# Characterizing Dynamic Graphs with Continuous-time Random Walks


## ABSTRACT

Dynamic networks pervade many aspects of our lives and provide a growing and rich set of services in commerce, government, communications, and connectedness. In this work we consider network graphs whose edges but not nodes vary over time. Some examples of such networks include mobile wireless networks and delay tolerant networks. Characterizing and measuring "fast" changing networks is challenging as traditional techniques devised for static networks are rendered unsuitable. In this work we study the application of continuous-time random walks (CTRW) to characterize Markov dynamic graphs. In this dynamic graph model, a fixed set of static graphs are modulated according to a Markovian process. We consider a CTRW that progresses at a fixed rate and another that progresses with a rate proportional to the vertex degree. We derive closed-form analytical results for the steady state distribution of these walkers and show that their behavior is strikingly different. We then apply CTRW to two problems: (1) estimating vertex and edge characteristics, and (2) spatiotemporal node clustering. The later is a fundamental problem as nodes should be clustered according to their connectivity is both space and time. Finally, we evaluate our methods on synthetic dynamic graphs as well as real world traces.


## 1. INTRODUCTION

There has been a rapid increase in the number and types of digital networks over the last decade beginning with the Internet, its constituent networks, and the World Wide Web, and with the addition of on-line social networks such as Facebook, Twitter, etc., and mobile wireless networks (MANETs), delay tolerant networks (DTNs), and online information networks (CDNs, P2P file sharing, Wikipedia). These networks pervade many aspects of our lives and provide a growing and rich set of services in commerce, government, communications, and connectedness.

These networks can be in a dynamic state of flux and differ from infrastructure networks such as the Internet in that they are both large and dynamic even over small time intervals. Analyzing and measuring such "fast" changing networks are challenging, due to their dynamism and size. For example, the topologies of MANETs and DTNs are constantly changing, and that of an OSN such as Facebook and online knowledge networks (CDNs, P2P, Wikipedia) is rapidly growing and evolving with users and information coming and going. As a consequence, traditional techniques designed for static networks are rendered unsuitable.

Dynamic networks are commonly represented by graphs that change over time. Graph dynamics fall into two classes. The first class includes graphs that grow over time with the addition of nodes and edges (and infrequent node and/or edge deletions). Such dynamic graphs capture the evolution of the Internet over time and the growth of online social networks such as Facebook. Examples of models that attempt to capture this dynamic include linear preferential attachment [6]. The second class includes graphs that change but do not grow over time. A particular class, which is our focus, includes graphs whose edges but not nodes vary over time. Such a graph models the behavior of mobile networks and DTNs over time. Other examples include email graphs and network traffic graphs. It is worth noting that our focus is not on modeling dynamic networks, a topic that is in its infancy [7, 15, 19, 37], but rather to study how to characterize and measure such networks.

### *Random walks on static graphs.*

Static networks have been thoroughly studied in the literature. A rich variety of structural properties have been proposed and measured on real-world networks including degree distribution, assortativity, centrality, and clustering. Random walks (RWs) have an important role in understanding and characterizing these metrics. A RW can be described as a walker that at each step chooses uniformly at random a neighbor from the set of neighbors. RWs have proven useful for identifying "central nodes" [32], searching for content [8, 17, 18, 25, 21, 41], clustering nodes [4, 23, 24, 27, 31, 33], and characterizing unknown graphs [16, 26, 35, 36]. The simple closed form solution of the distribution of the number of visits a stationary RW pays to a node (which is proportional to the node degree) is the basis of principled RW applications in sampling and clustering static graphs [24, 36] (more details can be found in Sections 3 and 4). In contrast, little is known about random walks on dynamic continuous-time-varying networks and their applications.

### *Contibutions.*



*This work is under review.*

As RWs have been shown to be invaluable in understanding and characterizing static graphs, we believe that continuous-time random walks (CTRW) can have a similar role in characterizing dynamic graphs, a belief reinforced by the promising results presented in this work. The choice of a CTRW is a natural one as the graph dynamics require a walker that also has a continuous time component. In this context, the main contributions of this work are as follows.

