A Graph Theoretical Approach to Traffic Engineering and Network Control Problem

Ali Tizghadam
School of Electrical and Computer Engineering
University of Toronto, Canada
Email: ali.tizghadam@utoronto.ca

Alberto Leon-Garcia
School of Electrical and Computer Engineering
University of Toronto, Canada
Email: alberto.leongarcia@utoronto.ca

Abstract—**This paper looks at the problem of traffic engineering and network control from a new perspective. A graph-theoretical metric, betweenness, in combination with a network weight matrix is used to characterize the robustness of a network. Theoretical results lead to a definition of ”criticality” for nodes and links. It is shown that this quantity is a global network quantity and depends on the weight matrix of the graph. Strict convexity of network criticality is proved and an optimization problem is solved to minimize the network criticality as a function of weight matrix which in turn provides maximum robustness. Investigation of the condition of optimality suggests directions to design appropriate control laws and traffic engineering methods to robustly assign traffic flows. The choice of the path for routing the flow in these traffic engineering methods is in the direction of preserving the robustness of the network to the unforeseen changes in topology and traffic demands. The proposed method is useful in situations like MPLS and Ethernet networks where path assignment is required.**

Index Terms—**Robustness, Graph Theory, Betweenness, Autonomic, Traffic Engineering, Markov Theory.**

I. Introduction

From control-theoretical point of view, robustness is the capability of a network to keep itself in stable mode when changes take place in different parameters of the network due to uncertainties. In order to fix our notions of robustness we begin with our definition of robustness. There are three major types of changes that may affect the performance of the network: network topology, community of interest (active source-destination pairs), and traffic demand. Throughout this paper, we call a ”network control strategy” or a ”traffic engineering method” robust if its performance is not sensitive to changes in topology, traffic or community of interest. We aim to study the interaction between flow assignment and network structure with the help of graph-theoretical concepts.

In a previous work [1], we used a modified deterministic interpretation of ”betweenness centrality”, a metric from graph theory which characterizes the topological load of a node or link in a network graph. While the results were encouraging, the analytical study of the results was not feasible. In contrast, this paper uses a probabilistic interpretation of betweenness to investigate the problem of designing robust traffic engineering methods. In this new direction we were able to investigate the problem analytically using metrics from graph-theory. We have discovered some useful aspects of the robust network control and traffic engineering problem in networks.

The rest of this paper is organized as follows. Section II introduces the notion of criticality and investigates its properties thoroughly. Section III is dedicated in solving our convex optimization problem to minimize network criticality as a function of weight matrix. In section IV a traffic engineering method for robust routing and flow assignment is proposed using the results of previous sections. Section V provides some simulation results and validation for the case of MPLS networks in section . The paper is concluded in section VI.

II. Network Criticality

To model the robustness in a network, one needs to consider the topology as well as the effect of load on different nodes/links. In particular, the impact of a new flow on existing ones needs to be modeled. This motivates the use of betweenness metrics from graph theory. We consider the probabilistic definition of the node (link) betweenness as the main metric to quantify the criticality of a node or link and we use the criticality metric to model the degree of robustness of the network.

In [2] a probabilistic interpretation of the betweenness is defined based on random walks in a graph. A random-walk starts from a source node s, chooses one of the neighbors at random with equal probabilities, and uses the link between source s and that neighbor to get there. The random walk continues wandering around until reaches at a specified destination d, where it stops. The betweenness b_k of a node (link) k for source-destination pair (s,d) is the expected number of times that a random walk passes node k in its journey from source s to destination d. The total betweenness of node k is the sum of this quantity over all possible (s,d) pairs.

We use a generalized definition of random-walk betweenness based on the weighted adjacency matrix of a graph. To this end, we consider a network which is shown by its graph G=(N,E,W), where N, E, and W are the set of nodes, links, and weights of the graph respectively. Each node has a certain probability to send its data to the adjacent nodes. Let's assume a random walk at node s wants to go to node d as its final destination. Destination node is an absorbing state for this random walk and the walk is stopped in destination. The probability of passing node k in next step is shown by $p_{sk}(d)$ and defined as:

$$p_{sk}(d) = \frac{w_{sk}}{\sum_{q \in A(s)} w_{sq}}(1 - \delta_{sd}) \tag{1}$$

where $A(s)$ is the set of adjacent nodes of s and w_{sk} is the weight of link (s, k) (if there is no link between node s and k, then $w_{sk} = 0$), and δ_{sd} is the Kronecker delta function (i.e. if $s = d$, then $\delta_{sd} = 1$, otherwise $\delta_{sd} = 0$). The delta function in equation 1 is due to the fact that the destination node d is an absorbing node, and any random-walk coming to this state, will be absorbed or equivalently $p_{dk}(d) = 0$. Clearly, equation 1 defines a Markovian system.
Now, we define the node criticality for a weighted network simply as the random-walk betweenness of that node over the weight of the node.

$$\eta_k = \frac{b_k}{W_k}, \quad W_k = \sum_{j \in A(k)} w_{kj} \tag{2}$$

where η_k, b_k, W_k are the criticality, betweenness, and weight of node k (or weighted degree of the node) respectively. W_k is equal to the sum of all link weights incident to node k (weight of link (k,j) is shown by w_{kj}). Similarly, the link betweenness of link (i,j) (b_{ij})is defined as the expected number of times a random walk traverses the link summed over all source-destination pairs. The criticality of a link (i,j) (η_{ij}) is defined as the betweenness of the link over its weight: $\eta_{ij} = \frac{b_{ij}}{w_{ij}}$.
We will use criticality of the nodes (and links) to assess different networks based on their robustness to the changes in traffic demand, topology, and community of interest (source-destination pairs).

Observation 2.1: Equation 1 shows that if the weight increases, the goodness of that link (probability of being chosen) also increases. This means that for specific definition of weight, the QoS parameters which are in the direction of increasing the goodness (such as available bandwidth) should be positively related to the weight.

We assume that SLA (Service Level Agreement) parameters are already mapped to the weights with an appropriate method. Some of these methods are discussed in [3].

