2014

Dynamic State Determination of a Software-Defined Network via Dual Basis Representation

Parker, Thomas

Dynamic State Determination of a Software-Defined Network via Dual Basis Representation

Thomas Parker, Jamie Johnson, Murali Tummala, John McEachen, James Scrofani

Department of Electrical and Computer Engineering
Naval Postgraduate School, Monterey, CA
{tcparke1, jljohns2, mtummala, mceachen, jwscrofa}@nps.edu

***Abstract*— To maximize the performance of a software-defined network, a network observer must develop a state that can be tracked and controlled. We propose a novel method that uses the entire eigenspace of the Laplacian matrix to determine the state of a SDN. Our approach exploits the double orthogonality of the Laplacian matrix in order to define the dual basis. Each basis uses the entire reachability space with the objective of fully describing the centrality of each node over time. The reachability space is defined by the dual basis once the null space has been removed. The definition of the dual basis allows the network controller to observe the network state to determine which areas are most utilized and least utilized. Once the state has been estimated, the controller may choose to correct the network state by rerouting flows or preventing additional flows.**

Keywords—Software-defined network; Laplacian matrix; algebraic connectivity; spectral graph theory; control theory

I. INTRODUCTION

Software-defined networking (SDN) is poised to completely change the way large, complex networks are managed and controlled. Software-defined networking is first the concept that centralizes the controller, which provides unprecedented control over packet routes and collection of network statistics [1]. SDN is also the network that implements this concept. To implement closed loop control, the centralize controller measures network flows and determines the network state. The success of these algorithms is dependent on the ability of the observer to provide an accurate estimate of the current state.

In the context of classical control theory, closed loop control requires an accurate observer in the system to measure network flows and to determine the system state. To close the control loop in a SDN, the observer must pass this information to the controller, which can adjust the flows to improve the network's performance. The SDN switches are the devices where control signals are applied. The flows are the objects to be controlled [2]. The network traffic is the object that is measured. Optimal solutions have been developed for the determination of flows in a network, but it was not shown how the controller would develop a state for the network [3].

A. Closed Loop Control of SDNs

A goal of SDN is to improve performance of a network by centralizing the controller, which allows the network to be modelled as a closed loop control system. Networks can now be considered a control system that requires optimal estimation and optimal control. Standard linear control system methods do not easily apply to complex networks. It is difficult to develop a linear model of a network because of the number of protocols that are present and coupling caused by physically attaching devices to each other. The development of a method to determine the state of a SDN has not been sufficiently explored. There is a need to develop an analytical method to determine the time-varying network state.

B. SDN Network Observer

To develop a network observer for an SDN, graph theory must be explored for possible solutions to the state determination problem. Graph theory is an analytical tool that is used to model complex networks. A state-dependent graph is one in which the network model includes time-varying weights [4]. In many cases, time-varying weights between nodes in the graph are based on measurements that are made within the network. Round-trip time, signal-to-noise ratio, and data rate are all examples of measurements that can be used to weigh each of the links [5]. However, the state-dependent graph does not provide the state of the network. Spectral graph theory provides the tools necessary to develop the state of the network.

Spectral graph theory is a subfield of graph theory that uses the eigendecomposition of graph theory defined matrices to gain a better understanding of the properties of a given graph. One of the main applications of spectral graph theory has been to determine how to rearrange the links in the graph to maximize the robustness or connectivity of the graph. It has been shown the state provided by spectral graph theory is well correlated with robustness and performance of various networks [6] [7].

Our method combines spectral graph theory and SDN to develop a network observer that produces the network state and passes that information to the controller. In our method, the state of the network is the dual basis, which is the eigenvector matrix of the state-dependent graph theory model [8]. Based on the eigenvector matrix, the observer is able to pass the congested and underutilized areas of the network to the controller. The controller's routing function can then use that information to determine the optimal routes in a network to reduce congestion and improve performance.

This paper is organized as follows. In section II, we describe how a SDN is modelled in matrix form, propose a

method to weigh each link and then describe the eigendecomposition of the weighted graph. Section III describes the dual basis, eigencentrality basis, and nodal basis. Section IV shows the simulations of the modeled network and analysis of the results. Section V provides our conclusions and future work.

II. Eigendecomposition of a SDN

To gain the most benefit from using a SDN, the controller must have a monitoring function that determines the state of the network. In control theory, this function is known as the observer. A controller can request a large quantity of information from the network, but unless it is able to process the data in a useful manner, the benefit of having this data is lost. The state determination must be accomplished efficiently in order to ensure the computed state has not changed significantly from the current state. The eigendecomposition produces a state that is complete and uncorrelated [8]. A mathematical representation of the network is required first and then the eigendecomposition can be applied to this representation to determine the state.