- We model and analyze a continuous time random walk (CTRW) on a dynamic Markov modulated time varying graph. We consider two walks, one where the walker progresses at a steady rate (CTRW) and a second where the rate is proportional to the degree of the node currently being visited (CTRW-D).
- In the case of a fixed rate CTRW, we observe that the steady state distribution depends on the walker rate. By considering this process as nearly-completely decomposable, we characterize the stationary distribution in the extremes where the walker rate is large enough (i.e., the walker is much faster than the graph dynamics) and where the rate is small enough (i.e., the walker is much slower than the graph dynamics).
- In the case of the degree dependent walker rate CTRW-D, we obtain the stationary state distribution in closed form and find that it is insensitive to the base walker rate, graph dynamics and graph structure. This promising result enables the design of principled mechanisms to characterize dynamic graphs.
- We present two applications of CTRW-D:
  **Sampling:** estimate simple graph characteristics such as the fraction of blue and red nodes in a colored graph and the fraction of time that a given edge exists.
  **Clustering:** A dynamic graph clustering algorithm must classify nodes not only according to their spatial separation (how they are connected on each snapshot of the dynamic graph) but also on how well they are connected in time. We propose a principled technique to uncover spatiotemporal clusters, which clusters nodes defined in space and time.

*Outline.*

The remainder of this paper is organized as follows. In Section 2 we present the Markov dynamic graph models and continuous-time random walks. In Section 3 we evaluate the steady state behavior of the CTRW and CTRW-D on Markov dynamic graphs. In Section 4 we define how CTRW-D yields a principled mechanism for clustering dynamic graphs. Section 5 presents some results related to sampling and clustering. Section 6 describes the related work and finally Section 7 presents a discussion and the conclusion.

# 2. DYNAMIC GRAPHS & RANDOM WALKS

In this section we describe a class of Markov dynamic graph models and random walks. The dynamic graph models we consider assume that nodes are always present but that edges can come and go over time. Random walks on these graphs behave as follows. When a walker takes a step, only incident edges that are present at that time are considered as possible alternatives. We next describe these models in detail.

## 2.1 Markov dynamic graphs

A Markov dynamic graph is a set of graphs having the same vertex set and a Markov process over them. In particular, let $S = \{G_1, ..., G_m\}$ be a set of undirected graphs all having the same vertex set $V$, thus, $G_k = (V, E_k)$, where $E_k$ denotes the set of edges of graph $G_k$, for $k = 1, \ldots, m$. Let $n = |V|$ denote the number of vertices in a graph. Moreover, let $A_k$ denote the adjacency matrix of graph $G_k$. Thus, for all $i, j \in V$, $A_k(i, j) = 1$ if $(i, j) \in E_k$ and 0 otherwise. Finally, let $\deg(i, k)$ denote the degree of vertex $i \in V$ in graph $G_k$.

We introduce the dynamic graph process $\{G(t)\}$ as a continuous-time stationary Markov process with state space $S$. Let $\lambda_{kl} \geq 0$ denote the transition rate from state (graph) $G_k$ to $G_l$. Note that the graph dynamics are fully determined by the transition rates.

## 2.2 Edge Markov graphs

A special class of Markov dynamic graph are edge Markov graphs $G = (V, E, \Lambda)$, where $V$ and $E$ are the set of vertices and edges of the graph, respectively, and $\Lambda$ a function over the edges. In edge Markov graphs, edges alternate between being present and absent from the graph according to independent on-off processes. Let $\Lambda_0(e)$ and $\Lambda_1(e)$ denote the rate at which edge $e \in E$ changes from the on state to the off state and from the off state to the on state, respectively. Note that these rates can be edge dependent.

A few observations on the model follows. Let $q_e$ denote the steady state fraction of time that edge $e \in E$ is on, which is simply given by $q_e = \Lambda_1(e)/(\Lambda_0(e) + \Lambda_1(e))$. Let $G_k$ be a particular *configuration* of the graph. In particular, $G_k = (V, E_k)$ is a graph with the same vertex set $V$ where a subset of edges are present, $E_k \subset E$. In particular, the edge Markov graph induces a total of $2^{|E|}$ configurations, which represent all possible labelled subgraphs over an edge set with $|E|$ edges. Finally, let $P(G_k)$ denote the steady state fraction of time that the edge Markovian model spends in configuration $G_k$, with $k = 1, \ldots, 2^{|E|}$. In particular, we have that

$$P(G_k) = \prod_{e \in E_k} q_e \prod_{e \in E \setminus E_k} (1 - q_e) \qquad (1)$$

We note that edge Markov graphs are a special type of Markov dynamic graphs, where the set of $S$ of graphs correspond to the set of all configurations and transitions among configurations can be determined from the on-off process of the edges. In the remainder, we focus on the more general Markov dynamic graph.