The betweenness of node (link) k for the source-destination pair (s, d) is denoted by $b_{sk}(d)$ and defined as the expected number of times that a random walk from s to d traverses k. Note that the path from i to k could be of length 0 to infinity. If we specify the probability values $p_{sk}(d)$ for destination d with matrix P_d, then for all $k \neq d$, the probability of entering node k at q^{th} step for different values of s and k can be obtained from corresponding entries of the matrix ${P_d}^q$ and in case of $k = d$ it would be 0. In our calculations, we treat destination d as a fixed point and write all matrices based on this assumption. At the end we obtain general results for our metrics by adding up the results for different destinations. In other words, the matrix P_d can be viewed as routing matrix to destination d when the random walk starts from node s. One can write this relationship in matrix form as follows:

$$B_d = \begin{cases} \sum_{q=0}^{\infty} P_d^q & if\ k \neq d \\ 0 & otherwise \end{cases} = \begin{cases} (I - P_d)^{-1} & if\ k \neq d \\ 0 & otherwise \end{cases} \tag{3}$$

where B_d is the betweenness matrix for destination d. By examining equation 3 one can easily see that the removal of column and row d from betweenness and probability matrices does not affect the other entries. We use $M(i|j)$ to denote a reduced matrix obtained by removing the i^{th} row and j^{th} column of matrix M. Equation 3 can be written as:

$$B_d(d|d) = (I - P_d(d|d))^{-1} \tag{4}$$

Let $W = [w_{ij}]$ be the weight matrix of the graph, D be the diagonal matrix of weighted degrees or graph nodes, and L be the Laplacian of the graph [4], [5]. We know that:

$$\begin{aligned} L &= D - W = \sum_{(i,j) \in E} w_{ij} u_{ij} u_{ij}^t \\ D &= diag(W_1, W_2, ..., W_n), \quad W_i = \sum_{k \in A(i)} w_{ik} \\ P_d(d|d) &= D^{-1}(d|d) \times W(d|d) \end{aligned}$$

where $u_{ij} = [0 \ldots 1 \ldots -1 \ldots 0]^t$ (1 and -1 are in i^{th} and j^{th} position). The last equation is the direct result of equation 1. Now we have:

$$I - P_d(d|d) = I - D^{-1}(d|d) \times W(d|d) = D^{-1}(d|d) \times L(d|d) \tag{5}$$

Replacing equation 5 in 4 results in:

$$B_d(d|d) = L^{-1}(d|d) \times D(d|d) \tag{6}$$

Note that graph $G(N, E)$ is assumed to be connected which means that the rank of graph Laplacian L is $(n - 1)$. As a result, the inverse of reduced Laplacian $L(d|d)$ exists and equation 6 has a unique solution.
Now we need to write equation 6 in terms of the Laplacian of the original graph.

Lemma 2.2: For entries of the reduced inverse of the Laplacian matrix, one can write:

$$(L^{-1}(d|d))_{sk} = l_{sk}^+ - l_{sd}^+ - l_{dk}^+ + l_{dd}^+ \tag{7}$$

where l_{sk}^+ shows the entry of row s and column k in Moore-Penrose inverse of Laplacian matrix L.

Proof: See Appendix A. ▮

■

Now, according to equation 6, we can obtain the betweenness of the node k for source-destination pair (s, d):

$$\begin{aligned} B_d(d|d) &= L^{-1}(d|d) \times D(d|d) \\ (B_d(d|d))_{sk} &= (l^+_{sk} - l^+_{sd} - l^+_{dk} + l^+_{dd}) \times W_k \\ \frac{b_{sk}(d)}{W_k} &= l^+_{sk} - l^+_{sd} - l^+_{dk} + l^+_{dd} \end{aligned}$$

To obtain the total betweenness of node k, we need to consider the effect of all source-destination pairs.

$$\begin{aligned} \frac{b_k}{W_k} &= \frac{1}{W_k}\sum_s\sum_d b_{sk}(d) = \frac{1}{W_k}\sum_s\sum_d \frac{b_{sk}(d) + b_{dk}(s)}{2} \\ \frac{b_k}{W_k} &= \sum_s\sum_d \frac{l^+_{dd} - l^+_{sd} - l^+_{ds} + l^+_{ss}}{2} \\ \frac{b_k}{W_k} &= \frac{1}{2}\sum_s\sum_d \tau_{sd} = \frac{1}{2}\tau \quad where \quad \tau = \sum_{s,d}\tau_{sd} \qquad (8) \\ \tau_{sd} &= l^+_{ss} + l^+_{dd} - 2l^+_{sd} \quad or \quad \tau_{sd} = u^t_{sd}L^+u_{sd} \qquad (9) \end{aligned}$$

To get equation 9 we used the fact that Laplacian matrix (and its Moore-Penrose inverse) is symmetric.
A similar result can be derived for a link of the graph as well. For a link (i,j), one can find the betweenness of a link based on the betweenness of its two end nodes.

Lemma 2.3: Betweenness of link $l = (i, j)$ is equal to $b_{ij} = b(l) = \tau w_{ij}$. Equivalently, the criticality of link (i,j) would be $\eta_{ij} = \eta(l) = \frac{b_{ij}}{w_{ij}} = \tau$.
Proof: See Appendix B ▮

■

Observation 2.4: Equation 8 and lemma 2.3 show that τ is independent of the node/link position. In fact τ is a global quantity of the network. From now on we call τ "network criticality". These equations show that node (link) criticality is a combination of a local parameter (weight) and a global metric (network criticality). This motivates the rest of our work in this paper. We investigate network criticality as a network-wide metric to capture and optimize network robustness. Further, equation 9, and lemma 2.3 give exact formulas to measure the generalized criticality of a node or a link of a graph. This allows analytical study of network robustness problem.

Before proceeding, we derive an alternative formula for τ which will be used later.