A. Graph representation of a SDN

Fig. 1 shows the physical topology of a SDN. The controller network has been ignored, but the controller network needs to be represented and monitored as well. The focus of this paper will be on monitoring the production traffic network. In Fig. 1, there are seventeen SDN switches that are connected by 132 links. The full mesh network switches are the light blue circles, the switches with a various number of links are the green circles, the disconnected node is the red circle and the dark blue lines are the links connecting the nodes. The disconnected node is one that is either physically disconnected due to a network device failure or it has been overwhelmed by traffic such as in a denial-of-service attack. The full mesh is intended to model a service provider's core network, and the outlying nodes are the access network onto which the customers are connected.

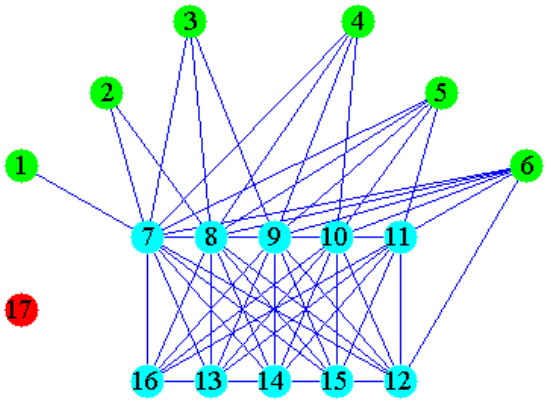

Fig. 1. Seventeen node network with a ten node core network, six access network nodes, and one disconnected node

The Laplacian matrix is a well-documented graph theory matrix representation of networks [9]. The Laplacian matrix is a combination of the adjacency matrix, A, which describes how nodes are connected, and the degree matrix, D, which describes how many links connect each node to the network [7]. In a weighed graph, $G=(N,L,W)$, N is the set of nodes that are connected by the links, l_{ij}, in set L. Each link is given a weight, w_{ij}, based on a link metric [4].

For the purpose of monitoring the state of the SDN, we propose using link utilization as the link metric. Link utilization is a value between zero and one that represents the fraction of the bandwidth that is utilized at any given time. In general the link weight is defined as

$$w_{ij}=1-\frac{x_m(t)}{x_{\max}} \tag{1}$$

where $x_m(t)$ is the network measurement as a function of time, and $x_{\max}$ is the fixed, maximum allowed metric. For most applications, the denominator of (1) will not change, but in certain instances it could change. There are many applications in which the bandwidth may vary with time; these include wireless and military applications. The model allows for this variation by dynamically changing the denominator. We propose using link utilization, which would use maximum bandwidth as the denominator in (1), and the measured current data rate as the numerator. This choice of definition is used to ensure that the eigenvalues decrease as a function of increased network traffic, i.e. smaller eigenvalues are associated with less connectedness. These definitions follow the conventional approach to spectral graph theory.

The Laplacian matrix is defined as

$$Q=D-A \tag{2}$$

To define it directly, (2) can be restated as

$$Q_{ij}=\begin{cases}-w_{ij} & \text{if } l_{ij}\in L\\ \sum_{i=1}^{n} w_{ij} & \text{if } i=j\\ 0 & \text{otherwise}\end{cases} \tag{3}$$

Network traffic is dynamic, and therefore, the representation of the network must be dynamic. The Laplacian matrix alone does not provide any information, but it allows the monitoring function of the controller to sort and organize the information requested from the switches.

B. Eigendecomposition of a Weighted Graph

The Laplacian matrix is positive semi-definite; therefore, all eigenvalues are real, and there will always be at least one zero eigenvalue [9]. The eigenvalues are typically ordered from smallest to largest by

$$0=\lambda_1\le\lambda_2\le\cdots\le\lambda_{n-1}\le\lambda_n \tag{4}$$

Additional zero eigenvalues are interpreted as additional disconnected nodes. For each connected group within the graph model, there will be a zero eigenvalue [9].

To improve performance, it has been proposed to maximize the smallest non-zero eigenvalue, algebraic connectivity [10]. Maximizing algebraic connectivity does lead to improved

performance in some cases, but it is not enough to determine the state of the network. This maximization only minimizes the difference between the largest and smallest eigenvalues. In an unweighted graph, the largest eigenvalue is bound by [9]

$$\lambda_n \le n \tag{5}$$

Eqn. (5) indicates that the maximum eigenvalue is bounded by the number of nodes, and not links or link weights. Using the smallest as a metric alone, only provides insight into the least connected area of the network. It provides no information about any other portion of the network. This could provide a false state of the network.

