Continuum of All-Pair Shortest-Path to All-Path via Random Walk

Golshan Golnari and Daniel Boley

April 19, 2013

Continuum of All-Pair Shortest-Path to All-Path via Random Walk

Golshan Golnari
Department of Electrical
and Computer Engineering
University of Minnesota
Minnesota, Minneapolis 55455
Email: golnari@cs.umn.edu

Daniel Boley
Department of Computer Science
and Engineering
University of Minnesota
Minnesota, Minneapolis 55455
Email: boley@cs.umn.edu

***Abstract*—A method is proposed to compute the continuum of paths, from shortest paths to all random paths between all pairs of nodes at once in a unified way. The analysis is based on treating the network as a random walk with an additional absorbing state named evaporating state reachable with non-zero probability from any state (so called "evaporating random walk"). The probability of avoiding absorption is tuned by a single parameter varying between 0 and 1, with lower values favoring shorter paths. A computational example is used to illustrate the method.**

I. INTRODUCTION

A. *GENERAL INTRODUCTION*

Recently many sciences confront large networks and need to do network analysis. Routing in wireless networks, protein interactions, transportation problems and social networks are some of the examples that use methods to analyze large networks. Shortest-path is an important primitive in this regard. The total of the shortest-path distances of a node from the other nodes in the network is a measure of its centrality based on closeness for purposes like information spread [1]. For social networks, Freeman [2] introduced a betweenness centrality stating that a node with high probability of occurring on the shortest-path between two randomly chosen nodes, has a high betweenness. Both centrality measure and betweenness are also used for active inference in collective classification where the objective is to maximize classification performance while minimizing the amount of data that should be labeled in advance [3]. Shortest-path also has many direct applications. shortest-path routing is most commonly used in modern IP-based data networks; Open-Shortest-Path-First protocol (OSPF) is one of the adaptive-routing algorithms that dominate the Internet [4]. Also, for finding directions between physical locations on web mapping websites like Google map, fast specialized shortest-path algorithms are applied [5]. Operations research, plant and facility layout, robotics, transportation, and VLSI design are other examples of fields where shortest-path algorithms are widely used [6].

Random walk based metrics like Hitting Time and Commute Time [7] are other important metrics used for network analysis. These metrics are defined for a random walk on a network and have the concept of average distance between nodes. The random walk randomly chooses its path and wanders around until it hits the destination for the first time. Thus, it is also called "all-path"in contrast with "shortest-path" and Hitting Time is the average length of this "all-path". Doyle and Snell [9] proved that the effective resistance in electrical networks has a close relation to the Commute Time in undirected networks. Newman [1] introduced a centrality measure stating that the node with biggest average number of passages by a random walk is the center of the network. Chen, Liu, and Tang [10] presented a clustering algorithm via random walk Hitting Time on directed graphs. Hitting Time and Commute Time are also used in figuring out the protein interactions in biological systems [11] and also applied in recommendation systems [12].

In many applications like routing in computer networks, using "all-path" does not make sense and almost using "shortest-paths" is desired. However, in some situations, pure shortest-path is not satisfactory and some other alternative paths are also required. Some of these reasons are congestion reduction in data networks, avoiding complete predictability of the routing strategy, and increasing the robustness of the network. In wireless networks, using only the "shortest" path is not reliable, because the channels are not stable and their characteristics vary over the times. Hence effective strategies have been proposed which do not rely only on the "shortest" path and use multiple paths [13][14][15][16]. In this paper we present a novel method which gives the path lengths and the details of the paths in a continuum from shortest-path to all-path by varying a parameter named α. For this purpose, we first derive formulations to calculate the conditional path lengths reaching one of the absorbing states in a network with two absorbing states. Then we model an evaporating network with a network having two absorbing states. Next, we develop a method which calculates the length of paths among the nodes in a continuum from the shortest-path to all-path using the corresponding evaporating network. Also we derive a matrix for any choice of destination (or a 3 order tensor for all choices of destinations) which gives the details of the paths. Especially for the shortest-path case, these details include the information about the nodes occurring on the shortest-paths, the number of shortest-paths passing from each node, the number of common shortest-paths on which two nodes occur, a lower bound for the number of shortest-paths between a pair of nodes, and an upper bound for the number of disjoint shortest-paths between a pair of nodes. Considering our method just in its shortest-path case, its complexity is comparable to the popular shortest-path algorithms, discussed in the Related Work subsection, whose complexity is $O(n^3)$ for deriving the shortest-paths for all pairs of nodes.

B. RELATED WORK

There is a large literature dedicated to deriving the shortest-paths. Dijkstra's algorithm is a graph search algorithm to compute single source shortest-paths in non-negative weighted graphs. For a graph with n vertices and m edges, its complexity is in the order of $O(n^2)$ in general and can be reduced to $O(m + nlogn)$ for sparse graphs [17] [18]. Bellman-Ford algorithm is slower in compared to Dijkstra's algorithm, however it solves the single source problem for weighted but maybe negative edges and its complexity in worst case is $O(mn)$ [19]. Floyd algorithm [20] finds the length of the shortest-path between any pair of the nodes in the network but it does not give any detail about the paths themselves and its complexity is $O(n^3)$. There are also some other algorithms that are the extensions of the above algorithms [21] [22].

Some work has been done on Markov chains and modeling random walks and giving formulation for calculating random walk based metrics like Hitting Time [23][24]. The complexity of calculating the Hitting Time is $O(n^3)$.