Lemma 2.5: τ is equal to $2nTr(L^+)$.
Proof:

$$\begin{aligned} \tau &= \sum_{s,d}\tau_{sd} = \sum_d\sum_s l^+_{ss} + \sum_s\sum_d l^+_{dd} - 2\sum_s\sum_d l^+_{sd} \\ &= n\sum_s l^+_{ss} + n\sum_d l^+_{dd} - 2\times 0 = 2n\sum_i l^+_{ii} = 2nTr(L^+) \end{aligned}$$

■

Now we are ready to investigate the convexity of τ.

Theorem 2.6: τ is a strictly convex function of graph weights. Further, τ is a non-increasing function of link weights.
Proof: We note that function $f(X) = Tr(X^{-1})$ is strictly convex on X, if X is positive definite (see [6]). Therefore, using equation $L^+ = (L + \frac{J}{n})^{-1} - \frac{J}{n}$ [6], where J is an $n \times n$ matrix with all elements equal to 1, we have $\tau = 2nTr(L^+) = 2nTr(L+\frac{J}{n})^{-1} - 2n$ is strictly convex on matrix $L + \frac{J}{n}$ (since L is positive semi-definite, $L + \frac{J}{n}$ is always positive definite).
It is also not difficult to show that $\frac{\partial\tau}{\partial w_{ij}} = -2n\|L^+u_{ij}\|^2$, which is always negative, therefore, τ is a monotone decreasing function of link weights.
▮

■

Theorem 2.6 has some direct consequences.

Observation 2.7: The problem of finding graph weights to optimize network criticality is a convex optimization problem and all the related literature can be used to solve it.

Observation 2.8: Due to the fact that the τ is a strictly convex function of the weights, an optimization problem with some constraints has a unique solution. As τ is a non-increasing function of weights, our optimization problem would be to minimize network criticality where some constraints are imposed on weight matrix.
The ultimate goal is to find a method to minimize network criticality. Hence, we consider the minimization of τ under some constraints. We set $\sum_{(i,j)\in E} z_{ij}w_{ij} = C$ as a reasonable constraint for our optimization problem, where z_{ij} is the cost of having link i, j, and C can be considered as the maximum fixed budget for total network weight.

Theorem 2.9: Consider the following optimization problem on graph G(N,E):

$$\begin{aligned} & Minimize \quad \tau \\ Subject\ to \quad & \textstyle\sum_{(i,j)\in E} z_{ij}w_{ij} = C \quad ,C\ is\ fixed \qquad (10) \\ & w_{ij} \geq 0 \end{aligned}$$

For the optimal weight set, W, the derivative of τ with respect to any link weight w_{ij} is proportional to the τ if the link weight is non-zero. More precisely, for the optimal weight set W^*:

$$w^*_{ij}(C\frac{\partial\tau}{\partial w_{ij}} + z_{ij}\tau) = 0 \quad (i, j) \in E \qquad (11)$$

Proof: See Appendix C. ▮

■

Observation 2.10: Equation 11 provides a set of control laws to guide the network to the direction of minimizing criticality. In other words, differential equations obtained in equation 11 give the dynamics of control agents in a distributed (or centralized) network control system. The controller design should be in the direction of minimizing the changes in network criticality.

Next section discusses the convex optimization problem 10 in detail.

III. Convex Optimization Problem for Network Criticality

In order to investigate the behavior of τ when the optimality condition is not met, we need to explore the properties of the optimization problem introduced in theorem 2.9. Our approach is to find an upper bound for the optimality gap (the difference between the optimal and sub-optimal objective values of the optimization problem). The goal is then to minimize this upper bound. In order to establish the upper bound, we use the fact that according to the duality theorem [7], any value of the dual objective function is a lower bound for the optimal value of the primal one. The duality gap, which is the difference between the value of the primal and dual objective functions, provides an upper bound for the optimality gap of the optimization problem In the optimal case the duality gap is zero because we optimize a strictly convex function of the variables, but in sub-optimal cases, it provides an upper bound for the optimality gap. We try to find this upper bound and use it as a metric to quantify the behavior of network criticality. This section tries to provide some directions towards this goal.

In order to investigate the behavior of τ when the optimality condition is not met, we need to explore the properties of the optimization problem introduced in theorem 2.9. We start by finding the dual of the optimization problem 10. Using lemma 2.5 and considering well-known equation $L^+ = \Gamma^{-1} - \frac{J}{n}$ ([8]), where $\Gamma = L + \frac{J}{n}$, the optimization problem 10 can be written as:

$$\begin{array}{rl} Minimize & 2nTr(\Gamma^{-1}) - 2n \\ Subject\ to & \Gamma = \sum_{(i,j)\in E} w_{ij}u_{ij}u_{ij}^t + \frac{J}{n} \\ & \sum_{(i,j)\in E} z_{ij}w_{ij} = C \quad ,C\ is\ fixed \\ & w_{ij} \geq 0 \end{array} \tag{12}$$

Now we write the lagrangian of the optimization problem 12.

$$L(\Gamma, W, T, \lambda, \rho) = 2nTr(\Gamma^{-1}) + Tr(T\Gamma) - 2n - C\lambda$$
$$- \sum_{(i,j)\in E} w_{ij}Tr(Tu_{ij}u_{ij}^t) + \sum_{(i,j)\in E} w_{ij}(\lambda z_{ij} - \rho_{ij}) - \frac{1}{n}\sum_{i,j\in N} t_{ij}$$

To find the dual formulation, it is enough to take the infimum of the lagrangian over Γ, W.

$$\begin{aligned} d(T,\lambda,\rho) &= inf_{\Gamma,W}L(\Gamma, W, T, \lambda, \rho) \\ &= inf_{\Gamma}Tr(2n\Gamma^{-1} + T\Gamma) + inf_W(-2n - C\lambda \\ &- \sum_{(i,j)\in E} w_{ij}(-u_{ij}^tTu_{ij} + \lambda z_{ij} - \rho_{ij}) - \frac{1}{n}\sum_{i,j\in N} t_{ij}) \end{aligned}$$

$$d(T,\lambda,\rho) = \begin{cases} inf_{\Gamma}Tr(2n\Gamma^{-1} + T\Gamma) - \frac{1}{n}\sum_{(i,j)\in E} t_{ij} - 2n - C\lambda \\ if \quad -u_{ij}^tTu_{ij} + \lambda z_{ij} - \rho_{ij} = 0 \quad and \quad T \geq 0 \\ -\infty \quad otherwise \end{cases}$$

where $T = [t_{ij}]$ and $\Gamma = [\gamma_{ij}]$, and $T \geq 0$ means that matrix T is positive semi-definite. We have also used equation $Tr(Tu_{ij}u_{ij}^t) = u_{ij}^tTu_{ij}$ to simplify Lagrangian. The infimum can also be obtained analytically.