In the context of a SDN, each eigenvalue describes a level of connectedness that ranges from zero for a disconnected node to n for a fully connected node with zero link utilization. To illustrate this point, Fig. 2 shows the eigenvalues of the Fig. 1 network as the link weights for node six approach zero. The physical meaning of the values approaching zero is that all the links are load balanced and the utilization is increasing; the link weight is inversely proportional to the link utilization.

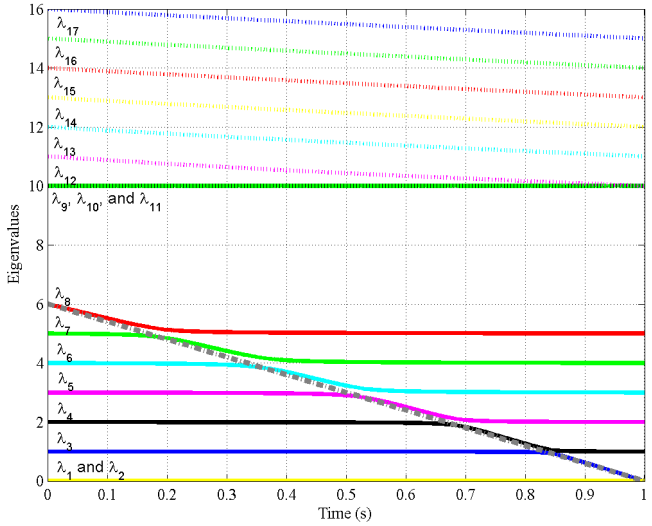

Fig. 2. Eigenvalues of the 17 node network in Fig. 1 as the links to node six go to zero

Similar to graph partitioning methods [11], large gaps between eigenvalues are indicative of distinct sets of nodes. The eigenvalues belonging to the mesh core in Fig. 1 are associated with the larger eigenvalues, and the smaller eigenvalues are associated with the access network. From this perspective, the overall SDN consists of two distinct networks, but all of which is managed by a logically centralized controller [12].

From Fig. 2, one can also see the reduction of node six's links to zero in the structure of eigenvalues three through eight. Eigenvalues nine through 17 also change due to the relationship between the access and core networks. At the end of the simulation, all of the affected eigenvalues have shifted down by one. The number of zero eigenvalues has increased from two to three because there are now three separate networks: nodes six, node seventeen and the remaining connected nodes. The slope of the line that connects all of the transition phases of each eigenvalue is

$$\frac{d\lambda}{dw} \approx -\deg(node_k) + \lambda_o \tag{6}$$

The degree of node k determines the approximate slope of each eigenvalue as it shifts from its current value to the next value down, and the starting value, λ_o, determines the y-intercept . In this case, node six has a degree of six, which means that the slope is six and the starting value is six. In Fig. 2, the dashed gray line is the plot of (6) in this case.

By tracking the eigenvalues over time, the controller can dynamically calculate the state of the network. All the eigenvalues must be considered in order to fully determine the state. The next step for the controller is to determine which node or nodes are associated with each eigenvalue. By calculating the dual basis, the controller can analytically relate the nodes in the network to their associated eigenvalue. In other words, the observer can determine which node is most influential over which eigenvalue.

III. Dual Basis of a SDN

Spectral graph theory uses the eigendecomposition of the Laplacian matrix to describe characteristics of the network. The characteristics that are described are dependent upon how the links are modeled. Our objective is to determine the state of the network by measuring the link utilization of each link and then determining the Laplacian matrix's dual basis. The results of the eigendecomposition are the robustness of the network and the reachability within the network. These characteristics are fully described by the dual basis, the reachability space and null space.

A. Double Orthogonality

The eigenvector associated with each eigenvalue creates an orthogonal basis for the network model. Because the Laplacian matrix is a symmetric matrix, a self-dual basis or simply a dual basis is always created [13]. A dual basis is defined as

$$V^T V = I \tag{7}$$

where V is an orthonormal matrix of eigenvectors, and I is the identity matrix.

