij}:

$$a_{ij} = \begin{cases} w_{ij}, & \text{if node } i \text{ and node } j \text{ are connected} \\ & \text{by an edge } e_{ij} \text{ with weight } w_{ij}; \\ 0, & \text{if node } i \text{ and } j \text{ are not connected.} \end{cases} \tag{1}$$

where the weights are usually bounded by an upper limit W because a weighting scheme without an upper bound is normally not applicable in practice.

The *weighted Laplacian matrix* L is defined based on the adjacent matrix A. Each item l_{ij} of L can be written as:

$$l_{ij} = \begin{cases} -a_{ij}, & \text{if } i \neq j; \\ \sum_{k=1}^{n} a_{ik}, & \text{if } i = j. \end{cases} \tag{2}$$

The second smallest eigenvalue λ_2 of L is the *weighted algebraic connectivity*, which is the focus of this paper.

3.2. Problem formulation

$G(V, E_0)$ is the graphical description of an existing integral weighted air transportation network, where the node set V is the collection of all the airports in this network and the edge set E_0 contains the existing links between the airport pairs. The size of set V is n and the size of E_0 is m. The objective is to maximize λ_2 with a fixed number k of edge additions or deletions based on E_0 while the edges to be added or deleted are given in a pre-determined set P (for addition) or Q (for deletion). All the weights of the edges in sets E_0, P, Q are non-negative integral and bounded by W. We denote the routes to be added or deleted as a set of ΔE. Thus the *flight routes addition problem* is:

$$\begin{aligned} & \max \lambda_2(G(V, E_0 + \Delta E)) \\ \text{s.t.} \quad & |\Delta E| = k, \\ & \Delta E \subseteq P, P \cap E_0 = \emptyset, \\ & w_{ij} \in \mathbb{I}, w_{ij} < W. \end{aligned} \tag{3}$$

The *flight routes deletion problem* is:

$$\begin{aligned} & \max \lambda_2(G(V, E_o - \Delta E)) \\ \text{s.t.} \quad & |\Delta E| = k, \\ & \Delta E \subseteq Q, Q \subseteq E_0, \\ & w_{ij} \in \mathbb{I}, w_{ij} < W. \end{aligned}$$

The flight routes addition problem and the flight routes deletion problem are formulated as two disjoint problems for different needs. For simplicity of demonstration, we only study the flight routes addition problem. The algorithms for solving flight routes deletion problem can be developed similarly.

3.3. Alternative problem formulation

Since $\lambda_2(G)$ is computed based on the weighted Laplacian matrix of G, we can also denote $\lambda_2(G)$ as $\lambda_2(L)$, in which L is the weighted Laplacian matrix of graph G. According to Ghosh and Boyd [13], the weighted Laplacian matrix L can be represented by the dot product summation of *edge vectors*. For an edge e connecting two nodes i and j, we define the edge vector $h_e \in \mathbb{R}^n$ as $h_e(i) = 1$, $h_e(j) = -1$, and all other entries 0. w_e is the non-negative integral weight on e. Suppose there are m edges in graph G, the weighted Laplacian matrix L of G is the $n \times n$ matrix:

$$L = \sum_{e=1}^{m} w_e h_e h_e^T. \tag{4}$$

which is equivalent to the weighted Laplacian matrix definition in Eq. (2).

Note that $\lambda_2(L)$ is monotone increasing with the edge set: if $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ are such that $E_1 \subseteq E_2$, then $\lambda_2(G_1) \leq \lambda_2(G_2)$ [12]. That is, the more connected graph (on the same vertex set) has the greater algebraic connectivity.

According to the edge vector description of L in Eq. (4) and omitting the "constraints" on given weights w_{ij}, the flight routes addition problem (3) can be written as:

$$\begin{aligned} & \max \lambda_2(L_0 + \textstyle\sum_{e=1}^{|P|} x_e w_e h_e h_e^T) \\ \text{s.t.} \quad & \mathbf{1}^T x = k, \\ & x \in \{0, 1\}^{|P|}. \end{aligned} \tag{5}$$

where L_0 is the weighted Laplacian matrix of the existing network $G(V, E_0)$. A fixed number of k edges are to be added. P is the pre-determined set with size $|P|$ which contains the candidate edges to be added. e is the index for candidate edges in P. x_e is a boolean variable, in which 1 means that edge e from P is selected into ΔE in (3) and 0 means that e is not selected into ΔE. x is a vector consisting of all x_e's whose length is $|P|$, illustrating which candidate edges are to be added and which are not. We observe that $\lambda_2(L)$ is a function of x, which can be denoted as $\lambda_2(L(x))$.

3.4. NP-hardness

Theorem 1. *The non-negative integral weighted flight routes addition problem is NP-hard.*

Proof 1. *The proof is a process of two steps reduction.*

Step 1: As we defined in the problem, the weights are non-negative integral and bounded by W. Let's set W as 2, now all the weights can only be 0 or 1. The problem becomes a regular unweighted problem.

Step 2: Another reduction happens in set P. Based on the same node set V, we construct a complete graph G_c and denote the edges of the complete graph as set E_c. Now if we reduce P to all the other edges which do not exist in E_0, i.e. the set $(E_c - E_0)$, our problem is transformed to the maximum algebraic connectivity augmentation problem [25], which is proved to be NP-hard.

Since the flight routes addition problem can be reduced into a proved NP-hard problem, it is also NP-hard. □

As a result, we seek heuristic algorithms to solve the flight routes addition problem instead of deriving the closed-form optimal solution.

4. Modified Greedy Perturbation

The second smallest eigenvalue $\lambda_2(L)$ is called the *algebraic connectivity*, and the corresponding normalized eigenvector is called the *Fiedler vector* [12]. Ghosh and Boyd [13] present a greedy local heuristic, they add the k edges one at a time based on the calculation of the Fiedler vector. In this section we extend their heuristic into the Modified Greedy Perturbation Algorithm (MGP) for the weighted problem.

4.1. The bond between λ_2 and L

According to Mohar [24], no matter L is weighted or not, the algebraic connectivity can be computed by:

$$\lambda_2(L(x)) = \min\{\frac{y^T L(x) y}{y^T y} \,|\, y \neq 0, \mathbf{1}^T y = 0\}, \tag{6}$$

where y is a $n \times 1$ non-zero vector and it is orthogonal with all-one vector $\mathbf{1}$.