$$-2n\Gamma^{-2} + T = 0 \;\Rightarrow\; T = 2n\Gamma^{-2} \quad and \quad \Gamma = (\frac{T}{2n})^{-\frac{1}{2}} \tag{13}$$

hence, we will have

$$inf_{\Gamma}(2nTr\Gamma^{-1} + T\Gamma) = Tr(2n(\frac{T}{2n})^{\frac{1}{2}} + T(\frac{T}{2n})^{\frac{-1}{2}}) = 2Tr(2nT)^{\frac{1}{2}} \tag{14}$$

From equation 13 one can see that if L be the optimal solution for Laplacian of the graph, then $T = 2n(L + \frac{J}{n})^{-2}$.

$$\begin{aligned} T &= 2n(L + \frac{J}{n})^{-2} \\ T\overrightarrow{1} &= 2n(L + \frac{J}{n})^{-2}\overrightarrow{1} \end{aligned}$$

where $\overrightarrow{1}$ is an $n \times 1$ column vector with all entries equal to 1. Further, we we know that:

$$\begin{aligned} (L + \frac{J}{n})\overrightarrow{1} &= L\overrightarrow{1} + \frac{1}{n}J\overrightarrow{1} = 0 + \frac{1}{n} \times n\overrightarrow{1} = \overrightarrow{1} \\ or \quad T\overrightarrow{1} &= 2n\overrightarrow{1} \end{aligned} \tag{15}$$

Equation 15 is valid for optimal solution of the optimization problem 13. Hence, we can use it as a constraint in our optimization problem. We need one more step to take before writing the dual optimization problem. We can remove the dual variable set ρ_{ij} considering the fact that $\rho_{ij} \geq 0$. Using this inequality in the condition part of the equation 13 results in

$$\begin{aligned} -u_{ij}^tTu_{ij} + \lambda z_{ij} - \rho_{ij} = 0 \quad and \quad \rho_{ij} \geq 0 \\ \Rightarrow -u_{ij}^tTu_{ij} + \lambda z_{ij} \geq 0 \\ or \quad \frac{1}{z_{ij}}u_{ij}^tTu_{ij} \leq \lambda \end{aligned}$$

Hence, the dual optimization problem can written as follows:

$$maximize \quad 2Tr(2nT)^{\frac{1}{2}} - 2n - C\lambda - \frac{1}{n}\overrightarrow{1}^tT\overrightarrow{1} \tag{16}$$
$$subject\ \ to \quad \frac{1}{z_{ij}}u_{ij}^tTu_{ij} \leq \lambda \tag{17}$$
$$T\overrightarrow{1} = 2n\overrightarrow{1}, \quad T \geq 0 \tag{18}$$

Lemma 3.1: The optimization problem 16 can be converted to the following equivalent one:

$$\begin{array}{rl} maximize & 2nC^{-1}(Tr(Q))^2 \\ subject\ \ to & \frac{1}{\sqrt{z_{ij}}}\left\|Q_i - Q_j\right\| \leq 1 \\ & Q\overrightarrow{1} = 0, \quad Q \geq 0 \end{array} \tag{19}$$

where $Q = (\frac{T-2J}{\lambda})^{\frac{1}{2}}$

Proof: See appendix D.

■ ■

Finally, we extract the exact solution of the optimization problem 19. According to equation 13 for the optimal solution of the optimization problem, one can write

$$\begin{aligned} T &= 2n\Gamma^{-2} = 2n(L + \frac{J}{n})^{-2} = 2n(L^+ + \frac{J}{n})^2 \\ &= 2n(L^+)^2 + 2n \times \frac{1}{n^2} \times n \times J = 2n(L^+)^2 + 2J \end{aligned} \tag{20}$$

to obtain equation 20 we used the fact that $L^+J = 0$ and $J \times J = nJ$. Now we use equation 20 to obtain matrix Q for the optimal case.

$$Q = (\frac{T-2J}{\lambda})^{\frac{1}{2}} = (\frac{2nL^{+2} + 2J - 2J}{\lambda})^{\frac{1}{2}} = (\frac{2n}{\lambda})^{\frac{1}{2}} L^+ \tag{21}$$

On the other hand

$$\begin{aligned} \lambda &= \max_{(i,j)\in E}(\frac{1}{z_{ij}} u_{ij} T u_{ij}^t) = \max_{(i,j)\in E}(\frac{1}{z_{ij}} u_{ij} 2n\Gamma^{-2} u_{ij}^t) \\ &= \max_{(i,j)\in E}(\frac{2n}{z_{ij}} u_{ij} L^{+2} u_{ij}^t) = \max_{(i,j)\in E}(\frac{2n}{z_{ij}} \left\|L^+ u_{ij}\right\|^2) \\ &= 2n \max_{(i,j)\in E} \frac{1}{z_{ij}} \left\|L_i^+ - L_j^+\right\|^2 \end{aligned} \tag{22}$$

Replacing equation 22 in 21 we get

$$\begin{aligned} Q &= (\frac{2n}{\lambda})^{\frac{1}{2}} L^+ = (2n)^{\frac{1}{2}} \frac{1}{(2n \max_{(i,j)\in E} \frac{1}{z_{ij}} \left\|L_i^+ - L_j^+\right\|^2)^{\frac{1}{2}}} L^+ \\ &= \frac{1}{\max_{(i,j)\in E} \frac{1}{\sqrt{z_{ij}}} \left\|L_i^+ - L_j^+\right\|} L^+ \end{aligned} \tag{23}$$

Equation 23 gives the matrix Q based on the Moore-Penrose inverse of the graph Laplacian L^+.

