Transportation Research Part E 61 (2014) 13–27

Contents lists available at ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

Algebraic connectivity maximization of an air transportation network: The flight routes' addition/deletion problem

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

[a] School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN 47907, USA
[b] Department of Industrial Engineering, University of Louisville, Louisville, KY 40292, USA

ARTICLE INFO

Article history:
Received 5 October 2012
Received in revised form 26 December 2012
Accepted 3 October 2013

Keywords:
Air transportation network
Algebraic connectivity
Optimization

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 the air transportation network. The Modified Greedy Perturbation algorithm (MGP) provides a local optimum in an efficient iterative manner. The Weighted Tabu Search (WTS) is developed 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 then three rounding techniques are applied to obtain feasible solutions. The simulation results show 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 the air transportation networks of different sizes. Finally a real air transportation network of Virgin America is investigated.

1. Introduction

An air transportation network consists of the nodes that represent airports and the edges that represent the flight routes which directly link two airports (Wei and Sun, 2011; Guimera and Amaral, 2004; Vargo et al., 2010). It can be described as a graph G with n nodes and m edges. With the fact that if a direct flight route exists between airport v_i and airport v_j, normally the direct return flight route from v_j to v_i also exists (ICAO, 2010), G is constructed as an undirected graph, where the airports are indexed as $\{v_i | i = 1, 2, \ldots, n\}$ and the edge between airports v_i and v_j is named as e_{ij}.

In an air transportation network either a node failure or an edge failure may happen due to weather disturbance, long Ground Delay Program (GDP), long Airspace Flow Program (AFP), aircraft mechanical problem, upline flight delay/cancel and other unforeseen events. How to build a robust or well connected network, which has the ability to transport passengers between any two airports via one edge or through multiple edges under the unpredictable node or edge failures, is a practical problem that has significant economic impact. In this paper we measure and optimize the robustness of an air transportation network by computing its *algebraic connectivity*, which is one of the network metrics from graph theory research. Compared to the betweenness, degree and clustering coefficient that are defined on each node (Bigdeli et al., 2009), the algebraic connectivity is selected as the network robustness metric in this work because researchers have shown that it has the tightest bound to the network robustness in terms of node and edge connectivities and it is the most computational efficient network robustness metric (Jamakovic and Uhlig, 2007; Jamakovic and Mieghem, 2008; Byrne et al., 2009).

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

1366-5545/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.tre.2013.10.008

Traditionally, the *node connectivity* and the *edge connectivity* are the two metrics to evaluate a graph's robustness (Gibbons, 1985). 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 Fig. 1a is an N-node line topology and Fig. 1b is an 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 Fig. 1b the network will be disconnected only when the central node fails, while in Fig. 1a any node failure can cause the network to disconnect 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.

The algebraic connectivity is defined by Fiedler as the second smallest Laplacian eigenvalue of a graph (Fiedler, 1973). According to this definition, when $N = 4$, the algebraic connectivities of Fig. 1a and b are 0.586 and 1 respectively, which show that the star topology is more robust than the line topology. This example demonstrates that the algebraic connectivity is a finer measurement for network robustness. Researchers from graph theory and network theory have also proved that the algebraic connectivity provides better resolution on how well a graph or network is connected and it is a fair measurement of the network robustness (Jamakovic and Uhlig, 2007; Byrne et al., 2009).

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 (FAA, 2010). The expanding traffic demand on the current air transportation networks of different airlines will cause more and more flight cancelations with the limited airport resources and airspace capacities. As a result, more robust air transportation networks are desired to sustain the increasing traffic demand for each airline and for the entire National Airspace System (NAS). That is the major motivation of this work.