The dual basis, in this case, is a result of solving for the eigenvector, v, in the equation

$$(Q - \lambda I)v = 0 \tag{8}$$

By solving (6) for all eigenvalues and eigenvectors, the matrices Λ and V can be defined and used to recreate Q. In most applications, only one eigenvector matrix is required, the right eigenvector. Transformations using the right eigenvector can be demonstrated by rearranging

$$Q = V\Lambda V^T \tag{9}$$

$$QV = V\Lambda\left(V^T V\right) = V\Lambda \tag{10}$$

$$QV = z_1 \tag{11}$$

The right eigenvector's dual is the left eigenvector which can be demonstrated by rearranging (9) to result in

$$V^T Q = \left(V^T V\right)\Lambda V^T = \Lambda V^T \tag{12}$$

$$V^T Q = z_2 \tag{13}$$

Together the left and right eigenvector matrices create a self-dual basis for the network. The dual basis provides two orthogonal bases in which the network can be defined. V transforms Q into the eigencentrality space [8], which is the space where each column vector of V is associated with one eigenvalue. V^T transforms Q into the nodal space, which is the space where each column vector of V^T is associated with a specific node in the network. These two transformations allow an analysis of the network from an eigencentrality perspective or a nodal perspective. Both of which are useful depending on the specific application of the model.

B. Eigencentrality Basis

The eigencentrality basis is the basis that defines how influential a specific node is at a given eigenvalue. Fig. 3 shows a three-dimensional representation of the eigencentrality basis. Each eigencentrality vector will be an n-dimensional vector. In this example, the eigencentrality vectors are 17-dimensional vectors; one value for each node. Plotting the first three eigencentrality vectors typically produce good visual representation of the network because they place the least connected nodes at the edge and the most connected nodes at the center [14]. Networks are typically drawn this way; the core of the network is in the center of the diagram, and the access network is at the edge. Any disconnected nodes are placed at the origin, which in Fig. 3 is denoted by the red circle.

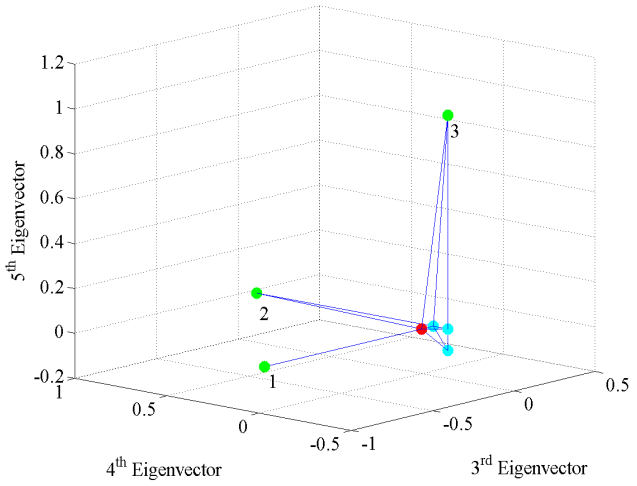

Fig. 3. Eigencentrality basis plotted using the three eigenvectors associated with the three smallest, non-zero eigenvalues with nodes one, two, and three as the least central nodes in the network

The network could be as easily plotted using the three eigenvectors associated with the three largest eigenvalues. This representation would place the most connected nodes at the edge of the graph and the least connected in the center as shown in Fig. 4. Humans have difficulty visualizing beyond three dimensions, but the controller does not have this limitation, and it can use arbitrarily large vectors that are required to describe the network's state.

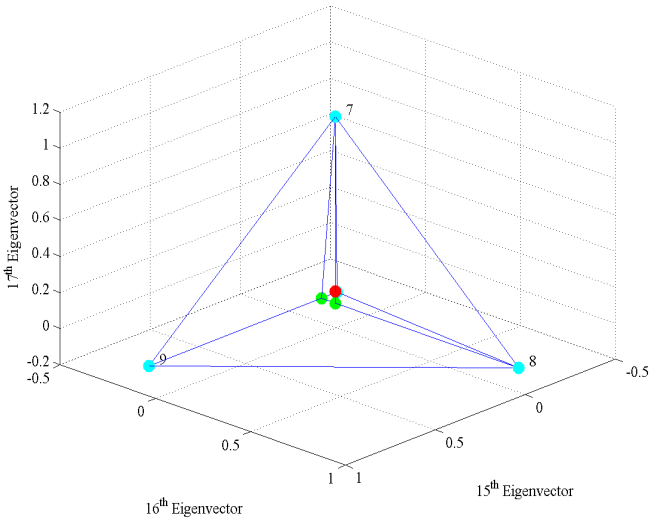

Fig. 4. Eigencentrality basis plotted using the three eigenvectors associated with the three largest eigenvalues with nodes seven, eight and nine as the most central nodes in the network

Our method decouples nodes in a network to the extent possible. Notice in Fig. 3 and Fig. 4 that there is a single node that dominates each x, y, and z-axis. The result of the eigendecomposition of the network is that nodal behavior is isolated to as small of a set of nodes as possible. This is an important feature of our approach that allows analysis of nodal behavior in isolation.