A. Main Result

Now we are ready to state the main result. Let τ^* denote the optimal value of τ. According to the duality theorem, the objective value of the dual problem is a lower bound for the optimal objective value of the primal optimization problem, thus:

$$\tau^* \geq \tau_{dual} \quad or \quad \tau - \tau^* \leq \tau - \tau_{dual} \tag{24}$$

Inequality 24 shows that the duality gap $\tau - \tau_{dual}$ provides an upper bound for the optimality gap $\tau - \tau^*$. The objective function of the dual problem 19 can be easily obtained by applying equation 23 to equation 19.

$$\begin{aligned} \tau_{dual} &= 2nC^{-1} Tr(Q)^2 |_{Q = \frac{1}{\max_{(i,j)\in E} \frac{1}{\sqrt{z_{ij}}} \left\|L_i^+ - L_j^+\right\|} L^+} \\ &= 2nC^{-1} \frac{1}{\max_{(i,j)\in E} \frac{1}{z_{ij}} \left\|L_i^+ - L_j^+\right\|^2} (TrL^+)^2 \end{aligned} \tag{25}$$

The difference between the primal and dual problem can be obtained as follows:

$$\begin{aligned} \tau - \tau_{dual} &= 2nTrL^+ - 2nC^{-1} \frac{1}{\max_{(i,j)\in E} \frac{1}{z_{ij}} \left\|L_i^+ - L_j^+\right\|^2} (TrL^+)^2 \\ &= 2nTrL^+ + (2n)^2 C^{-1} \frac{1}{-2n \max_{(i,j)\in E} \frac{1}{z_{ij}} \left\|L_i^+ - L_j^+\right\|^2} (TrL^+)^2 \\ &= \tau + C^{-1} \frac{\tau^2}{\min_{(i,j)\in E} \frac{1}{z_{ij}} \frac{\partial \tau}{\partial w_{ij}}} \\ \frac{\tau - \tau_{dual}}{\tau} &= 1 + \frac{\tau}{C \min_{(i,j)\in E} \frac{1}{z_{ij}} \frac{\partial \tau}{\partial w_{ij}}} \end{aligned} \tag{26}$$

We can summarize the result in the following theorem.

Theorem 3.2: Consider the following optimization problem:

$$\begin{aligned} & Minimize \quad \tau \\ Subject\ to \quad & \textstyle\sum_{(i,j)\in E} z_{ij} w_{ij} = C \quad ,C\ is\ fixed \\ & w_{ij} \geq 0 \end{aligned} \tag{27}$$

For any sub-optimal solution of this convex optimization problem, the deviation from optimal solution $(\frac{\tau - \tau_{dual}}{\tau})$ has the upper bound of $1 + \frac{\tau}{C \min_{(i,j)\in E} \frac{1}{z_{ij}} \frac{\partial \tau}{\partial w_{ij}}}$.

Theorem 3.2 can be used as the foundation for building centralized and distributed algorithms for flow assignment and routing problem. We will see one application of theorem 3.2 designing centralized traffic engineering algorithms in next section.

IV. Design of a Traffic Engineering Method

We now discuss how the analytical results extracted in previous section can be used to design a robust routing scheme that is able to cope with unpredicted changes in traffic and topology. Theorem 3.2 shows the control mechanism that needs to be implemented to maximize the robustness. The evolution of the management state should be in the direction of minimizing the upper bound of theorem 3.2. We define the cost of link $l = (i, j)$ as follows.

$$cost(l) = cost(i, j) = 1 + \frac{\tau}{\frac{C}{z_{ij}} \frac{\partial \tau}{\partial w_{ij}}} \tag{28}$$

Now we can apply the Dijkstra's algorithm to find the shortest path between every two nodes. This is the main idea of our traffic engineering algorithm, which is detailed in the following section.

A. Random Walk Path Criticality Routing (RW-PCR)

The idea of RW-PCR is simple. We label each and every link of the graph with its cost according to equation 28 and use Dijkstra's algorithm to obtain the shortest path(s) from a source s to a destination d using the assigned cost for the links. Bear in mind that this cost is not the same as w_{ij}. In fact the cost has the role of Link

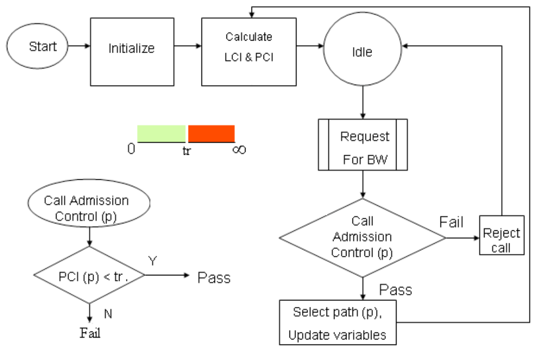

Fig. 1. Flowchart of RW-PCR Algorithm

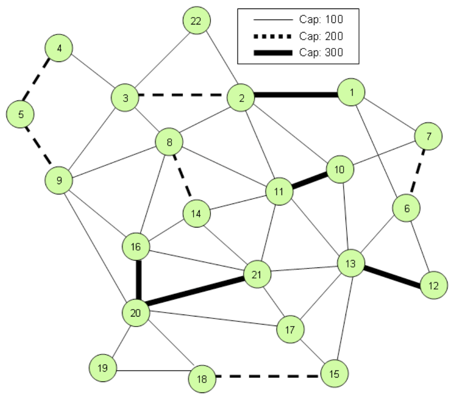

Fig. 2. Test Network to Evaluate RW-PCR Algorithm

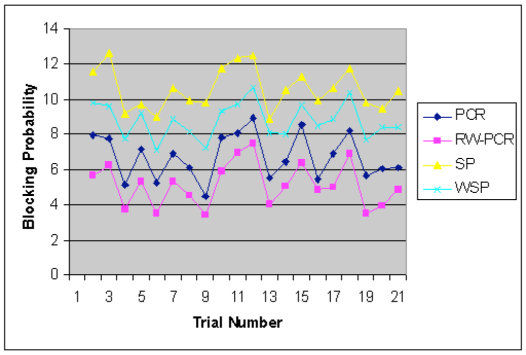

Fig. 3. Static Case- Result of Applying PCR, RW-PCR, SP, and WSP to the Network under Test

Criticality Index (LCI) in our heuristic Path Criticality Routing (PCR) algorithm described in [1].