In reality imposing weights on edges is necessary because the weights bring more information to an air transportation network. Usually different routes have different edge (link) strength. For example, the route failure rate between JFK and BOS during summer is higher than that between SFO and LAX because of the crowded northeastern airspace (AFP is more frequent) and more summer thunder storms. Another example is that a shorter route is easier to fail than a longer transcontinental route because: (a) airlines usually put larger aircraft on transcontinental route and these aircraft are more robust to weather disturbance; (b) airlines are more likely to cancel shorter route flights because the flight frequency on a shorter route is higher therefore the passengers on the canceled flight are easier to be reaccommodated to later flights. In summary, the routes have different possibilities to fail and in this study we use different edge weights to describe the varying link strength. A stronger edge under random failure is assigned with a greater edge weight value while smaller edge weights are assigned to those edges that are easier to fail.

The goal of this work is to maximize the algebraic connectivity in a weighted air transportation network under given constraints. Although the maximized algebraic connectivity value is abstract, the optimized air transportation network design is applicable. 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 decision makers from airline companies can maintain or modify the structure of an existing regional or nationwide network and design strategies for the future development of their air transportation network.

The rest of the paper is organized as follows. The related work from literature is presented 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 for the unweighted graph is extended to solve the weighted problem. The tabu search algorithm is developed 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 numerical simulations. A real air transportation network of Virgin America is investigated in Section 8. Section 9 concludes this paper.

2. Related work

The air transportation network and its robustness have been studied over the last several years. Guimera and Amaral (2004) first studied the scale-free graphical model of the air transportation network. Conway (2004) showed that it was better to describe the national air transportation system or the commercial air carrier transportation network as a system-of-systems. Bonnefoy (2008) showed that the air transportation network was scale-free with aggregating multiple airport

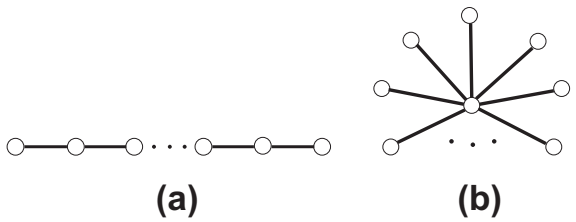

Fig. 1. N-node line topology and star topology.

nodes into mega nodes. Alexandrov (2004) defined that on-demand transportation networks would require robustness in system performance. 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. (2011) surveyed different metrics for air transportation network robustness, including betweenness, degree, centrality, connectivity, etc. They selected clustering coefficient and eigenvector centrality as the network robustness metrics in their machine learning approach. Bigdeli et al. (2009) compared algebraic connectivity, network criticality, average degree, average node betweenness and other metrics. Jamakovic and Uhlig (2007); Jamakovic and Mieghem, 2008 found that the algebraic connectivity was an important metric in the analysis of various robustness problems in several typical network models. Byrne et al. (2009) showed that algebraic connectivity was the efficient measure for the robustness of both small and large size networks. Kim and Mesbahi (2006) 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. (2010) 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 we measure the robustness of air transportation network by computing the algebraic connectivity, which is usually considered as one of the most reasonable and efficient evaluation methods (Byrne et al., 2009; Jamakovic and Uhlig, 2007; Jamakovic and Mieghem, 2008; Wei and Sun, 2011). 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 provide advice to the decision makers from 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 an entirely new air transportation network either in a local region or for a whole country because the flight route coverage in modern world 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 routes (edges) 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 adjacency 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 the edge weight w_{ij}:

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

where the weights w_{ij}'s 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 adjacency 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 the Laplacian matrix 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 edges between 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. 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:

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

The flight routes addition problem and the flight routes deletion problem are formulated as two independent 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 accordingly.

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 (2006), the weighted Laplacian matrix L can be represented by the dot product summation of the *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 equal to 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 an $n \times n$ matrix:

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

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

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

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

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. $\lambda_2(L)$ 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-step reduction.*

Step 1: As we defined in the problem formulation, the weights are non-negative integral and bounded by W. If we set W as 2, then 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 all 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 but are included in E_c, i.e. the set $(E_c - E_0)$, our problem is transformed to the maximum algebraic connectivity augmentation problem (Mosk-Aoyama, 2008), which is proved to be NP-hard.