A combination of shortest-path and all-path has been addressed in some recent works. Li, Zhang, and Boley [25] show that shortest-path routing can be interpreted as L_1 flow optimization and all-path routing as L_2 flow optimization. Then a theoretical framework based on mixed (weighted) L_1/L_2-norm optimization is suggested. It uses a trade-off parameter θ that makes the optimization solution span from the one extreme (shortest-path) to the other extreme (all-path). This framework is theoretical and not suitable for practical purposes since its complexity is very high. For just one pair of source and destination its complexity is $O(n^5)$ in the worst case. It should also be mentioned that this method is for undirected graphs only. Mantrach et al. [26] consider all possible paths of the network in a set and introduce an optimization problem on the probability of taking these paths to minimize the total expected cost between all pairs of nodes. The constraint of this optimization is the total relative entropy in the network to be below some threshold and the solution is a Boltzmann distribution on the set of paths. This distribution pattern is for a function of a parameter θ that makes the optimization solution span from the one extreme i.e. shortest-path to the other extreme i.e. all-path. The main drawback of this work is that they consider the paths to be independent of each other which is a limiting assumption. For a network with n nodes and m edges where $m > n$, there are only $m - n$ degree of freedom for assigning the probabilities to the edges while they consider the degree of freedom to be as big as the number of all possible paths in the network. Tahbaz-Salehi and Jadbabaie [27] present a one-parameter family of consensus algorithms which recovers the well-known Bellman-Ford iteration and the Hitting Time iterations as two extremes of the parameter values. The connection between this family of algorithms and Markov decision processes has been postponed to future work by the authors. The output of their work is the length of the paths and says nothing about the details of the paths.

C. CONTRIBUTION AND PAPER ORGANIZATION

The main contribution of this paper is presenting a novel method to derive the continuum from shortest-path to all-path between all pairs of nodes in a network. The method identifies the nodes involved in the shorter paths and the average path length for all pair of source-destination as the parameter α varies. The complexity of this method is $O(n^3)$ for each parameter value of α and for all pair of nodes.

The rest of this paper is organized as follows. A background of the theory of our work and a notation guideline used in this paper come in Section 2. Then we dedicate Section 3 to present formula to calculate average path length conditioned on reaching a specific destination. In Section 4 we model the shortest-path problem and the average path length in a continuum as the Hitting Time in evaporating networks. Section 5 is about the inferred path information using the derived formulations in Section 4. In Section 6 we explain how to generalize the formulations for any choice of the destination. Computational aspects and simulations come in Section 7. We conclude our work and propose some future work in Section 8.

II. BACKGROUND AND NOTATION

A directed graph with $n + 1$ number of nodes, denoted by $G = (V, E)$, has $V = 1, 2, ..., n + 1$ as its set of nodes and E as the set of its edges which are directed $(i \to j)$. The digraph G is represented by its $(n + 1) \times (n + 1)$ adjacency matrix A, where $A(i, j) = 1$ is showing the edge $(i \to j)$ and $A(i, j) = 0$ if $(i \to j) \notin E$. A directed graph G is called strongly connected if there is a path from i to j for any pair of nodes i, j.

A random walk over a graph can be modeled by a Markov chain with probability transition matrix $P_{org} = D^{-1}A$, where D = Diag(d) = Diag(Ae) is the diagonal matrix of out-degrees (the number of edges attached to a node which are pointing out) and e denotes the vector of all ones. In this paper, the words "node" and "state" are used interchangeably. There are several defined metrics for a random walk over a graph which we use in this paper. One of them is the definition of"Hitting Time" from node i to node j, which is the average number of steps taken by the random walk when starting from node i to reach j for the first time. That is why it is also called average path length in this paper. The other one is the "average number of passages through nodes" which is defined for an ordered triple (i, k, j) giving the average number of passages of a random walk through node k starting from node i before first arrival at node j.

Note that a destination in a network can be modeled as an absorbing state since hitting it, the journey of packet/random walke finishes. Hence for a network with $n+1$ nodes and the original transition matrix P_{org} if we assume node $n+1$ as the destination, from mathematical aspect of view, the new probability transition matrix P is the same as the original transition matrix P_{org} except for its $(n+1)$-row, which become e_{n+1}^T. Since the out-links of destinations do not appear in the calculations, this replacement of the rows does not change anything in the result. Hence, we use the probability transition matrix of corresponding absorbing Markov chain model P for both cases, unless otherwise specified and we want to address the exact P_{org}. In this paper, words "destination" and "absorbing state" are used interchangeably.

The "average number of passages through nodes" for node $n+1$ as the destination denoted by N can be obtained by turning $n+1$ into an absorbing state, and counting how many times one passes a given node k before absorption. Then the average number of times a random walk would pass node k when starting at node i (before absorption by node $n+1$) is the i,k-th entry of matrix N. This matrix is given by [8]:

$$\begin{aligned} N &= (I - P(-(n+1)))^{-1} \\ &= I + P(-(n+1)) + (P(-(n+1)))^2 + ..., \quad (1) \end{aligned}$$

where I is the identity matrix and $P_{-(n+1)}$ is the matrix obtained by suppressing row and column $n+1$ of the original probability transition matrix P. In this equation, $(I - P_{-(n+1)})$ should be non-singular which is true whenever node $n+1$ is reachable from any other node, something we assume throughout this paper.

The "Hitting Time" for node $n+1$ as the destination denoted by h is the sum of the average number of passages through all possible intermediate nodes:

$$h = \sum_k N(:,k) = Ne, \quad (2)$$

where $k \in V \backslash \{n+1\}$.

Beside being analytically provable, equation (2) is conceptually understandable. The average distance between node i and $n+1$ or equivalently the average number of steps/links should be taken/passed to reach node $n+1$ from node i, is the summation of average number of passing the nodes on the way (including the starting node i at the first step).

For the convenience of readers, we bring the notations used in this paper in the following list:

List of notations:

The following summarizes our notation measuring random walks over an $n+1$-state Markov chain. Note that throughout this paper, node $n+1$ is always an absorbing state. Hence, if there is no subscript in the notations it means that the only absorbing state is node $n+1$, and one subscript shows that there is also another absorbing node beside node $n+1$ and the metric belongs to the case that random walker reaches this absorbing node before node $n+1$. If the metric is for the case of reaching *either* of the absorbing nodes (node $n+1$ and one other absorbing node), both absorbing nodes come as an subscript of the metric.