Furthermore, Eq. (6) can be transformed into:

$$\lambda_2(L(x)) = \min\{\frac{y^T L(x) y}{\|y\|^2} \,|\, y \neq 0, \mathbf{1}^T y = 0\}, \tag{7}$$

in which we substitute vector y with normalized vector $v = y/\|y\|$ and we have Eq. (8):

$$\lambda_2(L(x)) = \min\{v^T L(x) v \,|\, \|v\| = 1, \mathbf{1}^T v = 0\}, \tag{8}$$

When the normalized vector v in Eq. (8) is also a Fiedler vector, since

$$\lambda_2(L(x))v = L(x)v, \tag{9}$$

we multiply v^T to the left of the both sides of Eq. (9):

$$v^T \lambda_2(L(x))v = v^T L(x) v. \tag{10}$$

Because vector v is normalized,

$$v^T \lambda_2(L(x))v = \lambda_2(L(x))(v^T v) = \lambda_2(L(x)) = v^T L(x) v. \tag{11}$$

Therefore if v is a Fiedler vector, the minimum in Eq. (8) can be achieved.

$$\lambda_2(L(x)) = v^T L(x) v, \tag{12}$$

Eq. (12) shows that the Fiedler vector v is the bond between the algebraic connectivity λ_2 and the Laplacian matrix L, both in unweighted and weighted cases.

4.2. Maximize $\lambda_2(L(x))$ in the weighted problem

Based on Formulation (5), the weighted Laplacian matrix after flight routes addition is:

$$L(x) = L_0 + \sum_{l=1}^{|P|} x_e w_e h_e h_e^T \tag{13}$$

The partial derivative of $\lambda_2(L)$ with respect to x_e gives the first order approximation of the increase for $\lambda_2(L)$, if the edge e is added to graph G. According to Eq. (12),

$$\frac{\partial}{\partial x_e}\lambda_2(L(x)) = v^T \frac{\partial L(x)}{\partial x_e} v. \tag{14}$$

Plug Eq. (13) into Eq. (14). Since the Laplacian L_0 before flight routes addition is not a function of x_e, we obtain:

$$\begin{aligned} & \frac{\partial}{\partial x_e}\lambda_2(L(x)) \\ = \; & v^T \frac{\partial L(x)}{\partial x_e} v \\ = \; & v^T \frac{\partial (L_0 + \sum_{l=1}^{|P|} x_e w_e h_e h_e^T)}{\partial x_e} v \\ = \; & v^T (w_e h_e h_e^T) v = w_e (v^T h_e)(h_e^T v) \\ = \; & w_e (v_i - v_j)^2. \end{aligned} \tag{15}$$

Thus the unweighted approach is extended into the Modified Greedy Perturbation Algorithm, which picks one edge from remaining candidates with maximal $w_e(v_i - v_j)^2$ at each iteration, where v_i and v_j are the ith and jth items of the Fiedler vector v of the current Laplacian L. The complete algorithm is listed in Algorithm 1.

Algorithm 1 Modified Greedy Perturbation

1: given graph $G(V, E_0)$, candidate edge set P and all the edge weights in E_0 and P
2: let $E = E_0$
3: **for** 1 to k **do**
4: calculate $\lambda_2(G(V, E))$ and its Fiedler vector v
5: $e_{ij} = \arg\max_{e_{ij} \in P} w_{ij}(v_i - v_j)^2$
6: $E = E + e_{ij}$
7: $P = P - e_{ij}$
8: **end for**
9: output $G(V, E)$

The computation complexity of Algorithm 1 heavily depends how fast we can compute the second smallest eigenvalue of the Laplacian matrix L. Line 4 takes polynomial $O(n^\omega)$

arithmetic operations if the square matrix multiplication algorithm [9] is applied, where $\omega = 2.376$. Line 5 takes $|P|$ operations. Line 6 and Line 7 both only need 1 operation. Thus the total complexity is $k \cdot O(n^\omega + |P| + 2) = O(kn^\omega + k|P|)$. Because the researchers believe that $2 < \omega < 2.376$, surveyed by Cohn et al. [7], Demmel et al. [10] and $|P|$ is usually smaller than n^2, we have total complexity $O(kn^\omega)$, which is polynomial.

5. Weighted Tabu Search

Besides a greedy search like the MGP, which usually results in a local optimum, a global optimum search can perform better in solving the maximum. Given the graph $G(V, E_0)$, we are interested in finding k edges from set P to add to G, which together maximally improve $\lambda_2(G(V, E_0))$ to $\lambda_2(G(V, E_0 + \Delta E))$. The global exhaustive search gives out $\binom{|P|}{k}$ different λ_2 and the biggest λ_2 is the final solution. Generally the number $\binom{|P|}{k}$ is so large that the computation time of the exhaustive search is extremely long.

5.1. Tabu search introduction

As an alternative of the exhaustive search, *tabu search* [15, 16] improves the efficiency of the search process by keeping track of the searching trajectory and operating flexible evaluation criteria. An appropriate implementation of memory is the key feature of tabu search. While most searching algorithms keep in memory essentially the best solution value visited so far, tabu search additionally records the searching trajectories to the recent found solutions. The recorded searching trajectories are designed to prevent the reversal or repetitive moves by defining some forbidden moves (tabu). The idea of the tabu search is to permit the method to go beyond local optimum while still running into a better solution value at each step. The tabu restriction can not be violated except when the searching meets *aspiration criteria* [15]. Tabus are sometimes too powerful and they may prohibit attractive moves, even when there is no danger of cycling. It is thus necessary to have aspiration criteria that will allow one to revoke tabus.

5.2. The first attempt in air transportation network connectivity with tabu search

To the best of our knowledge, the research in Kincaid et al. [22], Vargo et al. [28] is the first work that introduces algebraic connectivity into air transportation networks. They

implemented tabu search to solve their optimization problem, in which they link algebraic connectivity with the network synchronization. The synchronization is the dynamical behavior of a network that determines the transmission delays between each pair of nodes [3]. Their analysis of algebraic connectivity is performed on the unweighted Laplacian matrix.