Since the flight routes addition problem can be reduced to 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* (Fiedler, 1973). Ghosh and Boyd, 2006 presented a greedy local heuristic, they added 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 (1991), no matter L is weighted or not, the algebraic connectivity can be computed by:

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

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

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

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

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

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

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

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

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

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

Because vector u is normalized,

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

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

$$\lambda_2(L(x)) = u^T L(x) u. \tag{12}$$

Eq. (12) shows that the Fiedler vector u 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 edge e is added to graph G. According to Eq. (12),

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

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

$$\frac{\partial}{\partial x_e}\lambda_2(L(x)) = u^T \frac{\partial L(x)}{\partial x_e} u = u^T \frac{\partial(L_0 + \sum_{l=1}^{|P|} x_e w_e h_e h_e^T)}{\partial x_e} u = u^T (w_e h_e h_e^T) u = w_e (u^T h_e)(h_e^T u) = w_e (u_i - u_j)^2. \tag{15}$$

Thus the unweighted approach is extended into the Modified Greedy Perturbation Algorithm, which picks one edge from the remaining candidates with maximal $w_e(u_i - u_j)^2$ at each iteration, where u_i and u_j are the ith and jth items of the Fiedler vector u 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 u
5: $e_{ij} = \arg\max_{e_{ij} \in P} w_{ij}(u_i - u_j)^2$
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 (Coppersmith and Winograd, 1990) is applied, where $\omega = 2.376$. Line 5 takes $|P|$ operations. Line 6 and Line 7 both need only 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. (2005), Demmel et al. (2007) and $|P|$ is usually smaller than n^2, we have the 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 finding 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 checks $\binom{|P|}{k}$ different λ_2's, 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 to the exhaustive search, *tabu search* (Glover, 1989; Glover, 1990) improves the efficiency of the search process by tracking the search trajectory and operating flexible evaluation criteria. An appropriate implementation of memory is the key feature of tabu search. While most search algorithms keep in memory essentially the best solution value visited, tabu search additionally records the search trajectories to the recent found solutions. The recorded search 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 cannot be violated except when the search meets *aspiration criteria* (Glover, 1989). 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.

In this section we develop our own tabu search algorithm as the Weighted Tabu Search (WTS) to solve the flight routes addition problem.

5.2. Weighted Tabu Search for flight routes addition problem

Now we elaborate the details of the WTS developed in this work. The search is an iterative process and the solution $s\prime$ of the next iteration is generated from the neighbor of the current solution s, supervised by a dynamically updating tabu list T.

5.2.1. 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 is selected from its neighbor $N(s')$.

Instead of the swapping operation in Vargo et al. (2010), we define $N(s)$ for our problem. A solution s contains the k edges to be added. The pth $(1 \leqslant p \leqslant k)$ edge e_{ij} in solution s connects two nodes v_i and v_j, which is shown in Fig. 2. All the edges incident to v_i or v_j in the set P form the sub-neighbor $N(s, p)$ of solution s for the pth edge. The edges which exist in G and are not in candidate set P are not displayed in Fig. 2. To prevent $N(s, p)$ from being empty, a random jump inside P is also included in $N(s, p)$, which jumps to another edge in P but not the current edges 1 to k in solution s. If the random jump gives an existing edge in $N(s, p)$, we execute another random jump. The neighbor of the current solution s is $N(s)$, which is the union of the sub-neighbors of all the edges in s:

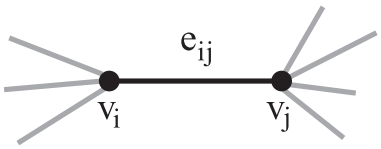

Fig. 2. The neighbor of the pth edge e_{ij} in the current solution s.

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

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

5.2.2. Tabu list

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

5.2.3. Aspiration criteria

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

The tabus will not be violated often because during the tabu search process, it is rare to find better λ_2's by reverse or repetitive moves.