- N: an $n \times n$ matrix whose i,k-th entry is the average number of passages through node k, starting from node i, before reaching node $n+1$.
- $N_{\{j,n+1\}}$: an $(n-1)\times(n-1)$ matrix whose i,k-th entry is the average number of passages through node k, starting from node i, before reaching *either* node j or $n+1$.
- N_j: an $(n-1)\times(n-1)$ matrix whose i,k-th entry is the number of passages through node k, starting from node i, averaged only over paths ending in node j (paths are partitioned into two groups, the paths ending in node n and the ones ending in node $n+1$).
- e is the vector of all ones, e_k is the k-th coordinate unit vector.
- h: the n-length vector of Hitting Time from any other node to node $n+1$.
- $h_{\{j,n+1\}}$: the $n-1$-length vector of Hitting Time from any other node to *either* node j or $n+1$.
- h_j: the $n-1$-length vector of Hitting Time from any other node to node j conditioned on that this destination is reached first (node j is reached before node $n+1$).
- l_j: the vector of shortest-path distances from all the other nodes to node j.
- b_j: the $n-1$-length vector of the absorption probabilities by node j before node $n+1$. Each entry corresponds to each starting point i, $i \in V\backslash\{j, n+1\}$.

In the sequel we will add a state number $n+1$ to a network with n states to model an evaporating network, identical to the original n-state network but with a finite probability $1-\alpha$ that one could evaporate (i.e., be absorbed by the new state $n+1$). All the above quantities will have a superscript (α) when they refer to such a network.

In general, if v is a vector, the notation $v(-n)$ denotes the vector with the n-th entry deleted. If M is a matrix, the notation $M(-k,-l)$ denotes the matrix with the k-th row and l-th column deleted. If $k = l$, we also use the notation $M(-k)$ for square matrices.

III. HITTING TIME CONDITIONED ON REACHING A SPECIFIC DESTINATION

Existing theory [9] gives the Hitting Time in a Markov chain with m absorbing states before absorption by *either* of the absorbing states. In this section we introduce a new metric which calculates the Hitting Time in the same Markov chain, but for a *specific* absorbing state conditioned on hitting this absorbing state first and avoiding the other absorbing states. In other words, we partition the paths into m groups each of them ending in one of m absorbing states. We present a formulation to calculate the average path length of each of these m groups conditioned on reaching the corresponding absorbing state.

In the following, we first bring the existing theory, then we present the derivation of new mentioned metric as a Theorem which has been specified for two absorbing states ($m = 2$) and is used to calculate the shortest path length in later sections.

Consider a network with $n+1$ nodes and two destinations. We renumber the nodes so that the destinations are numbered last, having indices n and $n+1$. The new probability transition matrix P is the same as the original transition matrix P_{org} except for its n-row and $(n+1)$-row, which become e_n^T and e_{n+1}^T respectively and it has the following form:

$$P = \begin{bmatrix} Q & R \\ 0 & I \end{bmatrix}_{(n+1)\times(n+1)}, \tag{3}$$

where I is a 2×2 identity matrix.

The average Number of passages through nodes before absorption by *either* of absorbing states is the following $(n-1)\times(n-1)$ matrix [9]:

$$N_{\{n,n+1\}} = (I-Q)^{-1}. \tag{4}$$

The vector h is defined as the expected number of steps taken by the random walk before absorption by *either* of absorbing states [9]:

$$h_{\{n,n+1\}} = N_{\{n,n+1\}}e. \tag{5}$$

The absorption probability matrix B is an $(n-1)\times 2$ matrix [9]:

$$B_{\{n,n+1\}} = N_{\{n,n+1\}}R \tag{6}$$

The formulation above simply says that to obtain the probability of getting absorbed by a given absorbing state, we add up the probabilities of going there from all the non-absorbing states, weighted by the number of times we expect to be in those (non-absorbing) states [9]. The two columns of B are complementary to each other and add up to **1**. We denote the first column which corresponds to node n by b_n and it is obtained from equation (6):

$$b_n = N_{\{n,n+1\}}R(:,1) \tag{7}$$

Using this introduction, we establish the following theorem.

Theorem 1: Consider a network with $n+1$ nodes where nodes n and $n+1$ are two destinations in this network. The "Hitting Time" and "number of passages through intermediate nodes", starting from any given node i and conditioned on reaching node n before node $n+1$ can be derived from the following formula respectively:

$$h_n = (N_{\{n,n+1\}}b_n) \oslash b_n, \tag{8}$$

$$N_n = Diag(b_n)^{-1}N_{\{n,n+1\}}Diag(b_n) \tag{9}$$

To calculate the same metrics for reaching node $n+1$ before node n, it is sufficient to replace the first column of $B_{\{n,n+1\}}$ denoted by b_n with the second column of this matrix in two equations above.

Proof: We add an imaginary absorbing node named super absorbing node (and numbered as node $n+2$). All the transition probabilities remain the same, except that the original destination nodes n and $n+1$ become transient nodes such that when reached, there is a probability of 1 to move to the node $n+2$ on the next step. The new probability transition matrix denoted by S, has dimension of $(n+2)\times(n+2)$, and has the following form:

$$S = \begin{bmatrix} Q & R & 0 \\ 0 & 0 & e \end{bmatrix} \tag{10}$$

where e is a vector of all ones with dimension 3×1. The leading $n+1$ rows and columns of S have the following form:

$$S(-(n+2)) = \begin{bmatrix} Q & R \\ 0 & 0 \end{bmatrix}, \tag{11}$$

with two rows of zeros. We define matrix G as below:

$$G = 1S + 2S^2 + 3S^3 + 4S^4 + ... \tag{12}$$

The special form of matrix S helps us to write the following equation from equation (11):

$$G(-(n+2)) = 1S(-(n+2)) + 2(S(-(n+2)))^2 + ... \tag{13}$$

The $(n-1)\times 1$ vector $z = G(1:n-1,n)$ is the desired part for us. The i,n-th entry of S^k (for $i = 1,\ldots,n-1$) is the probability that one will reach node n starting from node i in exactly k steps. By construction, one can pass node n only once. The elementwise division of $S^k(1:n-1,n)$ by b_n yields the conditional probability of reaching node n, *given* that one reaches destination n as opposed to the other destination $n+1$. Therefore dividing the vector $z = G(1:n-1,n)$ elementwise by b_n yields the average path length (Hitting Time) from any node i to node n conditioned on hitting node n before node $n+1$.