Although they also pursue the maximum of $\lambda_2(G)$, their problem is to design an air transportation network which requires preserving the related graph's degree distribution and keeping the graph connected. Since the problem is different, in this section we design our own tabu search algorithm as the Weighted Tabu Search (WTS) to solve the flight routes addition problem.

5.3. Weighted Tabu Search for Flight Routes Addition Problem

Now we elaborate the details of the WTS implemented in this work. The search is an iterative process and the solution s' of the next iteration is generated from the neighbor of the current solution s supervised by a dynamically updating tabu list T.

5.3.1. Neighbor

The WTS looks for the next iteration solution s' inside the neighbor $N(s)$ of the current solution s. After s' is chosen, the following iteration solution will be selected from its neighbor $N(s')$.

Instead of the swapping operation in Vargo et al. [28], we define $N(s)$ for our problem. An initial solution s contains k edges to be added. The pth ($1 \leq p \leq k$) edge e_{ij} in solution s connects two nodes v_i and v_j, which is shown in Figure 2. All the edges incident to v_i or v_j and in set P constitute the sub-neighbor $N(s,p)$ of solution s at the pth edge. The edges which already exist in G or are not in candidate set P are not displayed in Figure 2. To prevent $N(s,p)$ from being empty, a random jump inside P is also included in $N(s,p)$, which jumps to any edge in P but not current edges 1 to k in solution s. If the random jump gives out an existing edge in $N(s,p)$, then we execute another random jump. The neighbor of current solution s is $N(s)$, which is the union of the sub-neighbors of every edge in s:

$$N(s) = \bigcup_{p=1}^{k} N(s,p).$$

Notice that the k new edges will be selected from their own sub-neighbor $N(s,p)$ and form s' together.

5.3.2. Tabu list

The tabu list T records the most recent $|T|$ moves. For each edge p ($1 \leq p \leq k$) in the current solution s, all the candidate movings from edge p to its neighbor $N(s,p)$ are checked with the tabu list. If a candidate moving repeats one of the moves in T, this candidate will not be selected.

5.3.3. Aspiration criteria

The simplest and most commonly used aspiration criterion allows a tabu move when it results in a solution with an objective value better than that of the current best-known solution. When a searching move in the WTS method finds the solution with a better λ_2 value than the best observed value, this move will be performed. Therefore the best observed value λ_2^* needs to be recorded throughout the searching process.

The tabus will not be violated oftenly because during the tabu search process, it is quite rare that the better λ_2's are found by reverse or repetitive moves.

5.3.4. The complete Weighted Tabu Search algorithm

The complete Weighted Tabu Search (WTS) is shown in Algorithm 2. Line 2 sets an initial solution s_0. Line 3 initializes the parameters, where s is the solution in the current iteration, λ_2^* and s^* record the best λ_2 and its corresponding s respectively. Line 4 shows that the algorithm terminates after Φ iterations. Line 5 to 7 constructs the sub-neighbors of the current solution. Line 9 forms s' from $N(s)$. Line 10 to 13 checks the aspiration criteria. Line 14 to 16 checks whether the move from s to s' is in the tabu list T.

6. Relaxed Semidefinite Programming

In this section we study relaxed semidefinite programming (SDP) formulation of the flight routes addition problem (5). The relaxed SDP solution provides an upper bound for our heuristic algorithms. Furthermore, the relaxed SDP solution can also be used to generate feasible solution by three different rounding techniques, which will be compared in terms of performance and computation time.

Algorithm 2 Weighted Tabu Search

given $G(V, E_0)$, P and the edge weights in E_0
random pick k edges from P to construct s_0
$s = s_0$, $\lambda_2^* = 0$, $s^* = s_0$, T is set to a empty queue with the pre-fixed size $|T|$
for *iteration* $= 1$ to Φ **do**
  for $p = 1$ to k **do**
    construct $N(s, p)$ of the pth edge in s
  end for
  while 1 **do**
    pick one edge p' from each $N(s, p)$ to construct s'
    if $\lambda_2(s') > \lambda_2^*$ **then**
      $s = s'$, update T
      $\lambda_2^* = \lambda_2(s)$, $s^* = s$
    end if
    if s' is not in T **then**
      $s = s'$, update T
    end if
  end while
end for
output λ_2^* and s^*

6.1. Relaxed SDP Formulation

Relaxing the non-linear binary programming (5) by changing the boolean constraint to the linear constraint, we obtain the following relaxation:

$$\begin{aligned} & \max \lambda_2(L_0 + \textstyle\sum_{e=1}^{|P|} x_e w_e h_e h_e^T) \\ \text{s.t.} \quad & \mathbf{1}^T x = k, \\ & \mathbf{0} \leq x \leq \mathbf{1}. \end{aligned} \tag{16}$$

where x is a vector of length $|P|$. With a relaxed domain, the optimal solution of this linear relaxation gives an upper bound for the solution in (5).

The linear relaxation in (16) can be converted into a semidefinite programming format by letting $e_0 = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} e_i$, where e_i's are vectors of the standard basis. Then (16) is formulated as:

$$\begin{aligned} & \max \theta \\ \text{s.t.} \quad & L_0 + \textstyle\sum_{e=1}^{|P|} x_e w_e h_e h_e^T \succeq \theta(I_n - e_0 e_0') \\ & \mathbf{1}^T x = k, \\ & \mathbf{0} \leq x \leq \mathbf{1}. \end{aligned} \tag{17}$$

The relaxed SDP formulation (17) is equivalent to (16), which is explicitly shown by Nagarajan et al. [26]. In this paper SeDuMi [27] is used to solve (17), and the solution serves as an upper bound for the heuristic algorithms. The rounding techniques are then used to create feasible solution based on the relaxed optimal solution.

6.2. Rounding Techniques

Suppose we have found the relaxed optimal solution x^*. We want to select k edges from x^* to form our rounded solution $\hat{x}$ in which $\hat{x}_i = 1$ for k values and $\hat{x}_i = 0$ for the other $|P| - k$. Here we present the methods that have been studied and implemented in this paper.

Greedy: We choose the k biggest elements from the relaxed optimal solution x^*.

Random: We first normalize x^* and treat it as the probability distribution function. Then k elements are randomly selected according to the distribution.