5.2.4. 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, and λ_2^* and s^* record the best λ_2 and its corresponding s respectively. Line 4 shows that the algorithm terminates after Φ iterations. Lines 5 to 7 construct the sub-neighbors of the current solution. Line 9 forms s' from $N(s)$. Lines 10 to 13 check the aspiration criteria. Lines 14 to 16 check whether the move from s to s' is in the tabu list T.

Algorithm 2. Weighted Tabu Search

```
1:  given G(V, E_0), P and the edge weights in E_0
2:  randomly pick k edges from P to construct s_0
3:  s = s_0, λ_2^* = 0, s^* = s_0, T is set to an empty queue with the pre-fixed size |T|
4:  for iteration = 1 to Φ do
5:    for p = 1 to k do
6:      construct N(s, p) of the pth edge in s
7:    end for
8:    while 1 do
9:      pick one edge p' from each N(s, p) to construct s'
10:     if λ_2(s') > λ_2^*
11:       s = s', update T
12:       λ_2^* = λ_2(s), s^* = s
13:     end if
14:     if s' is not in T then
15:       s = s', update T
16:     end if
17:   end while
18: end for
19: output λ_2^* and s^*
```

6. Relaxed semidefinite programming

In this section we study the 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 a feasible solution with three different rounding techniques, which will be compared in terms of performance and computation time in this section.

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:

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

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:

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

The relaxed SDP formulation (17) is equivalent to (16), which is explicitly proved by Nagarajan et al. (2012). In this paper SeDuMi (Sturm, 1999) 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 feasible 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 of different rounding techniques

We generate a connected scale-free network with 20 nodes (the simulation setting details can be found at the beginning of the next section). Now k edges are going to be added onto the generated network. The results are presented in Fig. 3. Fig. 3a describes λ_2 as a function of k, while Fig. 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 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 the step by step method. We compare both the greedy and step by step rounding techniques in the next section with the MGP and the 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

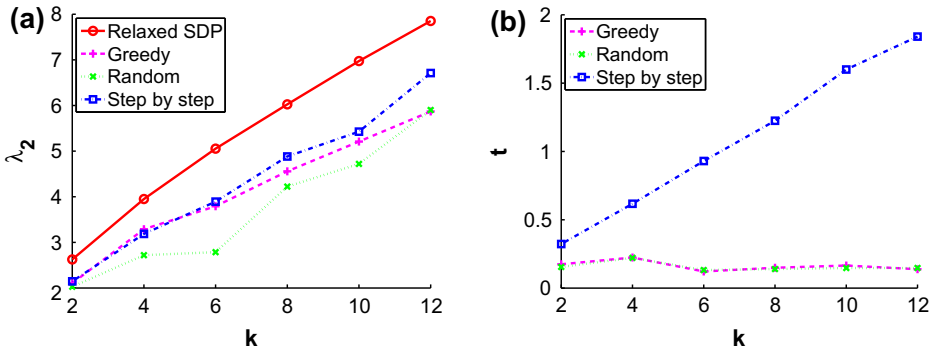

Fig. 3. The performance and computation time for three rounding techniques of the relaxed SDP.

assigned with weights $\{w_{ij}|w_{ij} = 1, 2 \text{ or } 3\}$. The number of edges to be added, k, is set as input. The total number of iterations in WTS, Φ, is set to be 1000.

7.1. Single edge addition experiment

An unweighted network with $n = 4$ is studied to show that the best edge to be added depends on the existing network topology. Then a weighted network of the same size is analyzed to show that the best edge 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 Fig. 4a and b. They are used as the existing network topologies in our experiment.

The candidate edges to be added to the existing topology α are e_{13}, e_{13} and e_{24}. The candidate edges for the existing topology β are e_{23}, e_{24} and e_{34}. We denote the algebraic connectivity of the existing topology as λ_2^0, the increased algebraic connectivity after adding one edge as λ_2' and $\Delta\lambda_2 = \lambda_2' - \lambda_2^0$. Tables 1 and 2 list when a single edge 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.