Definition 4.1: We call $1 + \frac{\tau}{\frac{C}{z_{ij}} \frac{\partial \tau}{\partial w_{ij}}}$ as Random-Walk Link Criticality Index (LCI_{RW}).

Definition 4.2: We define the maximum of the LCI_{RW} of the links in a path as Random-Walk Path Criticality Index (PCI_{RW}).

When a demand for source-destination pair $s-d$ arrives, the shortest path obtained in this way would be considered as a candidate to be assigned to the demand. A simple call-admission control is applied here by considering a threshold tr for the criticality of the path. If the PCI_{RW} is more than this threshold, then the flow would be considered too risky for the network and be rejected (blocked), otherwise the path is used as the route and the demand flow is assigned to this path. The available bandwidth of all the links on this path is updated and the LCI_{RW}'s are also modified accordingly. In case of having more than one shortest path, the path with least PCI_{RW} will be chosen. A simple flowchart of RW-PCR algorithm is shown in Fig. 1.

1) Time Complexity of RW-PCR Argorithm: To estimate the time complexity of the algorithm, we note that the running time to get the Moore-Penrose inverse is $O(mn^{\frac{1}{2}})$ [6], where m and n are the number of links and nodes in the graph respectively. The main part of the RW-PCR can be obtained in $O(nlog(n))$ as it is just a shortest path algorithm with link costs. Therefore, the complexity of the algorithm would be $O(mn^{\frac{3}{2}}log(n))$.

V. Evaluation

In order to investigate the effectiveness of our RW-PCR algorithm, we conduct a set of experiments on the network of Fig. 2. We apply RW-PCR to create LSPs (Label Switch Path) assuming that MPLS is used in the network to create the paths. In our simulations we assumed $z_{ij} = 0 \ \forall (i, j) \in E$. In the first experiment the requests for LSPs arrive at random and stay forever (no departures). In our tests the bandwidth requests for paths (LSPs) are taken to be uniformly distributed between 1 to 3 units. In Fig. 3 we show the percentage of rejected calls for this case and compare the performance to that of original deterministic PCR ([1]), shortest path (SP), and widest shortest path (WSP). The test is performed 20 times and each time with 2000 path requests. We measured the number of blocked requests. Fig. 3 shows that RW-PCR algorithm has the best performance and the performance of PCR and RW-PCR algorithms are much better than WSP and SP.

In another experiment we examined the behavior of the algorithms in the presence of dynamic traffic. Path requests arrive between each source-destination point (which is chosen at random) according to a Poisson process with an average rate λ, and the holding times are exponentially distributed with mean μ and $\frac{\lambda}{\mu} = 1800$. The bandwidth of input demands are taken to be uniformly distributed between 1 to 3 units and 7000 demands are generated in each experiment. Fig. 4 shows the percentage of the path requests rejected in 20 experiments (blocking probability).

From Fig. 4, one can see that the number of blocks in RW-PCR is less than other algorithms. PCR has the second best performance and WSP and SP are in next positions. The main reason for the success of RW-PCR is the fact that RW-PCR finds the path that has minimum

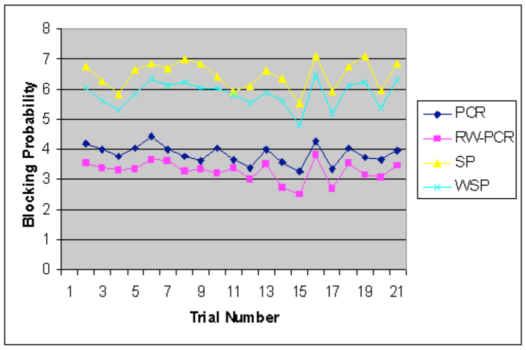

Fig. 4. Dynamic Case- Result of Applying PCR,RW-PCR, SP, and WSP to the Network under Test

effect on network criticality (or betweenness sensitivity).

VI. Conclusion

In this paper we analyzed the robustness of a network to unexpected changes in different parameters based on facts from graph theory. Network criticality is introduced as the main metric to quantify robustness. This metric is investigated in deep, and it is shown that it captures some global properties of a network including the effect of topology and load. we introduced a strictly convex optimization problem based on the properties of network criticality and proposed a control law to engineer the traffic of the network towards keeping the criticality as low as possible which in turn guarantees robustness.
We believe that the provided framework opens different venues for further research in traffic engineering as well as network design problem. Our results can be used to design different distributed control mechanisms (for example in multi-agent control networks), or distributed mechanisms for traffic engineering in core networks where robustness is required. The proposed optimization problem can be also used in network planning where the capacity assignment or topology design is the goal.

Appendix A
Proof of Lemma 2.2

We use small English letters to show column vectors and small Greek letters to show row vectors. We also use subscript to show the order of a vector. For example z_{n-1} is a $n-1\times 1$ column vector and υ_{n-1} is a $1\times n-1$ row vector. Without loss of generality, we rename the nodes so that the removed node becomes the last node of the graph (node n). Now, in order to write $L^{-1}(n|n)$ in terms of L, we use the Moore-Penrose generalized inverse matrix of L ([6]). The Moore-Penrose inverse of $L(n|n)$ and the $L^{-1}(n|n)$ are equal since $L(n|n)$ is an $(n-1)\times(n-1)$ matrix with rank $n-1$. In other words, $L(n|n)$ is full-rank and its inverse is the same as its Moore-Penrose inverse. To obtain L from $L(n|n)$, we first add a column to $L(n|n)$ to get:

$$Q = [L(n|n)\ \ z_{n-1}]$$

The column-vector z_{n-1} has to be chosen in a way to make the sum of every row of the matrix Q equal to zero. We use the following formula from [6] which is a recursive formula to obtain the Moore-Penrose inverse of a matrix when a column is added to the original matrix. Let $A \in \mathbb{F}^{p\times q}$ be a $p\times q$ matrix and $b \in \mathbb{F}^{p}$ be a $p\times 1$ column vector.