Step by step: We select the biggest element from x^* and update the Laplacian by adding this edge in the SDP formulation. Then we solve the SDP again and repeat these two steps for k times. When k is big, this rounding method needs to solve SDP for k times, which can take a long time. In that case, the "Log step by step" technique is adopted. At each step, we choose the best half of the remaining elements. Thus there are only $log(k)$ SDPs that have to be solved.

6.3. Numerical results

We generate a connected scale-free network with 20 nodes (the simulation setting details can be found at the beginning of next section). Now k edges are going to be added onto the generated network. The results are presented in Figure 3. Figure 3a is λ_2 as a function of k, while Figure 3b is the computation time t with k varying. The upper bound obtained by the SDP relaxation is plotted as well as the curves of three rounding techniques.

Each of the rounding methods presented has some advantages and drawbacks. The one that gives the best performance is the step by step method. However, since it needs to solve the relaxed SDP for k times, it is the slowest. At the same time, the greedy rounding is fast and it provides a performance close to step by step method. We take both the greedy and step by step rounding techniques in next section to compare with the MGP and WTS.

7. Simulation

We use simulations to compare the performances and computation times of the MGP, WTS and relaxed SDP with greedy and step by step rounding methods for the flight routes addition problem. By default $n = 20$ nodes are generated randomly as a scale-free network in a 2D plane. The 20 nodes network is applied in our experiments because the relaxed SDP with step by step rounding takes extremely long computation time when $n > 20$ and we want to compare all the methods together in this section. The generated network is denoted as $G(V, E_0)$. The existing edges in E_0 and the edges in candidate edge set P are assigned with weights $\{w_{ij}|w_{ij} = 1, 2 \text{ or } 3\}$. The number of edges k to be added is set as input. The total iterations Φ in WTS are set to be 1000.

7.1. Single link addition experiment

An unweighted network with $n = 4$ is studied to show that the best link to be added is determined by the existing network topology. Then a weighted network of the same size is analyzed to show that the best link to be added also depends on the weighting scheme.

There are only two kinds of minimum spanning tree topologies for four nodes. We call them topology α and topology β, which are illustrated in Figure 4a and 4b. They are used as the existing network topologies in our experiment.

The candidate links to be added to the existing topology α are e_{13}, e_{13} and e_{24}. The candidate links for the existing topology β are e_{23}, e_{24} and e_{34}. We denote the algebraic connectivity of the existing topology is λ_2^0, the increased algebraic connectivity after adding one link is λ_2' and $\Delta\lambda_2 = \lambda_2' - \lambda_2^0$. Table 1 and 2 list that when a single link is added to the existing topology, how much the λ_2 increases. The λ_2^0 for the unweighted topology α is 0.5858 and the λ_2^0 for the unweighted topology β is 1.

Table 1 and 2 show that the link additions and the algebraic connectivity increase are not trivially related [20]. The best link to be added depends on the existing network topology. Since topology β is already a robust topology, a single link addition does not increase the algebraic connectivity. On the other hand, topology α is vulnerable. So any of the three candiate links brings more robustness, especially when e_{14} turns the whole line topology into a circle topology.

To perform the analysis on the weighted networks, link weights 1, 2 and 3 are assigned to the existing networks. In topology α, we assign $w_{12} = 1$, $w_{23} = 2$ and $w_{34} = 3$. Thus the algebraic connectivity for the existing network is 0.9358. In topology β, we assign $w_{12} = 1$, $w_{13} = 2$ and $w_{14} = 3$. The algebraic connectivity for the existing network with topology β is 1.1944. Table 3 and 4 demonstrate how much a weighted link addition can increase algebraic connectivity.

Table 3 and 4 demonstrate that the link to be added is also related to the weighting scheme. The algebraic connectivity is monotone respect to the link weights. Furthermore, Table 4 shows that the link weights introduce more structural information to the existed network topology β. In the unweighted network of topology β, the effects of adding e_{23}, e_{24} and e_{34} used to be the same. However, to add e_{34} to the weighted network is apparently a

worse option than the other two links.

7.2. The impact of k

k is the number of routes added to graph $G(V, E_0)$ by four different approaches. This simulation in Figure 5 is performed based on the default settings with k varying. In Figure 5a we observe that when more routes are added to the network G, the algebraic connectivity increases monotonically. The WTS offers the best λ_2 enhancement. And the MGP and relaxed SDP with step by step rounding have the runner-up performance. When we are looking for a better performance instead of shorter computation time, the WTS should be considered. The MGP is the fastest method and it gives almost the same performance as the second best step by step rounding method. Therefore, when we prefer speed more than performance, the MGP should be choosen.

In detail, Table 5, 6 and 7 are listed to observe the performance to computation time trade-off among the algorithms when $k = 4$, 8 and 10. The trade-off function values of $\eta\lambda_2 + (1 - \eta)(-t)$ are given in these tables to help the network planner analyze the difference among algorithms and decide which method to use. The factor η is selected as $0.1, 0.3, 0.5, 0.7$ or 0.9 from the range $[0, 1]$. Note when η is small ($\eta = 0.1$ to 0.7), the trade-off function value of MGP is always larger than the values of the other thee methods, which means that the network planner should choose the MGP to obtain a shorter computation time. When η is large ($\eta = 0.9$), the WTS and SDP with step by step rounding should be chosen to achieve a higher λ_2.

In summary, the trade-off is that when the number k increases, the WTS always finds a better solution than the others do. However, the MGP can find a satisfactory solution within such a short time. The long computation time of step by step rounding is unacceptable when the network size is huge and the greedy rounding always has the worst performance. According to the simulations, if the problem is about a small network with fewer airports, the WTS should be selected for maximizing the network robustness; if it is a large network, the MGP should be adopted to provide an efficient computation speed with an acceptable robustness enhancement.

The following two sets of simulations both focus on the parameter settings of the WTS.

7.3. The impact of tabu list size $|T|$

Only WTS is studied in this simulation. We maintain the default network size $n = 20$ and set $k = 10$. During the simulation we hold the same initial positions of the k routes to be added. x-axis is used to represent different tabu list lengths $|T|$ as $5, 10, 15, 20, 25, 30$. Figure 6a shows that with the longer $|T|$, the WTS finds the better λ_2. The reason is that the longer tabu list contains more information which helps the algorithm search more other neighbors and get out of the local optimum. However, a longer tabu list leads to slower computation, which is shown in Figure 6b.