Tables 1 and 2 show that the edge additions and the algebraic connectivity increase are not trivially related (Jamakovic and Uhlig, 2007). The best edge to be added depends on the existing network topology. Since topology β is already a robust topology, a single edge addition does not increase the algebraic connectivity. On the other hand, topology α is vulnerable. So any one of the three candidate edges brings more robustness, especially when e_{14} turns the whole line topology into a circle topology.

To perform the analysis on the weighted networks, edge 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. Tables 3 and 4 demonstrate how much a weighted edge addition can increase algebraic connectivity.

Tables 3 and 4 show that the edge to be added is also related to the weighting scheme. The algebraic connectivity is monotone respect to the edge weights. Furthermore, Table 4 shows that the edge weights introduce more information to the existing network topology β. For example, in the unweighted network of topology β, the results of adding e_{23}, e_{24} and e_{34} are the same. However, to add e_{34} to the weighted network is apparently a worse option than the other two edges.

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 Fig. 5 is performed based on the default settings with k varying. In Fig. 5a we observe that when routes (edges) are added to the network G, the algebraic connectivity increases monotonically. The WTS offers the best λ_2 enhancement. The MGP and relaxed SDP with

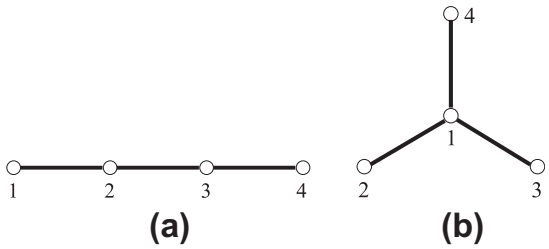

Fig. 4. The two minimum spanning tree topologies for four nodes.

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

Edge	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 edge addition analysis for the 4-node unweighted network with topology β.

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

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

Edge	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 edge addition analysis for the 4-node weighted network with topology β.

Edge	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

step by step rounding have the second best performance. So 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, Tables 5–7 are listed to observe the performance versus 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 decision makers 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 of $[0, 1]$. 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 three methods, which means that the decision makers should choose the MGP to obtain a shorter computation time. When η is large ($\eta = 0.9$), the WTS and the 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 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

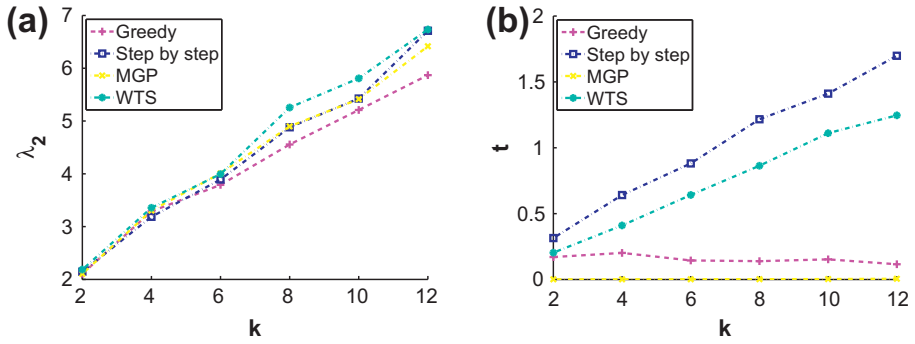

Fig. 5. The impact of k.

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

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 the 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 the 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. The x-axis is used to represent different tabu list lengths $|T|$ as $5, 10, 15, 20, 25, 30$. Fig. 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 neighbors and get out of the local optimum. However, a longer tabu list leads to slower computation, which is shown in Fig. 6b.