## 2.3 Random walks on dynamic graphs

We consider a continuous time random walk (CTRW) on a general Markov dynamic graph. Intuitively, the walker can only traverse edges that are incident to the vertex at which the walker resides, choosing uniformly at random among them. Let $\{W(t)\}$ be the continuous time process representing the vertex where the walker resides, thus $W(t) = i \in V$, for any $t$. The time between two consecutive steps of the random walk is exponentially distributed with rate $\gamma$. We refer to $\gamma$ as the *walking rate*.

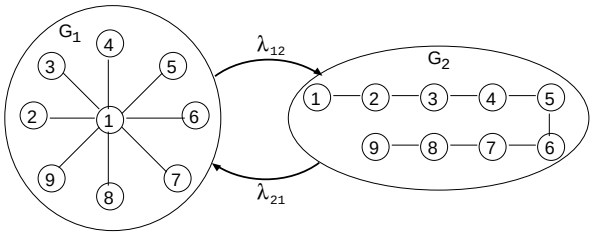


**Figure 1: Example of a simple Markov dynamic graph.**

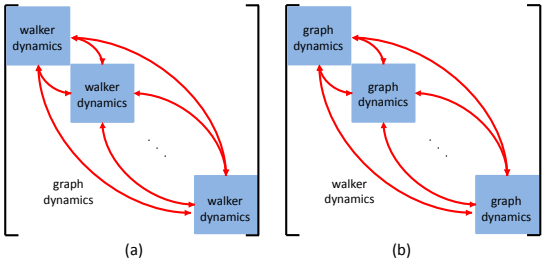


**Figure 2: Two different matrix arrangements for CTRW on Markov dynamic graphs.**

Without loss of generality, assume that when the random walk takes a step at time $t$, the Markov dynamic graph process is in state $G_k$, that is $G(t) = G_k$ and that $W(t) = i$. Let $N_{i,k}$ denote the set of neighbors of vertex $i$ in the graph $G_k$. Thus, the probability the walker steps to vertex $j \in N_{i,k}$ is given by $1/|N_{i,k}|$. Finally, if $N_{i,k}$ is empty, the walker stays at vertex $i$ during that step.

## 2.4 Joint graph and walker dynamics

The dynamic graph process and the random walk process can be represented as a continuous-time Markov process $\{R(t)\}$ where $R(t) \in V \times S$. Let $s_{i,k}$ represent a state of this process where $i$ represents a vertex of $V$ ($i \in V$) and $k$ represents a graph in $S$ ($k \in [1, \ldots, m]$). We call $s_{i,k}$ a spatio-temporal node (STnode). Let $U = \{s_{i,k} : \forall i \in V, k = 1, \ldots, m\}$ denote the set of all STnodes. The infinitesimal generator matrix of $\{R(t)\}$ is determined by combining the graph dynamics with the random walk, as we next describe.

Let $Q_k$ denote the infinitesimal generator matrix for the random walk dynamics conditioned on graph $G_k$. We have:

$$Q_k = \gamma D_k^{-1} A_k - \gamma I \,, \tag{2}$$

where $A_k$ is the adjacency matrix of $G_k$, $D_k$ is a diagonal matrix with the vertex degrees of $G_k$, $M^{-1}$ is the inverse of matrix $M$, and $I$ is the identity matrix with dimension $n$. The infinitesimal generator matrix of $Q$ of the joint process $\{R(t)\}$ is given by:

$$Q = \left[\begin{array}{c|c|c} Q_1 - \lambda_1 I & \ldots & \lambda_{1m} I \\ \hline \vdots & \ddots & \vdots \\ \hline \lambda_{m1} I & \ldots & Q_m - \lambda_m I \end{array}\right] , \tag{3}$$

where $\lambda_k = \sum_l \lambda_{kl}$ which denotes the aggregate rate out of graph $G_k$.

We illustrate this construction using a simple example. Consider the dynamic graph model shown in Figure 1 with two graphs $G_1$ and $G_2$ and transition rates $\lambda_{12}$ and $\lambda_{21}$ between them. The infinitesimal generator matrix $Q$ in this case is given by:

$$Q = \left[\begin{array}{c|c} Q_1 - \lambda_{12} I & \lambda_{12} I \\ \hline \lambda_{21} I & Q_2 - \lambda_{21} I \end{array}\right] \tag{4}$$

## 2.5 Block matrix representation of $Q$

The construction of matrix $Q$ shown in eq. (4) induces the following intuitive partition of the matrix. Let $P^w = \{P_1^w, \ldots, P_m^w\}$ be a partition of the state space $U$, where each subset $P_k^w$ consists of all random walk states corresponding to the graph $G_k$. In particular, $P_k^w = \{s_{i,k} | i \in V\}$. Thus, each partition corresponds to a square matrix block of size $n = |V|$ in the matrix arrangement described above. Moreover, transitions within a partition correspond to random walk dynamics and transitions among blocks correspond to graph dynamics, as illustrated in Figure 2(a). Finally, note that transitions among blocks depend only on the transition rates associated with the graph dynamics (i.e., $\lambda_{kl}$), but not $\gamma$.