7.4. The impact of different initial positions

In this simulation we study the initial positions of WTS. The initial positions are the initial k routes that are randomly selected from candidate set P and added onto the existing network $G(V, E_0)$. The parameter settings are $n = 20$, $k = 10$, $|T| = 40$. Figure 7 illustrates that the WTS is not very sensitive to the initial selected k routes. The x-axis is the index of different initial positions. Six different initial positions of k added routes produce slightly different λ_2 and their convergence times also vary little.

8. Case Study

In this section, a real air transportation network of Virgin America is studied. The first experiment shows that the weighted algebraic connectivity is a fair measurement for the weighted network robustness under the current Virgin America network topology. The second simulation with real flight info provides the advice that the top 5 and top 10 routes should be added to the current Virgin America network to improve its robustness.

8.1. The current air transportation network of Virgin America

According to the current route map of Virgin America in Figure 8, we consider the 16 airports in the United States and obtain the adjacency matrix as Table 8. The 16 US airports Boston, NYC/JFK, Philadelphia, DC/IAD, DC/DCA, Chicago/ORD, Orlando, Fort Lauderdale, Dallas Fort Worth, Seattle, Portland, San Francisco, Los Angeles, Las Vegas, San Diego and Palm Springs are indexed as numbers 1 to 16. The San Francisco International Airport (SFO) and the Los Angeles International Airport (LAX) are two

major hubs of the entire network. They have at least one direct flight to almost all the other airports.

8.2. Weighted algebraic connectivity and weighted network robustness

In order to study how well the weighted algebraic connectivity can measure the robustness of a weighted air transportation network, we create five different weighted air transportation networks with the same topology in Table 8 by randomly assigning one of the three types of weights to each route. The three types of link weights are mapped to different link failure probability (Table 9).

For each one of the five weighted networks, 1000 trials are performed. The number of the network failures is counted in 1000 random trials. The results are shown in Table 10 with λ_2 sorted in ascending order.

We can see that with a higher weighted algebraic connectivity, the network is more robust and has fewer failure cases. With a lower weighted algebraic connectivity, the network is easier to break down. In summary, the weighted algebraic connectivity is a fair robustness metric for the weighted air transportation network.

8.3. The top 5 and top 10 routes to be added to the Virgin America network

The current air transportation network is weighted based on the historical flight information obtained from September 15, 2012 to November 15, 2012. If there is only one flight between an origin-destination pair (O-D pair), the link weight of this O-D pair will be assigned based on Table 11. The percentage of canceled flights are from 0% to 5% in the historical data. We assign weight 3 to the links with cancellation rate in $[0, 2\%)$, weight 2 to the links with cancellation rate in $[2\%, 4\%)$ and weight 1 to the links with cancellation rate $[4\%, \infty)$. If there are more than one flight on a O-D pair, the probability that all the flights are canceled will be calculated by the joint probability of all the flights on this route and the corresponding link weight will be assigned based on this joint probability by looking up Table 11.

The cancellation rates of the candidate routes to be added can be estimated from the historical route information data published by the FAA. Then the link weights of the candidate routes are obtained through these estimated cancellation rates. Here we assume

that all the candidate routes to be added have the medium link strength with weight 2 and we perform the MGP and WTS methods to find the top 5 and top 10 routes to added to the current Virgin America network. The results have been shown in Table 12 and Table 13.

9. Conclusion

We formulate the flight routes addition/deletion problem to study the air transportation network robustness. Three methods are presented to maximize the algebraic connectivity of the weighted air transportation network in the flight routes addition problem. The Modified Greedy Perturbation Algorithm is a greedy heuristic derived to compute the maximal λ_2. The Weighted Tabu Search is developed to find the global optimum with longer computation time. Furthermore, the relaxed SDP with three rounding methods is adopted to solve the same problem, providing the performance upper bound as well as the feasible solutions.

The contributions of the paper to the literature are (a) formulating the flight routes addition/deletion problem, with which the network planners can maintain or modify the existing network and make strategies for the future development of the air transportation network; (b) comparing the MGP, WTS and the relaxed SDP with two different rounding methods to maximize the algebraic connectivity in a simulated scale-free network and a real air transportation network. The suggestions are provided to the network planners on how to select the appropriate algorithm.

Our future work is to evaluate the performances of the three methods and analyze their trade-offs in a large-scale real air transportation network. The top k flight routes to be added to the large-scale network will be determined to maximally increase the network robustness.

References

[1] , December 2012. Virgin America Flight Route Map. `http://www.virginamerica.com/travel/flight-routes.html`.

[2] Alexandrov, N., 2004. Transportation network topologies. Tech. rep., NASA Langley Research Center.

[3] Atay, F., Biyikoglu, T., Jost, J., 2006. Synchronization of networks with prescribed degree distributions. IEEE Transactions on Circuits and Systems I 53, 92–98.

[4] Bigdeli, A., Tizghadam, A., Leon-Garcia, A., 2009. Comparison of network criticality, algebraic connectivity, and other graph metrics. In: Proceedings of the 1st Annual Workshop on Simplifying Complex Network for Practitioners. No. 4.

[5] Bonnefoy, P. A., June 2008. Scalability of the air transportation system and development of multi-airport systems: A worldwide perspective. Ph.D. thesis, Massachusetts Institute of Technology.

[6] Byrne, R. H., Feddema, J. T., Abdallah, C. T., July 2009. Algebraic connectivity and graph robustness. Tech. rep., Sandia National Laboratories, Albuquerque, New Mexico 87185.

[7] Cohn, H., Kleinberg, R., Szegedy, B., Umans, C., 2005. Group-theoretic algorithms for matrix multiplication. In: 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05). pp. 379–388.

[8] Conway, S., 2004. Scale-free networks and commercial air carrier transportation in the united states. In: 24th International Congress of the Aeronautical Sciences. Yokohama, Japan.

[9] Coppersmith, D., Winograd, S., 1990. Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation 9 (3), 251–280.

[10] Demmel, J., Dumitriu, I., Holtz, O., Oct. 2007. Fast linear algebra is stable. Numer. Math. 108 (1), 59–91.

[11] FAA, December 2010. FAA Aerospace Forecasts FY 2008-2025. Federal Aviation Administration.