Laplacian eigencentrality is a measure of how important a node is to the network and what the impact is of its removal. The Laplacian eigencentrality is defined as

$$E_k^j = \left(v_k^j\right)^2 \tag{14}$$

or as the square of the k^{th} eigenvector's j^{th} value, [8]. This value indicates how influential each node is at each eigenvalue. Larger eigenvalues are associated with greater connectivity, and smaller eigenvalues are associated with less connectivity.

C. Nodal Basis

The nodal basis is the basis that defines how influential a specific node is across the entire eigenspectrum. This basis defines the magnitude of each value in this basis based on the Laplacian eigencentrality [15]. From this basis, a node's importance to the rest of the graph can be determined.

To see the nodal space more clearly, (2) shows that the Laplacian matrix is formed using two more fundamental matrices. The degree matrix is solely related to the nodes in the network. The adjacency matrix is solely related to the links in the matrix. When the matrix V is decomposed into

individual vectors and related back to the matrix Q, the vector form of the degree matrix is

$$D_{i,i} = Q_{i,i} = \left(v_i^1\right)^2 \lambda_1 + \left(v_i^2\right)^2 \lambda_2 + \cdots + \left(v_i^n\right)^2 \lambda_n \qquad (15)$$

where the $D_{i,i}$ is the i^{th} node in the degree matrix, which corresponds to the i^{th} value along the diagonal of Q. Eqn. (15) show that any node's degree is only a function of one eigenvector and all eigenvalues. Eqn. (15) can be simplified to

$$D_{i,i} = Q_{i,i} = \left(v_i^2\right)^2 \lambda_2 + \cdots + \left(v_i^n\right)^2 \lambda_n \qquad (16)$$

because λ_1 is always zero. Eqns. (15) and (16) demonstrate the reason for the definition of eigencentrality in (14).

The nodal basis of node six has 15 values because there are 15 non-zero eigenvalues, and the eigenspectrum is shown in Fig. 5. The nodal basis of node six clearly indicates that it has the most influence over λ_8. The magnitude response across the eigenspectrum demonstrates a node's importance at each eigenvalue. The magnitude spectrum explains why the first eigenvalue to change in the access network is λ_8. In the beginning of the simulation, it is expected that λ_8 will change first because it is most closely related to the behavior of node six. This is an example of the decoupling effect due to the eigendecomposition. Node six's links only affect the eigenvalues of the nodes to which it is directly connected. All other eigenvalues remain constant until node six's nodal basis changes and it has influence over a different eigenvalue, which occurs at approximately 0.2 seconds.

Our novel method of state determination decouples nodes in a network to the extent possible. Node six is clearly decoupled from all other nodes. The behavior of node six does not directly affect any other access network node. The reduction is seen in the core network, but this is to be expected because the links being reduced are connected to the core network.

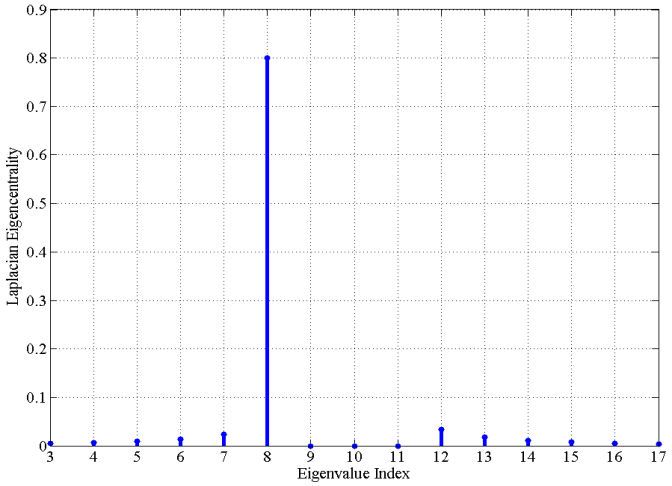

Fig. 5. Eigenspectrum of node six at time zero of the simulation

All of the nodes in the example network have similar eigenspectrum to the one shown in Fig. 5. The shape of the eigenspectrum is unique for each node due to the requirement that each vector in the basis is orthogonal to all others.

D. Null and Reachability Space

There will always be one or more zero eigenvalues of the Laplacian. The zero eigenvalues define the null space of the Laplacian. The null space is mathematically and physically interpreted as a part of the solution space or network that is unreachable. The reachability space is defined as the space in which there is guaranteed to be a route from all nodes to all other nodes. The reachability space is fully defined by the dual basis of V and V^T after the null space has been removed.