7.4. The impact of different initial edge positions

In this simulation we study the initial edge positions of WTS. The initial edge positions are the initial k routes that are randomly selected from the candidate set P and added onto the existing network $G(V, E_0)$. The parameter settings are $n = 20,\ k = 10,\ |T| = 40$. Fig. 7 illustrates that the WTS is not very sensitive to the initial k routes. The x-axis is the index

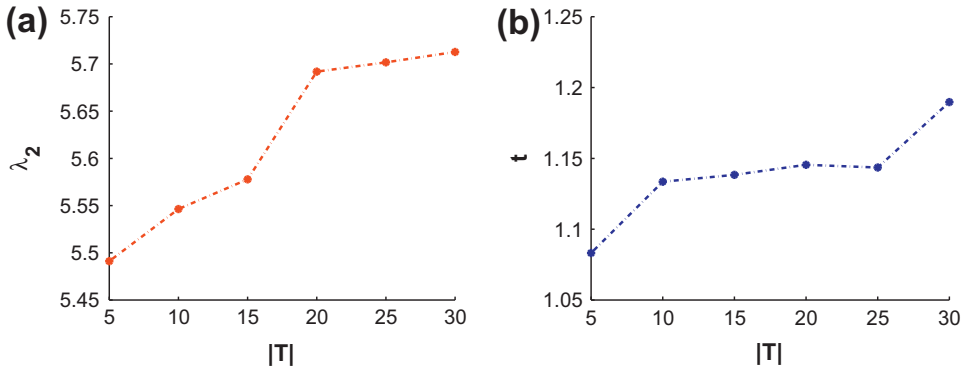

Fig. 6. The impact of $|T|$.

of different sets of initial edge positions. Six different sets 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 information 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 Fig. 8, we consider the 16 airports in the United States and obtain the adjacency matrix as Table 8. The 16 U.S. 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 the two major hubs of the entire network. Both of them 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 created 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 edge weights are mapped to different edge failure probabilities as shown in 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. Here the network failure means the network is disconnected. The results are shown in Table 10 with λ_2 sorted in ascending order.

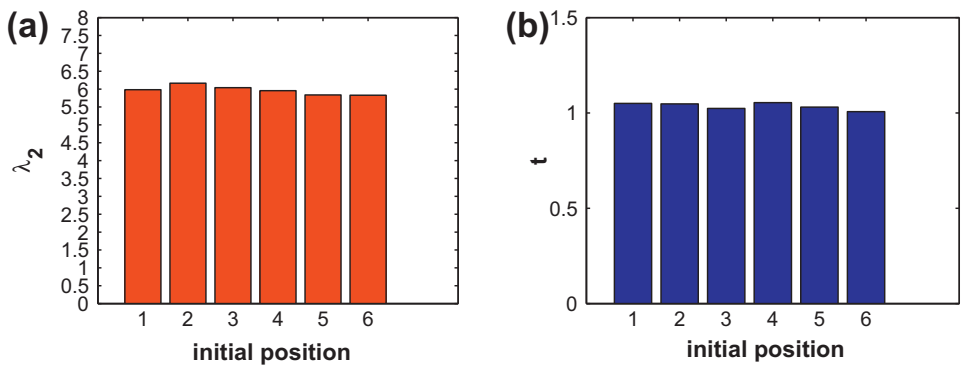

Fig. 7. The impact of initial edge positions.