We consider a second (also intuitive) partition of $Q$ where states $s_{i,k} \in U$ are sorted first by $i$ and then by $k$. We denote this second partition as $P^g = \{P_1^g, \ldots, P_n^g\}$, where each subset $P_i^g$ consists of all graph states corresponding to vertex $i$. In particular, $P_i^g = \{s_{i,k} | k \in [1, \ldots, m]\}$. Thus, each partition corresponds to a square matrix block of size $m$ (number of different graphs). Thus, within a block we have the graph dynamics and transitions among blocks represent the walker dynamics, as illustrated in Figure 2(b). Finally, note that transitions between blocks depend solely on $\gamma$, but not on $\lambda_{kl}$ for any $k, l$.

Considering the first arrangement, we present a more general procedure than simply aggregating individual graphs for the case where the graphs $G_k$ do not consist of a single connected component. In this case, each graph $G_k$ consists of one or more connected components. When considering a graph $G_k$, states are arranged contiguously according to their connected component. Thus, each connected component of a graph $G_k$ is represented by a block. In this matrix representation, each block represents a connected component of a graph.

*Remark:* One example where the second partition is intuitively evident consists of the edge Markov graphs, introduced above. In this case, the partition blocks are formed by the Kronecker product of two state on-off blocks (one block for each on-off edge) given that the edge dynamics are independent of each other. Clearly, edge Markov graphs give rise to very special matrix $Q$ structures that can be exploited.

## 2.6 Degree-dependent random walks

The random walker considered above walks at a constant rate $\gamma$ over the dynamics graph. We now introduce a variation where the walking rate is not constant, but instead depends on the degree of the vertex where the walker is located. Intuitively, our walker will move faster when it finds itself at vertices with large degrees and slower when located at vertices with small degrees. More precisely, the inter-step time of the random walk is exponentially distributed with rate $\deg(i,k)\gamma$, where $\deg(i,k)$ is the degree of vertex $i \in V$ in graph $G_k \in S$ and $\gamma > 0$. Note that each state $s_{i,k} \in U$

of the joint process will now have a possibly different transition rate for the walker. We will refer to this random walk as CTRW-D, where "D" stands for degree-proportional walking rates.

Similar to the CTRW, the infinitesimal generator of the joint graph and walker process $\{R(t)\}$ for the CTRW-D can also be constructed by combining their individual dynamics. For the general case, where we have $m$ distinct graphs $G_k$, the infinitesimal generator matrix is given by:

$$Q = \begin{bmatrix} \gamma A_1 - \gamma D_1 - \lambda_1 I & \dots & \lambda_{1m} I \\ \vdots & \ddots & \vdots \\ \lambda_{m1} I & \dots & \gamma A_m - \gamma D_m - \lambda_m I \end{bmatrix},$$

where $I$ is an $n \times n$ identity matrix, $\lambda_k = \sum_l \lambda_{kl}$, $A_k$ is the adjacency matrix of graph $G_k$, and $D_k$ is a diagonal matrix with the degrees of $G_k$ (row sum of $A_k$), $k = 1, \dots, m$.

Note that the arrangement of the matrix above corresponds to blocks that represent the walker dynamics, as illustrated in Figure 2(a). As with CTRW, all transitions that depend on $\gamma$ occur within a block, while transitions between blocks depend only on $\lambda_{kl}$.

Finally, as we describe in the next section, CTRW and CTRW-D behave very differently when walking on dynamics graphs, even when considering steady state characteristics.

# 3. STEADY STATE DISTRIBUTIONS

In this section we investigate the steady state distribution of continuous-time random walks on dynamic graphs. We first consider CTRW with constant rate and show that the steady state distribution depends on the walker rate. We follow with the analysis of CTWR-D and show that its steady state distribution is uniform, independent of base walker rate, graph dynamics and graph structure.