The number of non-zero eigenvalues and the size of the reachability space is equal to $\text{rank}(Q)$. The size of the null space is equal to $n - \text{rank}(Q)$. The size of the null space determines the length of the vectors in the nodal space; the nodal space eigenvectors will have dimensionality equal to the $\text{rank}(Q)$. The eigencentrality basis vectors will always have a length equal to n [8].

A goal of the SDN controller should be to ensure that nodes do not become disconnected from the network and enter the null space. As a function of time, the null space will grow and shrink. Nodes within the null space may be able to reach each other, but they are not useful because they are not able to connect to the larger SDN.

IV. Modeling and Analysis Simulations

In the previous sections, we have laid out the analytical groundwork to describe the application of the dual basis to determine the state of the network. The simulations we have developed demonstrate that the state of the network is a dynamic entity similar to that of the state vector in control theory. In the dual basis, the state is not a single vector, but it is actually a basis in a multi-dimensional space.

Fig. 1 shows the network we chose to simulate. The network is a model of a core network that is a fully connected mesh and an access network that has a range of links from one to six. The network was designed this way in order to demonstrate the utility of our approach. In order to show the dynamic behavior of our approach, the links of three nodes were reduced to zero in a similar fashion. From zero to one second, all of the links of node six are reduced to zero at a constant rate. From one to two seconds, all of the links of node five are reduced to zero at a constant rate. From two to three seconds, all of the links of node four are reduced to zero a constant rate.

A. Eigenvalue Results as Three Nodes are Disconnected

Fig. 6 shows the change in eigenvalues as the three nodes are consecutively dropped. The first significant result observed in Fig. 6 is the behavior of the core network's eigenvalues with the removal of three non-core nodes. The largest eigenvalues all have a negative slope until 10 is reached, which is the smallest degree of the nodes in the core mesh network. The reduction of the largest eigenvalue is because of (5) and is bounded on the lower end by minimum degree of the mesh. The maximum eigenvalue is bounded by the total number of connected nodes. In this case, the total starts at 16 and is reduced to 13 at the end of the simulation. The lower bound is

10 because the minimum degree of the mesh network is 10. Once an eigenvalue reaches 10, it becomes a repeated eigenvalue.

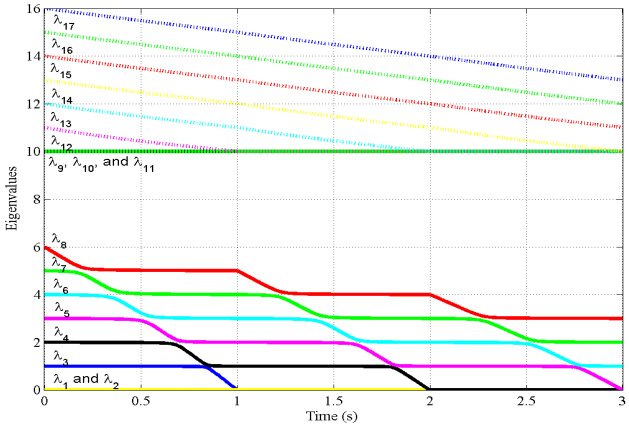

Fig. 6. Eigenvalues as all the links for nodes six, five and four go to zero

The second major result is the behavior of the access network's eigenvalues, which demonstrate the removal of the three nodes. Eqn. (6) holds for all three transitions. The slope of the first transition is -6, the second is -5, and the third is -4. The structure of the transitions between eigenvalues is the reason we propose using the full eigenspectrum of a network as opposed to simply using the algebraic connectivity. The full spectrum provides the controller with information about how the values are changing relative to the others. The link utilizations of node six are more than 80% prior to seeing any change in the algebraic connectivity, λ_3. By looking at all of the values, the controller can determine how the network is being loaded.

Because of use of (1), smaller eigenvalues are associated with greater traffic loads in the network. The eigenspectrum characterizes how the network is loaded and how well it is managing the traffic. As node six becomes disconnected from the network, the algebraic connectivity begins to change. This is an indication that the network is receiving more traffic than it can handle in at least one location. This pattern is repeated for the next two nodes. The controller can use this information to determine that there is too much traffic in an area of the network. The controller must next use the dual basis to determine which node or nodes in the network are driving the algebraic connectivity down.

B. Eigenvector Results as Three Nodes are Disconnected

Fig. 7 and Fig. 8 are plots of eigenvector components as a function of time. Fig. 7 are the components associated with eigenvectors three, four and five. Fig. 8 are the components associated with eigenvectors 15, 16, and 17. The eigencentrality values are the components of each vector. Eigenvectors one and two are in the null space of the dual basis and therefore, they do not provide any useful information about how the network is performing.