$$h_n = z \oslash b_n, \tag{14}$$

where $\oslash$ denotes elementwise division between two matrices or vectors. Vector h_n has $n-1$ rows which corresponds to $n-1$ non-absorbing states.

Writing vector z in terms of submatrices Q and R, we have:

$$\begin{aligned} z &= R(:,1) + 2QR(:,1) + 3Q^2R(:,1) + 4Q^3R(:,1) + ... \\ &= (I-Q)^{-2}R(:,1) \\ &= N^2{}_{\{n,n+1\}}R(:,1) \\ &= N_{\{n,n+1\}}b_n \end{aligned} \tag{15}$$

Substituting equation (15) in equation (14), the conditional average number of steps required to start from node i ($i \in \{1,...,n-1\}$) and end in node n is obtained:

$$h_n(i) = \frac{\sum_{k=1}^{n-1} N_{\{n,n+1\}}(i,k)b_n(k)}{b_n(i)}, \tag{16}$$

and in the matrix form:

$$h_n = (N_{\{n,n+1\}}b_n) \oslash b_n \tag{17}$$

As explained before, matrix $N_{\{n,n+1\}}$ is the number of passages through the non-absorbing nodes before absorption by *either* of the two absorbing states. The random walk starting from node i, conditioned on ending in node n before node $n+1$, passes node k for $N_{\{n,n+1\}}(i,k)b_n(k)/b_n(i)$ times on average. Thus we define matrix N_n as the average number of passages through the non-absorbing states before reaching node n for the first time, given that we hit this node before node $n+1$:

$$N_n = Diag(b_n)^{-1} N_{\{n,n+1\}} Diag(b_n) \tag{18}$$

■

To interpret how vector h_n has the concept of Hitting Time, consider its i-th entry:

$$\begin{aligned} h_n(i) &= z(i)/b_n(i) \\ &= G(i,n)/b_n(i) \\ &= \frac{1S(i,n)}{b_n(i)} + \frac{2S^2(i,n)}{b_n(i)} \\ &+ \frac{3S^3(i,n)}{b_n(i)} + \frac{4S^4(i,n)}{b_n(i)} + ... \end{aligned} \tag{19}$$

The intuition of this formula is that the conditional average number of steps needed to reach node n is "the conditional probability of getting there in one step" times 1 plus "the conditional probability of getting there in the second step" times 2 plus "the conditional probability of getting there in the third step" times 3 plus .... To make it more clear, we can write equation (12) as the following form:

$$\begin{aligned} h_n(i) &= z(i)/b_n(i) \\ &= \frac{R(i,1) + 2[QR](i,1) + 3[Q^2R](i,1) + ...}{[N_{\{n,n+1\}}R](i,1)} \\ &= \frac{R(i,1) + 2[QR](i,1) + 3[Q^2R](i,1) + ...}{R(i,1) + [QR](i,1) + [Q^2R](i,1) + ...} \\ &= \frac{\sum_{k=1}^{\infty} k\Pr_k(i,n)}{\sum_{k=1}^{\infty} \Pr_k(i,n)}, \end{aligned} \tag{20}$$

where k is the path length from node i to node n and $\Pr_k(i,n) = [Q^{k-1}R](i,1)$ is the probability of being in node n in k-th step starting from node i which equals to the summation of k-length paths' probabilities. As this formulation shows, $h_n(i)$ is the average path length from node i to node n conditioned on reaching node n before hitting node $n+1$.

Using equation (8) and (9) we have the following equation:

$$\begin{aligned} N_n e &= (Diag(b_n)^{-1} N_{\{n,n+1\}} Diag(b_n))e \\ &= ((N_{\{n,n+1\}} Diag(b_n)) \oslash (b_n e^T))e \\ &= (N_{\{n,n+1\}} b_n) \oslash b_n \\ &= h_n \end{aligned} \tag{21}$$

This result is similar to one in equation (2) which was for a network with one absorbing state. We can see that we have the same relation between Hitting Time and average number of passages through nodes for a network with two absorbing state for reaching one specific destination before the other one.

IV. MODELING THE SHORTEST-PATH PROBLEM AS THE CONDITIONED AVERAGE PATH LENGTH IN THE CORRESPONDING EVAPORATING NETWORK

In this section we present the shortest-path problem as a new model which uses the evaporating network concept to find the shortest path lengths in the network. We construct an evaporating network from an n-node recurring walk by adding an absorbing node $n+1$ with a uniform non-zero probability $(1-\alpha)$ of transitioning from any existing node to node $n+1$. We also show that by varying the evaporating parameter α we obtain a continuum from the shortest-path to all-path. As before, to simplify exposition we derive formulas for paths leading to node n, by turning node n into an absorbing state. Hence the transition matrix for the n-node network is

$$P = \begin{bmatrix} U & V \\ \underline{0} & 1 \end{bmatrix}_{n \times n}. \tag{22}$$

Definition 1: The evaporating network corresponding to (22) is the network represented by the $(n+1) \times (n+1)$ transition matrix

$$P_{evp} = \left[\begin{array}{c|cc} \alpha U & \alpha V & (1-\alpha)e \\ \hline \underline{0} & 1 & 0 \\ \underline{0} & 0 & 1 \end{array}\right] = \left[\begin{array}{c|c} Q & R \\ \hline 0 & I \end{array}\right], \tag{23}$$

where we identify the partitions with the partitions of (3).