[12] Fiedler, M., 1973. Algebraic connectivity of graphs. Czechoslovak Mathematics Journal 23, 298–305.

[13] Ghosh, A., Boyd, S., December 2006. Growing well-connected graphs. In: Proceedings of the 45th IEEE Conference on Decision and Control. pp. 6605–6611.

[14] Gibbons, A., July 1985. Algorithmic Graph Theory. Cambridge University Press.

[15] Glover, F., 1989. Tabu search-part i. ORSA Journal on Computing 1, 190–206.

[16] Glover, F., 1990. Tabu search-part ii. ORSA Journal on Computing 2, 4–32.

[17] Guimera, R., Amaral, L., 2004. Modeling the world-wide airport network. European Physical Journal B 38, 381–385.

[18] ICAO, 2010. Procedures for Air Navigation Services - Rules of the air and air traffic services. International Civil Aviation Organization, doc 4444-RAC/501.

[19] Jamakovic, A., Mieghem, P. V., 2008. On the robustness of complex networks by using the algebraic connectivity. In: et al, A. D. (Ed.), Networking 2008 Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet. pp. 183–194.

[20] Jamakovic, A., Uhlig, S., May 2007. On the relationship between the algebraic connectivity and graph's robustness to node and link failures. In: 3rd EuroNGI Conference on Next Generation Internet Networks.

[21] Kim, Y., Mesbahi, M., January 2006. On maximizing the second smallest eigenvalue of a state-dependent graph laplacian. IEEE TRANSACTIONS ON AUTOMATIC CONTROL 51 (1).

[22] Kincaid, R., Alexandrov, N., Holroyd, M., 2008. An investigation of synchrony in transport networks. Complexity 14 (4), 34–43.

[23] Kotegawa, T., DeLaurentis, D., Noonan, K., Post, J., 2011. Impact of commercial airline network evolution on the u.s. air transportation system. In: Ninth USA/Europe Air Traffic Management Research and Development Seminar (ATM2011). Berlin, Germany.

[24] Mohar, B., 1991. The Laplacian spectrum of graphs. Graph Theory, Combinatorics, and Applications 2, 871–898.

[25] Mosk-Aoyama, D., 2008. Maximum algebraic connectivity augmentation is NP-hard. Operations Research Letters 36 (6), 677–679.

[26] Nagarajan, H., Rathinam, S., Darbha, S., Rajagopal, K., 2011. Algorithms for synthesizing mechanical systems with maximal natural frequencies. Non-linear analysis - Real world applications.

[27] Sturm, J., 1999. Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. Optimization methods and software 11 (1), 625653.

[28] Vargo, E., Kincaid, R., Alexandrov, N., 2010. Towards optimal transport networks. Systemics, Cybernetics and Informatics 8 (4), 59–64.

[29] Wei, P., Sun, D., Aug 2011. Weighted algebraic connectivity: An application to air transportation network. In: the 18th IFAC World Congress. Milan, Italy.

Table 1: Single link addition analysis for the 4-node unweighted network with topology α.

link	λ_2'	$\Delta\lambda_2/\lambda_2^0(\%)$
e_{13}	1	70.71%
e_{14}	2	241.41%
e_{24}	1	70.71%

Table 2: Single link addition analysis for the 4-node unweighted network with topology β.

link	λ_2'	$\Delta\lambda_2/\lambda_2^0(\%)$
e_{23}	1	0%
e_{24}	1	0%
e_{34}	1	0%

Table 3: Single link addition analysis for the 4-node weighted network with topology α.

link	λ_2'	$\Delta\lambda_2/\lambda_2^0(\%)$
e_{13} with $w_{13}=1$	2	113.72%
e_{13} with $w_{13}=2$	2.5359	170.99%
e_{13} with $w_{13}=3$	2.7376	192.54%
e_{14} with $w_{14}=1$	2.4746	164.44%
e_{14} with $w_{14}=2$	3.1716	238.92%
e_{14} with $w_{14}=3$	3.2313	245.30%
e_{24} with $w_{24}=1$	1.1078	18.38%
e_{24} with $w_{24}=2$	1.1716	25.20%
e_{24} with $w_{24}=3$	1.2038	28.64%

Table 4: Single link addition analysis for the 4-node weighted network with topology β.

link	λ_2'	$\Delta\lambda_2/\lambda_2^0(\%)$
e_{23} with $w_{23}=1$	2	67.45%
e_{23} with $w_{23}=2$	2.0905	75.03%
e_{23} with $w_{23}=3$	2.1155	77.12%
e_{24} with $w_{24}=1$	1.8105	51.58%
e_{24} with $w_{24}=2$	1.9088	59.81%
e_{24} with $w_{24}=3$	1.9407	62.48%
e_{34} with $w_{34}=1$	1.2014	0.59%
e_{34} with $w_{34}=2$	1.2030	0.72%
e_{34} with $w_{34}=3$	1.2038	0.79%

Table 5: Trade-off analysis between performance and computation time for $k = 4$.

Methods	$\eta = 0.1$	$\eta = 0.3$	$\eta = 0.5$	$\eta = 0.7$	$\eta = 0.9$
Greedy	0.1481	0.8459	1.5438	2.2416	2.9395
Step by step	−0.2720	0.4964	1.2648	2.0332	2.8016
MGP	0.3293	0.9905	1.6517	2.3130	2.9742
WTS	−0.0552	0.7234	1.5019	2.2805	3.0590

Table 6: Trade-off analysis between performance and computation time for $k = 8$.

Methods	$\eta = 0.1$	$\eta = 0.3$	$\eta = 0.5$	$\eta = 0.7$	$\eta = 0.9$
Greedy	0.3201	1.2615	2.2028	3.1441	4.0855
Step by step	-0.5838	0.6308	1.8455	3.0601	4.2748
MGP	0.4877	1.4679	2.4480	3.4282	4.4084
WTS	-0.2537	0.9587	2.1711	3.3835	4.5960

Table 7: Trade-off analysis between performance and computation time for $k = 12$.