$$\begin{pmatrix} A & b_p \end{pmatrix}^{+} = \begin{pmatrix} A^{+}(I - b_p\zeta_p) \\ \zeta_p \end{pmatrix}, \quad \zeta_p = \{\begin{smallmatrix} (b_p - AA^{+}b_p)^{+} & if\ \ b_p \neq AA^{+}b_p \\ \frac{b_p^{*}(AA^{*})^{+}}{1+b_p^{*}(AA^{*})^{+}b_p} & b_p = AA^{+}b_p \end{smallmatrix}. \tag{29}$$

where ζ_p is a $1\times p$ row vector and $*$ means *conjugate transpose*. To satisfy the requirement of Laplacian matrix we need to have

$$[L(n|n)\ \ z_{n-1}]\overrightarrow{1}_n = 0 \tag{30}$$

From 30 one can easily see that:

$$L(n|n)\overrightarrow{1}_{n-1} + z_{n-1} = 0 \Rightarrow z_{n-1} = -L(n|n)\overrightarrow{1}_{n-1} \tag{31}$$

Now from 29 by replacing $A = L(n|n)$ and using 31, one can see:

$$\begin{aligned} Q^{+} &= \begin{pmatrix} L(n|n) & z_{n-1} \end{pmatrix}^{+} \\ &= \begin{pmatrix} L^{+}(n|n) - L^{+}(n|n)z_{n-1}\zeta_{n-1} \\ \zeta_{n-1} \end{pmatrix} \\ &= \begin{pmatrix} L^{+}(n|n) + L^{+}(n|n)L(n|n)\overrightarrow{1}_{n-1}\zeta_{n-1} \\ \zeta_{n-1} \end{pmatrix} \\ &= \begin{pmatrix} L(n|n)^{+} \\ 0 \end{pmatrix} + \overrightarrow{1}_n\zeta_{n-1} \end{aligned} \tag{32}$$

Equation 32 asserts that:

$$\begin{aligned} q^{+}_{sk} &= (L^{+}(n|n))_{sk} + (\zeta_{n-1})_k \quad if \quad s \neq n \\ q^{+}_{nk} &= 0 + (\zeta_{n-1})_k \qquad\qquad\quad if \quad s = n \end{aligned}$$

Subtracting these two equations show that:

$$\Rightarrow (L^{+}(n|n))_{sk} = q^{+}_{sk} - q^{+}_{nk} \tag{33}$$

With the same approach , we add the n^{th} row to Q to obtain the $n\times n$ Laplacian matrix L: L = $\begin{bmatrix} Q \\ d \end{bmatrix}$ With similar reasoning and using equation 29 one can obtain:

$$\Rightarrow q^{+}_{sk} = l^{+}_{sk} - l^{+}_{sn} \tag{34}$$

Using equations 33, 34 we can find our desired result.

$$(L^{+}(n|n))_{sk} = l^{+}_{sk} - l^{+}_{sn} - l^{+}_{nk} + l^{+}_{nn} \tag{35}$$

but $L^{+}(n|n) = L^{-1}(n|n)$, so:

$$(L^{-1}(n|n))_{sk} = l^{+}_{sk} - l^{+}_{sn} - l^{+}_{nk} + l^{+}_{nn}$$

Appendix B
Proof of Lemma 2.3

$$\begin{aligned} b_{ij} &= \frac{w_{ij}}{\sum_k wik}b_i + \frac{w_{ji}}{\sum_k wkj}b_j \\ &= \frac{w_{ij}}{W_i}\frac{1}{2}\tau W_i + \frac{w_{ji}}{W_j}\frac{1}{2}\tau W_j \\ &= \frac{1}{2}(w_{ij} + w_{ji})\tau = w_{ij}\tau \quad or \quad \frac{b_{ij}}{w_{ij}} = \tau \end{aligned}$$

Appendix C
Proof of Theorem 2.9

In order to proceed we need the following fact:

Lemma C.1: For any weight matrix W on the graph: $\nabla\tau.Vec(W) + \tau = 0$, where $Vec(W)$ is a vector obtained by appending all the rows of matrix W to get a vector of w_{ij}'s.

Proof: In lemma 2.3 we scale all the link weights with parameter t.

$$\tau(tVec(W)) = \frac{1}{t}\tau(Vec(W)) \tag{36}$$

By taking the derivative of τ with respect to t, we have

$$Vec(W)^t\nabla\tau = \frac{-1}{t^2}\tau(W) \tag{37}$$

It is enough to consider equation 37 at $t = 1$ to obtain $Vec(W)^t\nabla\tau + \tau = 0$.

■ ■

Now, to conclude the proof of theorem 2.9 we notice that for optimal weight matrix W^* we can write

$$(Vec(Z).Vec(W^*))\tau = (\sum_{(i,j)\in E} w^*_{ij}z_{ij})\tau = C\tau \tag{38}$$

Combining lemma C.1 and equation 38 one can see

$$\begin{aligned} C\nabla\tau.Vec(W^*) + Vec(Z).Vec(W^*)\tau &= 0 \\ Vec(W^*).(C\nabla\tau + \tau Vec(Z)) &= 0 \\ w^*_{ij}(C\frac{\partial\tau}{\partial w_{ij}} + \tau z_{ij}) &= 0 \end{aligned}$$