Theorem 2: Given a recurring random walk derived from a strongly connected network with n nodes and original probability transition matrix P_{org}:
As $\alpha \to 0$, $h_n^{(\alpha)}$ approaches shortest-path lengths l_n,
As $\alpha \to 1$, $h_n^{(\alpha)}$ approaches the regular Hitting Time for network with node n as its only absorbing state.
This provides a continuum between shortest-path and all-path as α ranges between 0 and 1.

Proof: With the identification in Definition 1, the average hitting time $h_n^{(\alpha)}$ (20) becomes a function of α:

$$\begin{aligned} h_n^{(\alpha)}(i) &= \frac{\sum_{k=1}^{\infty} k\Pr_k(i,n)}{\sum_{k=1}^{\infty} \Pr_k(i,n)} \\ &= \frac{\sum_{k=1}^{\infty} k\alpha^k [U^{k-1}V](i,n)}{\sum_{k=1}^{\infty} \alpha^k [U^{k-1}V](i,n)}, \end{aligned} \tag{24}$$

Formula (24) gives the average path length for a given α. Suppose $k = L$ is the length of a shortest path from i to n. Then the first nonzero term in both the numerator and denominator is the term where $k = L$. As $\alpha \to 0$, these terms dominate all subsequent terms, so the expression converges to L as $\alpha \to 0$.
For $\alpha = 1$, there is no evaporation and the network is a network with one absorbing state (node n) and the simple random walk formulation like the one in equation (1) and (2) for networks with one absorbing state can be applied. In this case, node $n+1$ is an isolated node with a self loop and row and column $n+1$ are a vector of all zero (except the $P_{evp}(n+1, n+1)$ entry) and so this row and column can be omitted to obtain the Probability transition matrix of the network. ■

As was mentioned in the related work section, there are many algorithms of solving the shortest-path problem [17][18][19][20][21][22]. But here we showed this as a special case in a continuum of paths which presents a new way of shortest-path problem solution.

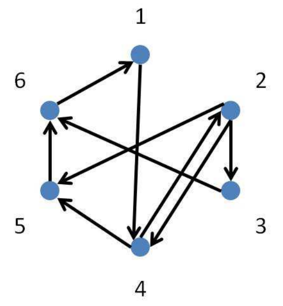


Fig. 1. A toy example

The intuition of the Theorem 2 is that when α gets very small, the evaporation of the network is very high and only the random walks that traverses the *short paths*, can survive. This fact also affects the matrix $N_n^{(\alpha\to 0)}$ and makes its entries corresponding to nodes not on a shortest path equal to zero. Non-zero entries of row i ($i \in \{1,...,n-1\}$), are the nodes occurring on the shortest-paths from node i to the desired destination (node n). There are also some other information which can be inferred from this matrix and the next section clarifies this better with a toy example. The superscript α shows that the formulation belongs to the corresponding evaporating network and uses P_{evp} as its basis.

By increasing α a little larger than zero, there would be also some other paths (which are still short but they are not the shortest path) that have non-ignoreable amount in equation (24) and lead the packet/random walker to the destination. Larger α will result to longer kind of these paths which shows this interesting finding that tuning the α can give a continuum from the shortest-path to all-path.

V. Inferring Paths and the Related Information

In the previous section we briefly pointed out the information that can be inferred from $h_n^{(\alpha\to 0)}$ and $N_n^{(\alpha\to 0)}$ which are for the shortest-path case. Based on Theorem 2, the vector $l_{\{n\}}$ which is the shortest-path length from node all the other nodes of the network to node n can be obtained from equation: $l_{\{n\}} = h_n^{(\alpha\to 0)}$. We also said that $N_n^{(\alpha\to 0)}$ gives the subgraph of nodes containing the shortest-path. This matrix gives other information which we discuss more in detail in this section.

Consider a toy example with 6 nodes and the shown connectivities in Figure 1. $h_n^{(\alpha=10^{-5})}$ and $N_n^{(\alpha=10^{-5})}$ for this network are as follows:

$$h_6^{(\alpha=10^{-5})} = \begin{bmatrix} 3.0000 \\ 2.0000 \\ 1.0000 \\ 2.0000 \\ 1.0000 \end{bmatrix} \tag{25}$$

$$N_6^{(\alpha=10^{-5})} = \begin{bmatrix} 1.0000 & 0.0000 & 0.0000 & 1.0000 & 1.0000 \\ 0 & 1.0000 & 0.5000 & 0.0000 & 0.5000 \\ 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0.0000 & 0.0000 & 1.0000 & 1.0000 \\ 0 & 0 & 0 & 0 & 1.0000 \end{bmatrix} \tag{26}$$

As can be seen, vector $h_6^{(\alpha=10^{-5})}$ gives the shortest-path lengths from node $i = 1, 2, ..., 5$ to node 6.

In matrix $N_6^{(\alpha=10^{-5})}$, the non-zero entries of i-th row ($i = 1, ..., 5$) are presenting the nodes occurring on the shortest-path from node i to node 6. For example from the first row, we infer that the shortest-path from node 1 to 6, passes nodes 1, 4, and 5. And from row 4 we see that it does not go through the longer path which includes node 2 and it picks the shortest-path passing node 5.

Counting the non-zero entries of the columns in matrix N_6, we can find that each node occurs on how many shortest-paths. In our toy example, $(1, 1, 2, 2, 4)$ is the sequence of number of shortest-paths that node 1 to 5 are passed on respectively. This quantity has a concept of centrality and importance of nodes in the network. For example in routings based on the shortest-paths, the node with larger number of shortest-paths passing from it, is more critical and its failure causes more damages to the network. Node 5 is the most critical node in our example from this point of view.