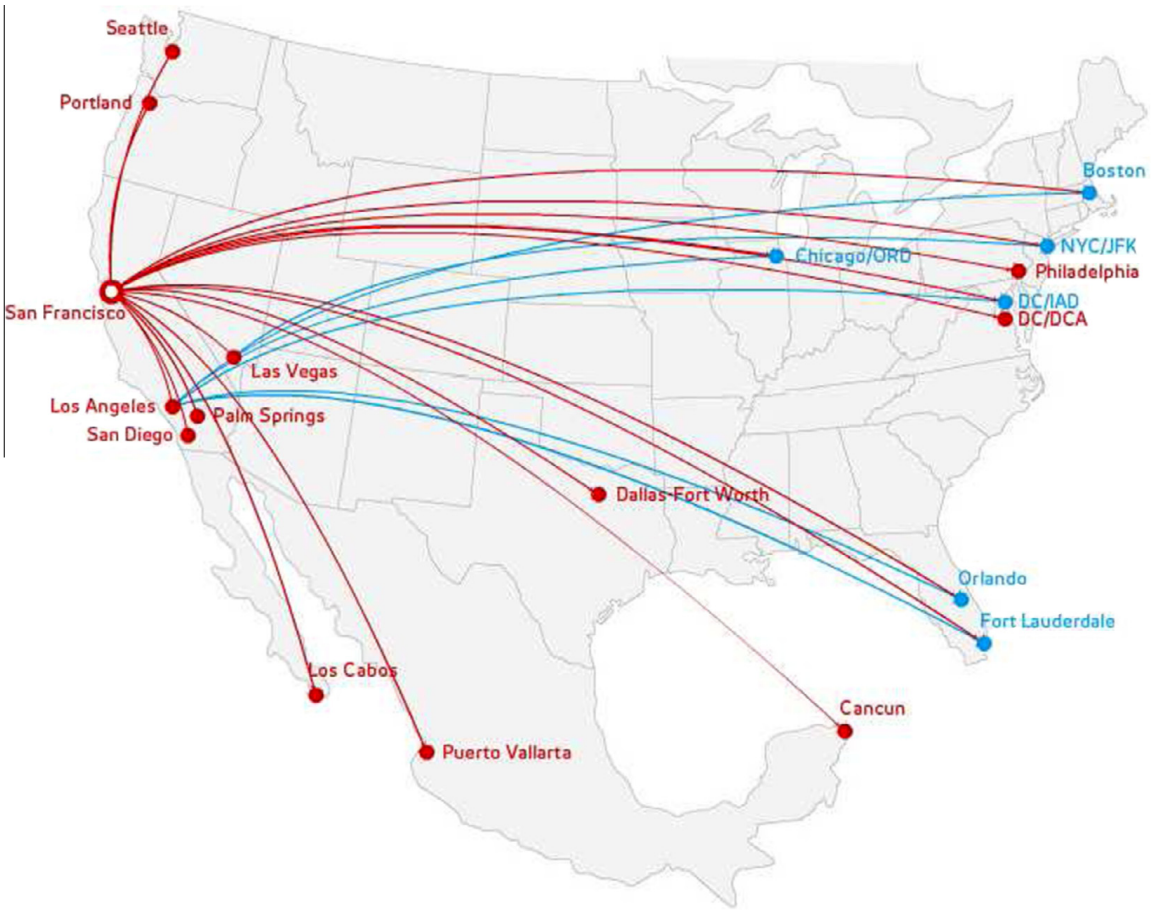

Fig. 8. An air transportation network route map for the Virgin America Airlines (obtained from (Virgin America, 2012)).

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

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Airport																
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 edge weights and their edge failure probabilities.

Edge weight w_{ij}	1	2	3
Edge 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 10,000 trials	1113	991	763	571	423	355

Table 11
The mapping between edge weights and the percentage of flight cancelation rate.

Edge weight w_{ij}	1	2	3
Edge 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

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

In this section 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 edge weight of this O–D pair will be assigned based on Table 11. The percentage of the canceled flights are from 0% to 5% in the historical data. We assign weight 3 to the edges with cancelation rate in $[0, 2\%)$, weight 2 to the edges with cancelation rate in $[2\%, 4\%)$ and weight 1 to the edges with cancelation rate in $[4\%, \infty)$. If there is more than one flight on an O–D pair, the probability of all the flights being canceled will be calculated by the joint probability of all the flights on this route and the corresponding edge weight will be assigned based on this joint probability by looking up Table 11.

The cancelation rates of the candidate routes to be added can be estimated from the historical route information data published by the FAA. Then the edge weights of the candidate routes are obtained through these estimated cancelation rates. In this section 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 be added to the current Virgin America network. The results have been shown in Table 12 and Table 13. We can see the trend that both MGP and WTS add more routes to DCA. At the same time, Palm Springs is another airport which both methods want to grow.