## 3.1 Existence of steady state

The $\{R(t)\}$ process defined by the joint graph and walker dynamics described in the previous section may not be ergodic. In particular, the Markov dynamic graph model does not impose any structural restrictions on the static graphs. Therefore, steady state for random walks on these graphs may simply not exist. For example, if all graphs are not connected and is never possible to go from one vertex $i \in V$ to some other vertex $j \in V$ over time.

In order to characterize the existence of steady state, we introduce the notion of paths in both space and time as follows.

DEFINITION 3.1. **STpath.** *A spatiotemporal path (or* STpath*) between STnodes $s_{i,k} \in U$ and $s_{j,l} \in U$ exists if the commute time of a CTRW between them is finite. The commute time is the expected time for a random walk starting at $i$ in static graph $G_k$ to first arrive at $j$ at static graph $G_l$ and then return to $i$ in static graph $G_k$.*

Note that STpaths can exists between vertices that are not part of the same connected component in some given graph.

Finally, it can be shown that if an STpath exists between any pair of STnodes $s_{i,k}$ and $s_{j,l}$ defined by the joint graph and walker dynamics, then process $\{R(t)\}$ is ergodic and therefore the random walk will have a steady state distribution.

In edge Markov graphs $G = (V, E, \Lambda)$, if the graph $G = (V, E)$ is connected, then the process $\{R(t)\}$ is ergodic, otherwise the process is not ergodic.

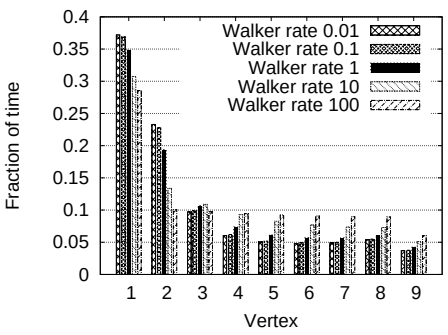


**Figure 3: Steady state distribution of random walker as a function of $\gamma$.**

## 3.2 CTRW steady state distribution

We are interested in the fraction of time that the random walk spends at each vertex $i \in V$ regardless of the graph dynamics. Clearly, this metric can be obtained from the steady state probability distribution of $\{R(t)\}$. Let $\pi(s_{i,k})$ denote the fraction of time in STnode $s_{i,k}$ where $i \in V$ and $k \in [1, \dots, m]$. In particular, assuming ergodicity in $\{R(t)\}$, we have $\pi(s_{i,k}) = \lim_{t\to\infty} P[R(t) = s_{i,k}]$. We denote vector $\pi = (\pi_1, \dots, \pi_n)$ in which the $i$-th component is the fraction of time the random walk spends in vertice $i$ of set $V$, where $n = |V|$. Therefore, $\pi_i = \sum_k \pi(s_{i,k})$. In what follows we investigate $\pi$, starting with a simple numerical example, and following with more general results.

Consider the Markov dynamic graph illustrated in Figure 1 where $\lambda_{12} = \lambda_{21} = 1$. Figure 3 presents the steady state distribution of the random walker for various values of $\gamma$. As the walking rate increases from $10^{-2}$ to $10^{2}$ we observe significant change in $\pi$. In particular, the steady state probability of state 2 reduces by more than 50% while that of state 5 increases by more than 50%. Thus, the random walk depends on the temporal structure of the dynamic graph and this dependency varies with the walker rate. Lets intuitively discuss the two extremes where $\gamma$ is sufficiently large and sufficiently small.

Consider a very fast walker, in particular, much faster than the timescale at which the graphs change. Thus, every time a graph is entered, the random walk quickly steps through all vertices in this graph, many times. Intuitively, the random walk converges to the steady state distribution of the static graph before the dynamic graph process changes to a new graph. In this scenario, the steady state distribution of the random walk is given by the unconditional steady state probability of each static graph.

Now consider a very slow walker, in particular, much slower than the timescale at which the graphs change. Every time the walker steps onto a vertex, the incident edges of this vertex change many times before the walkers steps off. Intuitively, when the walker steps off it will observe the incident edges in steady state. Thus, the probability the walker steps from vertex $i$ to vertex $j$ is given by the steady state probability that edge $(i, j)$ is present weighted by the probability that the walker chooses this edge. The steady

state distribution of the random walk is the solution of this new process on a weighted static graph. The intuition behind fast and slow walkers are formalized in the theorems of the next subsection.