The three smallest eigenvector components depict the transitions of nodes between the separate vectors within each basis. The red line in Fig. 7 is node six's component value. The transition just after 0.5 seconds in Fig. 7(a) coincides with the transition in Fig. 6 at the same time. As node six becomes more disconnected its eigencentrality magnitude changes in each vector. Notice how it peaks in the fifth eigenvector and then in the fourth and finally in the third. At the end of one second, node six enters the null space and its eigencentrality value jumps to one while all the others are zero. The physical meaning of a one in the vector and all remaining values equal to zero is that the node has become disconnected and can no longer reach any other node in the network. This same behavior is demonstrated by node five (green) and similarly node four (cyan).

The 15th, 16th, and 17th eigenvectors provide insight into the behavior of the core network. As shown in Fig. 8, the eigencentrality values of the nodes seven, eight, and nine do not change as nodes four, five and six are removed. Fig. 6 predicted this behavior. There are no transitions between eigenvalues and therefore, there is no change in the eigencentrality metrics of the three most connected nodes. The eigenvectors of the core network are isolated from the eigenvectors of the access network.

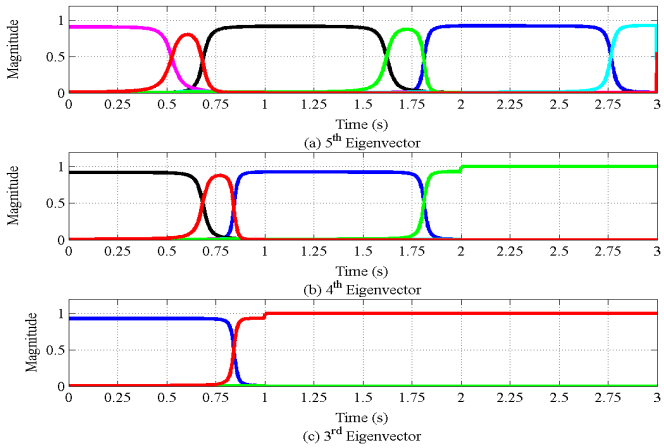

Fig. 7. Plot of the third, fourth and fifth eigencentrality components as all the links to nodes six, five and four go to zero. Node one is blue. Node two is black. Node three is magenta. Node four is cyan. Node five is green. Node six is red.

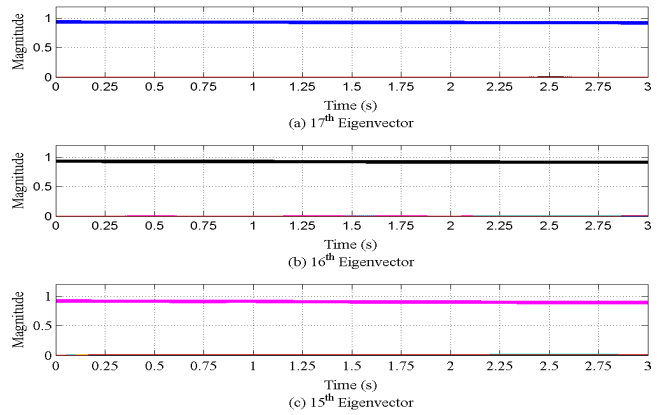

Fig. 8. Plot of the 15th, 16th, and 17th eigencentrality components as all the links to nodes six, five and four go to zero. Node 14 is magenta. Node 15 is black. Node 16 is blue.

Putting Fig. 3 into motion using the values as shown in Fig. 7 provides a useful way to look at the basis. Fig. 9 shows this movement in two-dimensions. To start, node one and node two are separated by 90°. This is due to the fact that the eigendecomposition ensures that the nodal vectors will be orthogonal to each other. Next notice that as node six's links are reduced to zero, the vector that points to node six first translates in the negative direction down the y-axis and then rotates over to the x-axis. This translation and rotation are also demonstrated as the other two nodes are removed from the network. The state of each node can be described in terms of a magnitude and angle about the center of the basis [16]. It is more difficult to visualize these rotations in multi-dimensional space, but these rotations can be computed in the multi-dimensional space that is described by the dual basis. The angles provide the controller with information about how nodes are switching between eigenvalues. These switches are described by the rate of change of the magnitude and angle of the node to any vector in the eigenspace.