There are also some other interesting information inferable from matrix $N_6^{(\alpha\to 0)}$ which we point out using the toy example without any proof.

The values of non-zero entries of $N_6^{(\alpha\to 0)}$ are also informative. If the value of the non-zero entries of row i are all equal to 1, there is only one shortest-path from node i to the destination. But if there are values other than 1 in row i, it means that there are more than one shortest-path from node i to the destination. For example the second row of matrix above shows that there are two shortest-path from node 2 to node 6. These two paths are $(2, 3, 6)$ and $(2, 5, 6)$. Order of nodes on the shortest-paths can be inferred from vector $h_6^{(\alpha=10^{-5})}$, the node with longer shortest-path is farther from the destination node and comes sooner in ordering the nodes on the path. However for larger number of shortest-paths between two nodes, we can only determine the level of the intermediate nodes by using the shortest-path length vector and it is not trivial to find the exact paths i.e. to figure out which of the non-zero entries are involved in a specific shortest-path. It needs applying some algorithms like depth-first search. Popular algorithms like Dijkstra's algorithm cannot handle this issue either [18] and needs to do a depth-first search too.

Since Number of passages through nodes has a concept of network flow, we can find a lower bound for the number of shortest-path between node i and the destination ($n_{shp}(i, destination)$). We denote the minimum value of non-zero entries in row i of matrix N_n by $v_{min}(i)$ and we have the following inequality:

$$n_{shp}(i, destination) \geq \lfloor \frac{1}{v_{min}(i)} \rfloor \quad (27)$$

For determining the level of intermediate nodes that was mentioned in the previous paragraph, we order the non-zero entries of a row based on their distance to the destination. In our toy example, the shortest-path from node 1 to 6 has two levels each with only one member. Node 4 is in level one and node 5 is in level two. The shortest-paths from node 2 to node 6 has only one level but with two members belonging to it (nodes 3 and 5). Considering the number of members belonging to levels of a row, the smallest number of members is an upper bound for the number of disjoint shortest-path.

For the cases other than the shortest-path case (larger α's), h_n^α gives the average path length and the non-zero entries of N_n^α that are larger than some threshold show the involved nodes on the paths.

VI. GENERALIZING THE FORMULATION FOR ANY DESTINATION AND THE ALGORITHM COMPLEXITY

In previous sections we presented an algorithm and the corresponding formulation to derive a continuum from the shortest-path to all-path via random walk approaches. For the sake of clarity and to avoid "index hell" during the proofs, we put the desired destination to be node n in all these formulation. In this section, we generalize the formulation for any destination j, $j \in \{1, ..., n\}$.

For a network with n nodes and P_{org} as its probability transition matrix, N^α is the number of passages through nodes for the corresponding evaporating network with evaporating node (node $n+1$) as the only absorbing node of the network. It can be obtained from the following equation:

$$N^\alpha = (I - \alpha P_{org})^{-1} \quad (28)$$

The following lemma uses the Schur complement to express the inverse of a submatrix in terms of the submatrices of the original matrix inverse.

Lemma 1: [28] There is a relation between the inverse of a matrix X and the inverse of its submatrix $X(-j,-j)$. We denote the inverse of X by matrix Y: $Y = X^{-1}$. Matrix Y can be partitioned into four parts:
1) $Y(-j,-j)$, which is the submatrix of Y by suppressing row and column j, 2) $Y(j,j)$, the j,j-th entry of Y, 3) $Y(-j,j)$, the j-th column of Y not including $Y(j,j)$, 4) $Y(j,-j)$, the j-th row of Y not including $Y(j,j)$. The relation between inverses can be written in terms of the introduced four parts of Y:

$$(X(-j,-j))^{-1} = Y(-j,-j) - \frac{Y(-j,j)Y(j,-j)}{Y(j,j)} \quad (29)$$

□

Based on this lemma, we can write $N^\alpha_{\{j,n+1\}}$ in terms of N^α partitions. The matrix $N^\alpha_{\{j,n+1\}}$ is the number of passages through nodes for the corresponding evaporating network before absorption by *either* node j or node $n+1$:

$$N^\alpha_{\{j,n+1\}} = N^\alpha(-j,-j) - \frac{N^\alpha(-j,j)N^\alpha(j,-j)}{N^\alpha(j,j)} \quad (30)$$

Based on equation (7), for q_j^α we have the following equation:

$$q_j^\alpha = N^\alpha_{\{j,n+1\}} R(:,1) = N^\alpha_{\{j,n+1\}}(\alpha V), \quad (31)$$

where $R(:,1)$ is the first column of R which is the non-absorbing to absorbing part of probability transition matrix.

Thus the general form of equation (8) and (9) for the evaporating network and any choice of destination j can be obtained by substituting $N^\alpha_{\{j,n+1\}}$ and q_j^α from equations above:

$$h_j^\alpha = (N^\alpha_{\{j,n+1\}} b_j^\alpha) \oslash b_j^\alpha \quad (32)$$

$$N_j^\alpha = Diag(b_j^\alpha)^{-1} N^\alpha_{\{j,n+1\}} Diag(b_j^\alpha) \quad (33)$$

This means that we can calculate h_j^α and N_j^α for *all* pairs of nodes in the network by just having N^α for each choice of α. This keeps the complexity of our algorithm equal to $O(n^3)$. Because the most complex part of our algorithms is to compute N^α from which we can derive $N^\alpha_{\{j,n+1\}}$ for all $j = 1, ..., n$ and having $N^\alpha_{\{j,n+1\}}$, we can obtain h_j^α and N_j^α. As can be seen from equation (30), the complexity of derivation of $N^\alpha_{\{j,n+1\}}$ from N^α is equal-less than $O(n^3)$. We can do even about one order of complexity better if the matrix $I - \alpha P_{org}$ in equation (28) is sparse [29].