Methods	$\eta = 0.1$	$\eta = 0.3$	$\eta = 0.5$	$\eta = 0.7$	$\eta = 0.9$
Greedy	0.4673	1.6676	2.8680	4.0683	5.2686
Step by step	−0.9350	0.7639	2.4627	4.1616	5.8605
MGP	0.6385	1.9225	3.2064	4.4903	5.7743
WTS	−0.4738	1.0793	2.6323	4.1854	5.7384

Table 8: The adjacency matrix consists of 16 Virgin America Airlines airports in the US.

Index / Airport	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Boston	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
NYC/JFK	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
Philadelphia	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
DC/IAD	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
DC/DCA	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Chicago/ORD	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
Orlando	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
Fort Lauderdale	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
Dallas Fort Worth	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
Seattle	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
Portland	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
San Francisco	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
Los Angeles	1	1	1	1	0	1	1	1	1	1	1	1	0	0	0	0
Las Vegas	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
San Diego	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Palm Springs	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Table 9: The mapping between link weights and their link failure probabilities.

link weight w_{ij}	1	2	3
link failure probability	5%	3%	1%

Table 10: The network failure probability is related to the weighted algebraic connectivity λ_2.

weighted λ_2	1.0306	1.7586	1.8661	1.9711	2.3128	2.7393
failures in 10000 trials	1113	991	763	571	423	355

Table 11: The mapping between link weights and the percentage of flight cancellation rate.

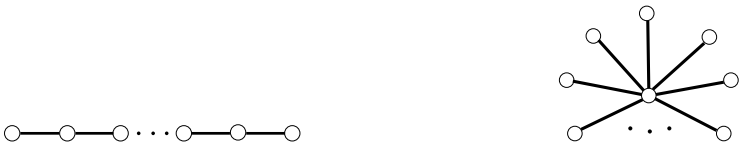
link weight w_{ij}	1	2	3
link failure probability	$[4\%, \infty)$	$[2\%, 4\%)$	$[0, 2\%)$

Table 12: Top 5 routes to be added to the Virgin America route map.

MGP	WTS
Boston-San Diego	Los Angeles-Palm Springs
Philadelphia-DCA	Los Angeles-Las Vegas
DCA-San Diego	NYC/JFK-DCA
DCA-Palm Springs	DCA-Los Angeles
Las Vegas-Palm Springs	Los Angeles-San Diego

Table 13: Top 10 routes to be added to the Virgin America route map.

MGP	WTS
Boston-Las Vegas	Los Angeles-San Diego
Boston-San Diego	DCA-Las Vegas
Philadelphia-DCA	DCA-Los Angeles
IAD-DCA	Philadelphia-DCA
DCA-San Diego	San Diego-Palm Springs
DCA-Palm Springs	Orlando-Palm Springs
Orlando-Palm Springs	Seattle-Las Vegas
Fort Worth-Palm Springs	DCA-Palm Springs
Seattle-San Diego	Fort Worth-Palm Springs
Las Vegas-Palm Springs	Boston-DCA



(a) (b)

Figure 1: N-node line topology and star topology.

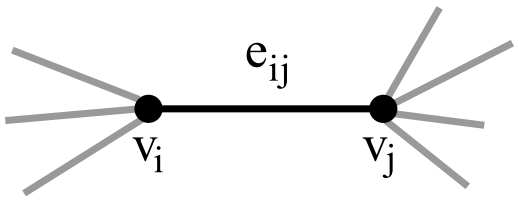


Figure 2: The neighbor of the pth edge e_{ij} in current solution s.

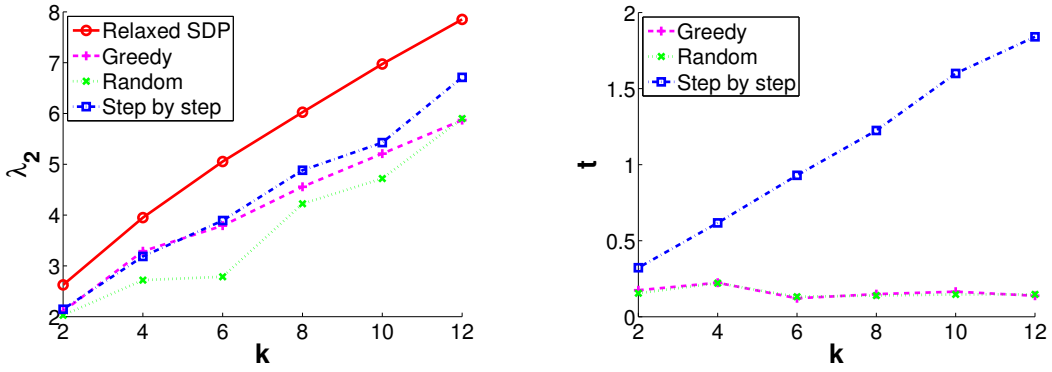


Figure 3: The performance and computation time for three rounding techniques of the relaxed SDP.