Appendix D
Proof of Lemma 3.1

We apply the change of variable $Q = (\frac{T-2J}{\lambda})^{\frac{1}{2}}$ to equation 18:

$$\begin{aligned} T &= \lambda Q^2 + 2J \\ T\overrightarrow{1} &= \lambda Q^2\overrightarrow{1} + 2J\overrightarrow{1} \\ 2n\overrightarrow{1} &= \lambda Q^2\overrightarrow{1} + 2n\overrightarrow{1} \\ \lambda Q^2\overrightarrow{1} &= 0 \\ Q\overrightarrow{1} &= 0 \quad (\textit{bear in mind} \quad Q \geq 0 \quad \textit{since} \quad T \geq 0) \end{aligned}$$

Now we apply the variable change to the dual function.

$$\begin{aligned} (2nT)^{\frac{1}{2}} &= (2n)^{\frac{1}{2}}(\lambda Q^2 + 2J)^{\frac{1}{2}} = (2n)^{\frac{1}{2}}(\lambda Q^2 + (\frac{2}{n})J^2)^{\frac{1}{2}} \\ &= (2n)^{\frac{1}{2}}(\lambda^{\frac{1}{2}}Q + (\frac{2}{n})^{\frac{1}{2}}J) = (2n\lambda)^{\frac{1}{2}}Q + 2J \\ \Rightarrow Tr((2nT)^{\frac{1}{2}}) &= (2n\lambda)^{\frac{1}{2}}Tr(Q) + 2Tr(J) \\ &= (2n\lambda)^{\frac{1}{2}}Tr(Q) + 2n \end{aligned} \tag{39}$$

Now we apply the equation 39 to the dual function 16.

$$\begin{aligned} d(T,\lambda) &= 2Tr(2nT)^{\frac{1}{2}} - 2n - C\lambda - \frac{1}{n}\overrightarrow{1}^tT\overrightarrow{1} \\ &= 2(2n\lambda)^{\frac{1}{2}}Tr(Q) + 4n - 2n - C\lambda \\ &- \frac{1}{n}\lambda\overrightarrow{1}^tQ^2\overrightarrow{1} - \frac{1}{n}\times 2\overrightarrow{1}^t\overrightarrow{1}\overrightarrow{1}^t\overrightarrow{1} \\ &= 2(2n\lambda)^{\frac{1}{2}}Tr(Q) + 4n - 2n - C\lambda - 0 - 2n \\ &= 2(2n\lambda)^{\frac{1}{2}}Tr(Q) - C\lambda \end{aligned}$$

The first constraint of the dual problem 16 (constraint 17)can be written in terms of new variable Q as follows:

$$\begin{aligned} \frac{1}{z_{ij}}u^t_{ij}Tu_{ij} &\leq \lambda \\ \frac{1}{z_{ij}}(\lambda u^t_{ij}Q^2u_{ij} + u^t_{ij}Ju_{ij}) &\leq \lambda \\ \frac{1}{z_{ij}}(\lambda(u^t_{ij}Q^t)(Qu_{ij}) + 0) &\leq \lambda \\ \frac{1}{z_{ij}}\lambda(Qu_{ij})^t(Qu_{ij}) &\leq \lambda \\ \frac{1}{z_{ij}}\left\|Qu_{ij}\right\|^2 &\leq 1 \\ \frac{1}{\sqrt{z_{ij}}}\left\|Q_i - Q_j\right\| &\leq 1 \end{aligned}$$

where Q_i shows the i^{th} column of matrix Q. To obtain this inequality we used the facts that Q is symmetric and $Ju_{ij} = 0$. Our dual optimization problem now can be written as:

$$\begin{aligned} \textit{maximize} \quad & 2(2n\lambda)^{\frac{1}{2}}Tr(Q) - C\lambda \\ \textit{subject to} \quad & \tfrac{1}{\sqrt{z_{ij}}}\left\|Q_i - Q_j\right\| \leq 1 \\ & Q\overrightarrow{1} = 0, \quad Q \geq 0 \end{aligned}$$

The maximization over λ in this optimization problem can be done analytically.

$$\begin{aligned} d(Q,\lambda) &= 2(2n\lambda)^{\frac{1}{2}}Tr(Q) - C\lambda \\ \frac{\partial d}{\partial\lambda} = 0 &\Rightarrow 2\times(2n)^{\frac{1}{2}}\times\frac{1}{2}\lambda^{-\frac{1}{2}}Tr(Q) - C = 0 \\ \lambda &= 2nk^{-2}(Tr(Q))^2 \\ d(Q) &= 2(2n)^{\frac{1}{2}}\times(2n)^{\frac{1}{2}}C^{-1}(Tr(Q))^2 - C\times 2nC^{-2}(Tr(Q))^2 \\ &= 2nC^{-1}(Tr(Q))^2 \end{aligned} \tag{40}$$

Using equation 40, the final form of our dual optimization problem would be

$$\begin{aligned} \textit{maximize} \quad & 2nC^{-1}(Tr(Q))^2 \\ \textit{subject to} \quad & \tfrac{1}{\sqrt{z_{ij}}}\left\|Q_i - Q_j\right\| \leq 1 \\ & Q\overrightarrow{1} = 0, \quad Q \geq 0 \end{aligned}$$

References

[1] A. Tizghadam and A. Leon-Garcia. A Robust Routing Plan to Optimize Throughput in Core Networks. *ITC20, Elsvier*, pages 117–128, 2007.

[2] M. Newman. A Measure of Betweenness Centrality Based on Random Walks. *arXiv cond-mat/0309045.*, 2003.

[3] P. Van Mieghem and F. A. Kuipers. Concepts of Exact QoS Routing Algorithms. *IEEE/ACM TRANSACTIONS ON NETWORKING*, 12(5):851–864, October 2004.

[4] Michael William Newman. *The Laplacian Spectrum of Graphs*. PhD thesis, Department of Mathematics, University of Manitoba, July 2000.

[5] Fan R. K. Chung. *Spectral Graph Theory*. CBMS Regional Conference Series On Mathematics, No. 92. American Mathematical society, 1997.

[6] Dennis S. Bernstein. *Matrix Mathematics*. Prinston University Press, 2005.

[7] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, April 2003.

[8] C. R. Rao and S. K. Mitra. *Generalized Inverse of Matrices and its Applications*. John Weily and Sons Inc., 1971.