VII. NUMERICAL EXAMPLES

In this section we bring two numerical examples, one is a synthetic network and the other one is a real network whose data has been collected through the internet. Consider a Broadcast Ring Network with n number of nodes. The structure of the network is in a way not to be sparse and is designed to have a large diameter ($n-1$) to make it more challenging for our method. Figure 2 shows the network for $n = 20$. The starting/source node is node 12 and the destination is node 5. The red nodes (dark nodes in B&W version) are the nodes occurring on the shortest path determined by our algorithm for $\alpha = 10^{-4}$. The connecting links (Thick edges) which show the shortest path have been determined based on the level of the nodes occurring on the shortest path as was discussed in section 5. The green nodes (grey nodes in B&W version) are the nodes occurring on the paths that appear by increasing α. The corresponding α for each green node has been shown in the figure. The yellow nodes (light nodes in B&W version) are not passed before that α gets to 9×10^{-1} which is very close to 1.

A numerical experiment has been done for the same structure but for larger network size like $n = 60$ where our algorithm still works well and the shortest-paths are derived perfectly.

Our algorithm has also been evaluated on some real networks with more than 1000 number of nodes and more than 10000 number of edges. One of the examples is a

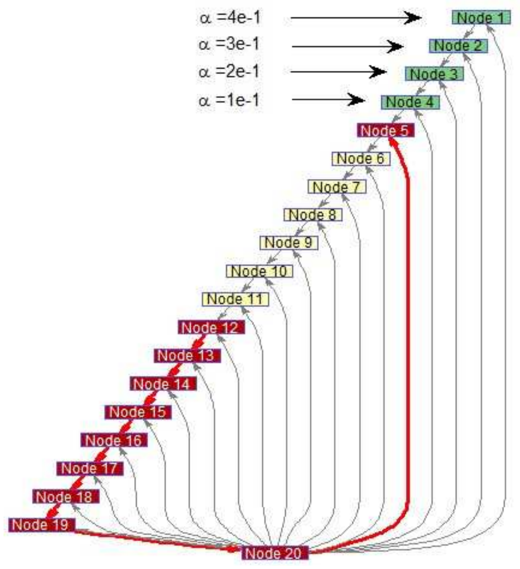
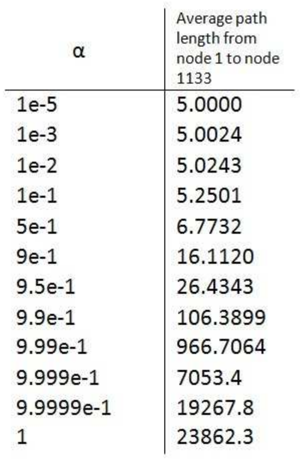


Fig. 2. Shortest path and other paths in Broadcast Ring Network



α	Average path length from node 1 to node 1133
1e-5	5.0000
1e-3	5.0024
1e-2	5.0243
1e-1	5.2501
5e-1	6.7732
9e-1	16.1120
9.5e-1	26.4343
9.9e-1	106.3899
9.99e-1	966.7064
9.999e-1	7053.4
9.9999e-1	19267.8
1	23862.3

TABLE I
AVERAGE PATH LENGTH FROM NODE 1 TO NODE 1133 FOR SEVERAL α

network of e-mail interchanges between members of the University Rovira i Virgili (Tarragona) [30]. This network has 1133 nodes and 10902 edges. Our program gives the shortest path lengths from one node to all other nodes and the details of the paths in just 0.8 seconds (written in MATLAB) and the results have been checked with well-established algorithms like Dijkstra and has proved to be exactly the same. By increasing α slightly, we obtain other alternative short paths which are longer than the shortest path but are still short in comparison to all existing paths in the network. Results of the path lengths between node 1 and node 1133 for several α has come in Table 1. The result for α=1e-5 is equal to the shortest path length from node 1 to node 1133 and $\alpha = 1$ is equal to the Hitting Time from node 1 to node 1133. For α=1e-5, the subgraph of involved nodes is composed of $\{1, 11, 23, 131, 269, 701, 1096\}$. These involved nodes are the non-zero entries of the first row of N_{1133} and their values are as follows: $\{1.0000, 0.7522, 0.2478, 0.7522, 0.2478, 1.0000, 1.0000\}$. For determining the zero entries and non-zero entries of matrix N_{1133}, we put the threshold to be one order of magnitude larger than α. Hence the reported nodes are the nodes that have values larger than 1e-4. The other entries had values smaller than 1e-5 (value of α). The smaller the α, the better seperation of zero and non-zero entries is provided. Comparing this result with results obtained from "graphshortestpath" function of MATLAB (which uses the popular shortest-path methods came in the related work), we observe that our set of involved nodes has two more entries $\{11, 131\}$. Analyzing the mentioned values of the involved nodes and their closeness level (their Hitting Time to node 1133), we figure out that there are more than one shortest-path from node 1 to node 1133: all of them have nodes $\{1, 701, 1096\}$ on their way, but they have one of $\{11, 23\}$ as their second level and one of $\{131, 269\}$ as their third level. The "graphshortestpath" function of MATLAB gives only $\{1, 23, 269, 701, 1096\}$ as the shortest-path and does not find the other ones. For α=1e-3, the subgraph of involved nodes would be larger: {1,3,4,11,13,18,19,20,21,23,31,43,45,59,131,137,228,256,269, 399,499,701,1096}. It can be seen that this set includes 23 nodes of which 8 nodes are from the shortest-path set. Although these real networks might have large size, It is shown that empirical networks mostly have the small-world property [31] and their diameter is not very large. That is why we designed a network in our synthetic example with large diameter to evaluate our method for large diameters.