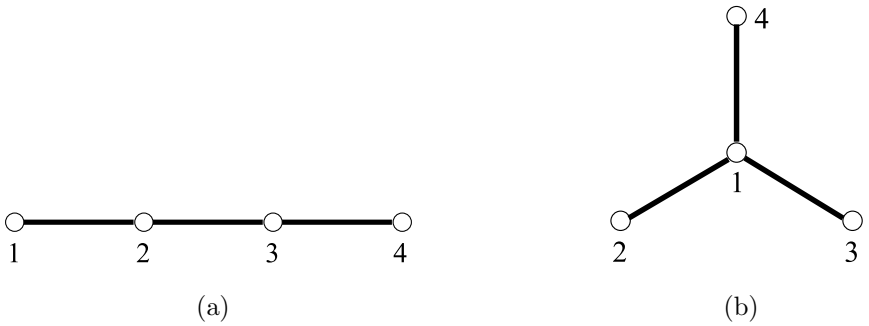
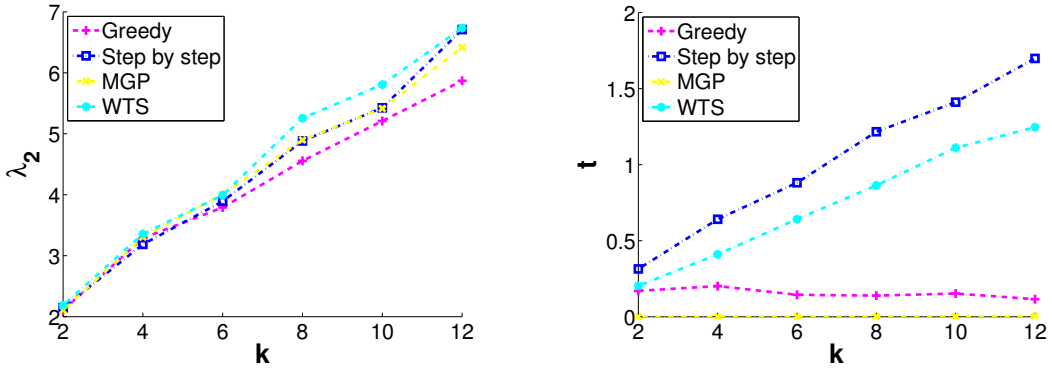
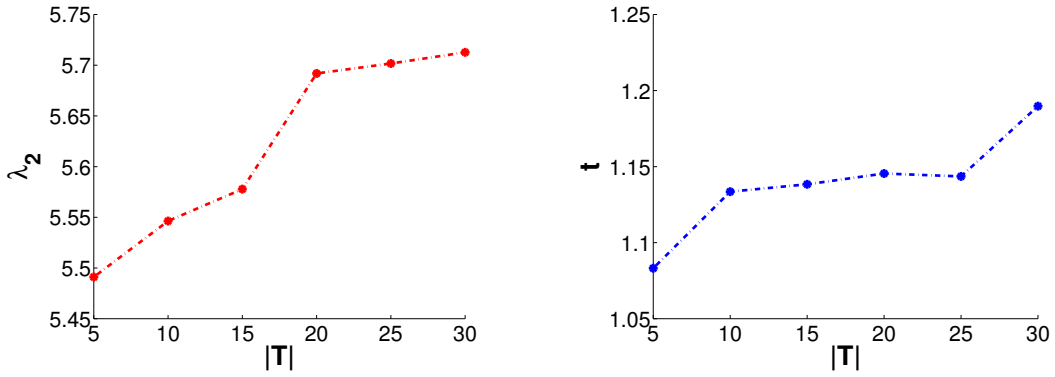


Figure 4: The two minimum spanning tree topologies for four nodes.



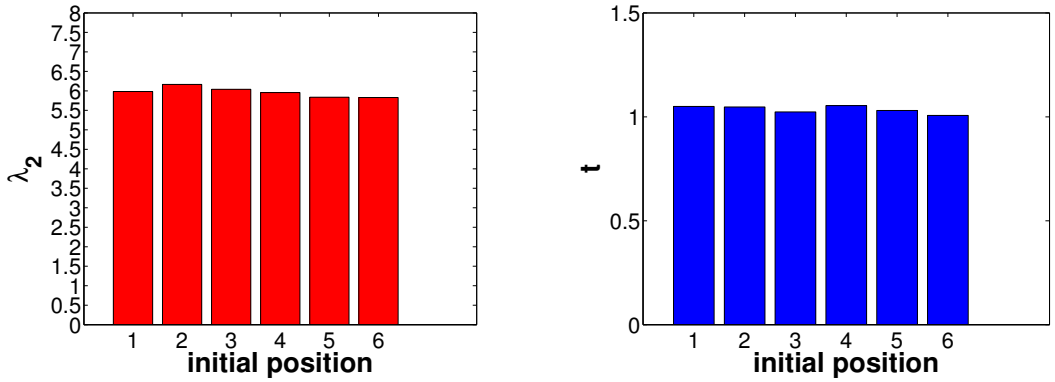
(a) (b)

Figure 5: Impact of k.



(a) (b)

Figure 6: Impact of $|T|$.



(a) (b)

Figure 7: Impact of initial position.

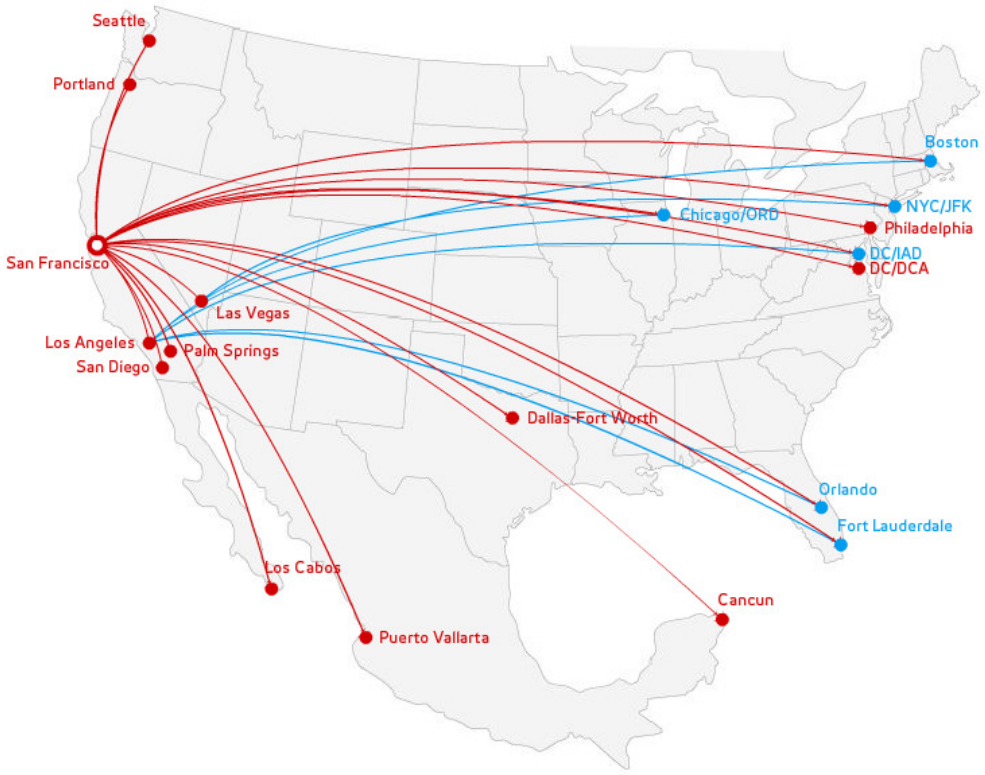


Figure 8: An air transportation network route map for Virgin America Airlines.