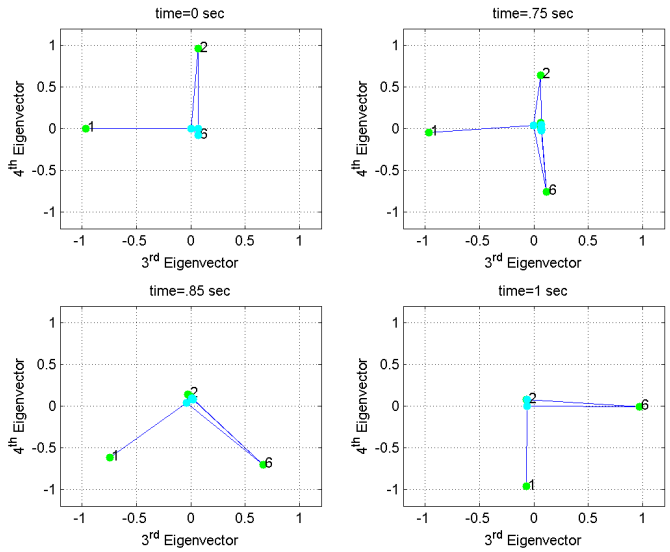

Fig. 9. Two-dimensional plot of the network as node six replaces nodes one and two as the primary node in each eigenvalue

V. Conclusions

Our novel approach shows that the SDN observer is able to dynamically determine the state of a SDN. We have demonstrated our approach both and in simulation. The proposed SDN observer determines which areas of the network are most utilized and which areas are least utilized. This information is passed to the controller, which can adapt flows to reduce congestion and load balance the network. In a network where the maximum bandwidth varies, the weighted graph can be dynamically updated to allow the controller to have greater understanding of the true network dynamics.

We have shown that the eigenvalues are an effective method to determine the state of the network and in combination with the eigenvectors, the observer can calculate the complete representation of the network. To close the control loop, the observer passes this information to update the controller on the current network state. The controller can then use optimal control methods to improve the performance of the network.

Our future research will focus on how to use this information to close the control loop in order to increase the performance of the network. We will explore how our state representation can be used by the controller to determine the difference between the ideal state and the current state. Once the difference is calculated, the controller must update flows to decrease that difference. The control algorithm must take into account that the controller may not have control over how much traffic is associated with each flow.

References

[1] S. Jain, et al., "B4: Experience with a globally-deployed software defined WAN," in *ACM SIGCOMM 2013 Conference on SIGCOMM*, Hong Kong, 2013.

[2] N. Nise, *Control Systems Engineering*. Menlo Park, CA: Addison-Wesley Publishing Compnay, 1995.

[3] S. Agarwal, M. Kodialam and T. Lakshman, "Traffic Engineering in Software Defined Networks," in *Proceedings IEEE INFOCOM*, Turin, Italy, 2013.

[4] Y. Kim and M. Mesbahi, "On Maximizing the Second Smallest Eignevalue of a State-Dependent Graph Laplacian," *IEEE Transactions on Automatic Control,* vol. 51, no. 1, 2006.

[5] HICCS paper

[6] A. Sydney, C. Scoglio and D. Gruenbacher, "The impact of optimizing algebraic connectivity in hierarchical communication networks for transmission operations in smart grids," in *Innovative Smart Grid Technologies (ISGT), 2013 IEEE PES*, Washington, DC, 2013

[7] R. Olfati-Saber, J. A. Fax and R. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceeding of IEEE,* vol. 95, no. 1, pp. 215-233, 2007.

[8] P. V. Mieghem, "Graph eigenvectors, fundamental weights and centrality metrics for nodes in networks," arXiv preprint arXiv:1401.4580 [math.SP] (2014).

[9] P. V. Mieghem, *Graph Spectra for Complex Networks.* New York: Cambridge University Press, 2011.

[10] M. Fiedler, "Algebraic Connectivity of Graphs," *Czechoslovak Mathematical Journal,* vol. 23, no. 98, pp. 298-305, 1973.

[11] M. E. J. Newman, "Finding Community Structure in Networks Using the Eigenvectors of Matrices," *Physical Review E,* vol. 74, no. 3, pp. 036104-1 - 036104-19, 2006.

[12] D. Levin, A. Wundsam, A. Heller, B. Handigol and A. Feldmann, "Logically Centralized? State Distribution Trade-offs in Software Defined Networks," in *HotSDN*, Helsinki, Finland, 2012.

[13] G. Strang, *Linear Algebra and Its Applications.* Belmont, CA: Brooks/Cole, Cengage Learning, 2006.

[14] D. Spielman, "Spectral Graph Theory and its Applications," in *48th Annual IEEE Symposium on Foundations of Computer Science*, 2007.

[15] U. von Luxburg, M. Belkin, and O. Bousquet, "Consistency of spectral clustering," The Annals of Statistics, vol. 36, no. 2, pp. 555-586, 2008.

[16] D. Cvetkovic, P. Rowlinson, S. Simic, *Eigenspace of Graphs*. New York, NY: Cambridge University Press, 1997.