VIII. CONCLUSIONS AND FUTURE WORK

We have presented a method for traversing the continuum of paths from shortest paths to all-paths for a network represented by a directed graph at a modest cost (cubic for all-paths together for each value of the sliding parameter). The method is based on the idea of an evaporating random walk, which allows on to vary the relative weights between shorter paths and longer paths. At one extreme we obtain the shortest paths between any pair of nodes at once, at the other extreme we obtain all paths with finite non-zero probability. Because we obtain almost-short paths for all pairs of source-destinations at once, this allows one to explore the effect on congestion or other effects due to the interplay of traffic on several paths on the network all at once. This exploration will be the subject of future research. Solving the shortest-path problem for the *weighted* graphs *efficiently* is another goal of us for future work. The presented algorithm can solve the weighted case if weighted links with weight w are substituted by $w-1$ number of nodes and w number of unweighted links (e.g. replacing a link with weight 3 with a chain of 2 nodes and 3 unweighted links) which makes the algorithm inefficient.

REFERENCES

[1] M. E. J. Newman (2005). A measure of betweenness centrality based on random walks, *Social Networks*, 27:39–54.

[2] L. Freeman (1977). A set of measures of centrality based upon betweenness, *Sociometry* 40:35–41.

[3] M. Rattigan, M. Maier, and D. Jensen (2007) Exploiting network structure for active inference in collective classification. *ICDM Workshop on Mining Graphs and Complex Structures*, pages 429–434.

[4] Overview of open shortest-path first, version 2 (OSPF V2) routing in the tactical environment.

[5] P. Sanders (2009). Fast route planning. Google Tech Talk.

[6] D. Z. Chen (1996). Developing algorithms and software for geometric path planning problems. *ACM Computing Surveys* 28 (4es): 18. doi:10.1145/242224.242246.

[7] D. Boley, G. Ranjan, and Z.-L. Zhang. Commute Times for a directed graph using an asymmetric Laplacian. *Linear Algebra and Appl.*, 435:224–242, 2011.

[8] C. M. Grinstead and J. L. Snell (2000). *Introduction to Probability. McGraw Hill*.

[9] P. Doyle and J. Snell (1984). *Random Walks and Electric Networks*. The Math. Assoc. of Am. `front.math.ucdavis.edu/math.PR/0001057`.

[10] M. Chen, J. Liu, and X. Tang (2008). Clustering via random walk Hitting Time on directed graphs. In *AAAI*, pages 616–621.

[11] A. Stojmirovic and Y. K. Yu (2007). Information flow in interaction networks. *J Comput Biol*, 14(8), 1115–1143.

[12] F. Fouss, A. Pirotte, J. Renders, and M. Saerens (2007). Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Trans. on Knowledge and Data Engineering*, 19:355–369.

[13] S. Biswas and R. Morris (2005). Exor: opportunistic multi-hop routing for wireless networks. *SIGCOMM 05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 133-144.

[14] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin (2001). Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *SIGMOBILE Mobile Computing and Communications Review*, 5(4):11-25.

[15] Y. Li and Z.-L. Zhang (2010). Random walks on digraphs: framework for estimating transmission costs in wireless *INFOCOM 10: Proceedings of the 29th IEEE Conference Communications*.

[16] L. Popa, C. Raiciu, I. Stoica, and D. Rosenblum (2006). Reducing congestion effects in wireless networks by multipath routing. *ICNP06: IEEE International Conference on Network Protocols*, pages 96-105.

[17] E. W. Dijkstra (1959). A note on two problems in connexion with graphs. *Numerische Mathematik* 1:269–271. doi:10.1007/BF01386390.

[18] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein (2001). Section 24.3: Dijkstra's algorithm. Introduction to Algorithms (Second ed.). MIT Press and McGraw-Hill. pp. 595–601. ISBN 0-262-03293-7.

[19] R. Bellman (1958). On a routing problem. *Quarterly of Applied Mathematics* 16: 87–90. MR0102435.

[20] R. W. Floyd (1962). Algorithm 97: shortest-path. *Communications of the ACM* 5 (6): 345. doi:10.1145/367766.368168.

[21] P. E. Hart, N. J. Nilsson, and B. Raphael (1972). Correction to "A Formal Basis for the Heuristic Determination of Minimum Cost Paths". *SIGART Newsletter* 37: 28–29.

[22] P. E. Black (2004), Johnson's Algorithm, Dictionary of Algorithms and Data Structures, National Institute of Standards and Technology.

[23] J.G. Kemeny and J.L. Snell (1976). Finite Markov Chains. Springer-Verlag.

[24] J. Norris, Markov Chains (1997). Cambridge Univ. Press.

[25] Y. Li, Z. L. Zhang, and D. Boley (2011). The routing continuum from shortest-path to all-path: A unifying theory. In *The 31st Int'l Conference on Distributed Computing Systems (IEEE ICDCS 2011)*.

[26] A. Mantrach, L. Yen, J. Callut, K. Francoisse, M. Shimbo, and M. Saerens (2010). The sum-over-paths covariance kernel: A novel covariance measure between nodes of a directed graph. *IEEE Trans. Patt. Anal. & Machine Intell*, 32(6):1112–1126.

[27] A. Tahbaz-Salehi and A. Jadbabaie (2006). A one-parameter family of distributed consensus algorithms with boundary: From shortest paths to mean Hitting Times. *45th IEEE Conference on Decision and Control*, 4664–4669.

[28] C. K. Koc and G. Chen (1994) Inversion of all principal submatrices of a matrix. *IEEE Trans. on Aerospace and Electronic Systems*, vol. 30, no. 1, pp. 280–281.

[29] S. Li, S. Ahmed, G. Klimeck, and E. Darve (2008). Computing entries of the inverse of a sparse matrix using the FIND algorithm. *J. Comput. Phys.*, 227(22):9408–9427.

[30] R. Guimera, L. Danon, A. Diaz-Guilera, F. Giralt and A. Arenas (2003) Physical Review E , vol. 68, 065103(R).

[31] D. J. Watts and S. H. Strogatz (1998). Collective dynamics of "small-world" networks. *Nature* 393:440–442.

ACKNOWLDGEMENTS

This research was supported in part by NSF grant IIS-0916750.