9. Conclusion

In this paper, we show that the algebraic connectivity is the most reasonable and efficient measurement for the air transportation network robustness. The flight routes addition/deletion problem is formulated to study the algebraic connectivity

of the air transportation network. 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 is a greedy heuristic method developed to compute the maximal λ_2. The Weighted Tabu Search is proposed to find the global optimum with longer computation time. Furthermore, the relaxed SDP with three rounding methods is adopted to provide the solution upper bound as well as the rounded feasible solutions. Numerical simulations have been performed to investigate the trade-off among the three methods and the real air transportation network of Virgin America is studied. Our results show that the MGP and the WTS should be adopted to solve large and small size networks respectively.

There was no existing literature defining the air transportation network robustness quantitatively. We propose to measure the air transportation robustness by the weighted algebraic connectivity. The weighted flight routes addition/deletion problem is formulated, with which the decision makers can maintain or modify the existing networks and design strategies for the future development of their air transportation networks. The MGP and WTS methods have been developed in this paper and the simulation results show that they outperform the benchmarking SDP method in either solution quality or computation speed. Our suggestions have been provided to the decision makers on how to select the appropriate algorithm according to the trade-off analysis. Furthermore, we are the first to analyze and optimize the robustness of the real air transportation network.

In the future we plan to evaluate the performances of the three methods and investigate their trade-offs in larger scale real air transportation networks with their international routes. Moreover, the top k flight routes to be added for legacy and low-cost carriers will be studied independently to achieve the maximal network robustness.

References

Wei, P., Sun, D., 2011. Weighted algebraic connectivity: an application to air transportation network. In: The 18th IFAC World Congress, Milan, Italy.

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

Vargo, E., Kincaid, R., Alexandrov, N., 2010. Towards Optimal Transport Networks, Systemics, Cybernetics and Informatics 8 (4), 59–64.

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

Bigdeli, A., Tizghadam, A., Leon-Garcia, A., 2009. Comparison of network criticality, algebraic connectivity, and other graph metrics. In: 1st Annual Workshop on Simplifying Complex Network for Practitioners, 4, Venice, Italy.

Jamakovic, A., Uhlig, S., 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.

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

Byrne, R.H., Feddema, J.T., Abdallah, C.T., 2009. Algebraic Connectivity and Graph Robustness, Tech. Rep., Sandia National Laboratories, Albuquerque, New Mexico 87185.

Gibbons, A., 1985. Algorithmic Graph Theory. Cambridge University Press.

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

FAA, 2010. FAA Aerospace Forecasts FY 2008–2025, Federal Aviation Administration.

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.

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

Alexandrov, N., 2004. Transportation Network Topologies, Tech. Rep., NASA Langley Research Center.

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.

Kim, Y., Mesbahi, M., 2006. On maximizing the second smallest eigenvalue of a state-dependent graph laplacian. IEEE Transactions on Automatic Control 51 (1).

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

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

Mohar, B., 1991. THE laplacian spectrum of graphs. Graph Theory, Combinatorics, and Applications 2, 871–898.

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

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.

Demmel, J., Dumitriu, I., Holtz, O., 2007. Fast linear algebra is stable. Numerical Mathematics 108 (1), 59–91, ISSN: 0029-599X.

Glover, F., 1989. Tabu Search – Part I. ORSA Journal on Computing 1, 190–206.

Glover, F., 1990. Tabu Search – Part II. ORSA Journal on Computing 2, 4–32.

Nagarajan, H., Rathinam, S., Darbha, S., Rajagopal, K., 2012. Algorithms for Synthesizing Mechanical Systems with Maximal Natural Frequencies, Non-linear Analysis – Real World Applications.

Sturm, J., 1999. Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. Optimization Methods and Software 11 (1), 625653.

Virgin America, 2012. Virgin America Flight Route Map. <http://www.virginamerica.com/travel/flight-routes.html>.