### 3.2.1 Block decomposition

In this section we show how the steady state distribution of CTRW can be obtained through aggregation/disaggregation theory of Markov chains using block decomposition. We wish to explore how we might be able to determine the equilibrium state probabilities without having to solve the original (large) CTRW model. For that purpose, we first briefly review basic results on aggregation/disaggregation theory, that will form the basis for the calculations. In order to facilitate the explanation on aggregation we consider a discrete-time Markov chain. However, from uniformization results, it is well known that we can indistinguishably work with either the continuous-time or discrete-time versions [13].

Consider a discrete-time MC and partition the state space into $S$ subsets $\mathcal{S}_1, \ldots, \mathcal{S}_S$. Let $\boldsymbol{\pi}_i^*$ be the conditioned state probability vector for the states in subset $\mathcal{S}_i$ and define $\boldsymbol{\alpha}$ as the vector in which the $i$-th entry is the fraction of time the MC is in subset $\mathcal{S}_i$. Then:

$$\alpha_i = \sum_{j=1}^{N} \alpha_j \boldsymbol{\pi}_j^* \mathbf{P}_{ji} \mathbf{e}, \tag{5}$$

where $\mathbf{P}_{ij}$ is the matrix that includes the transition probabilities from states in subset $\mathcal{S}_i$ to those in $\mathcal{S}_j$. It is known that if we define $o_{ji} = \boldsymbol{\pi}_j^* P_{ji} \mathbf{e}$ we can re-write (5) as

$$\boldsymbol{\alpha} = \boldsymbol{\alpha} \mathbf{O} \tag{6}$$

where $\mathbf{O}$ is an stochastic matrix with $(i.j)$-th entry equal to $o_{ij}$. The state probabilities $\pi_i$ can be obtained from

$$\pi_i = \alpha_i \pi_i^*. \tag{7}$$

From above it is clear that, once we obtain the conditional state probabilities $\boldsymbol{\pi}_i^*$ for each partition $\mathcal{S}_i$, we can easily obtain the unconditional state probabilities for the MC. The issue then is how to efficiently obtain the $\boldsymbol{\pi}_i^*$.

We obtain the $\boldsymbol{\pi}_i^*$'s from the stochastic complement for the partition $\mathcal{S}_i$. The stochastic complement $\mathbf{C}_i$ for partition $\mathcal{S}_i$ is an stochastic matrix:

$$\mathbf{C}_i = \mathbf{P}_{ii} + \sum_j \mathbf{P}_{ij} \left[\mathbf{I} - \mathbf{P}_{jj}\right]^{-1} \mathbf{P}_{ji}. \tag{8}$$

In (8), the $ij$-th entry of $\left[\mathbf{I} - \mathbf{P}_{jj}\right]^{-1}$ can be interpreted as the expected number of visits to state $j \in \mathcal{S}_j$ before leaving $\mathcal{S}_j$, if $\mathcal{S}_j$ was entered from state $i$.

The stationary state probabilities for the stochastic complement are the conditional state probabilities for the associated states of the original Markov chain and we have:

$$\boldsymbol{\pi}_i^* = \boldsymbol{\pi}_i^* \mathbf{C}_i \tag{9}$$

### 3.2.2 Limiting results

Although equations (8) and (9) provides an exact procedure to calculate the $\boldsymbol{\pi}_i^*$'s, the calculations may not be efficient unless the model has a special structure. One such special structure constitute the *nearly completely decomposable systems* (NCD). In an NCD system, the off diagonal blocks probabilities (the probabilities of moving from a state to another in different partitions) are relatively small as compared to those within each blocks. (Similarly, in an NCD system, the transition rates between states in different partitions are much smaller as compared to the rate of events that are responsible for jumping from one state to another in the same partition.) This suggests that each time the system enters subset $\mathcal{S}_i$ it stays there long enough for the system to have reached a *local equilibrium* and the exact state at which $\mathcal{S}_i$ was entered has little effect on the $\boldsymbol{\pi}_i^*$. In order words, if the sojourn time in $\mathcal{S}_i$ is long enough so that the effect of the transient behavior in subset $\mathcal{S}_i$ has vanish, then the expected number of visits in a state of $\mathcal{S}_i$ becomes independent of which state $\mathcal{S}_i$ was entered. Courtois [12] formalizes these arguments. In what follows, we apply these results to the state probabilities of a CTRW for some special cases.

Let us first assume that the random walker rate is much faster as compared to the rates of graphs $G_k$ changes. For a sufficiently large $\gamma$, the state state probabilities of the random walk $\pi$ is a linear combination of the corresponding probabilities of each individual static graph $G_k$.

THEOREM 3.1. *For a sufficiently large $\gamma$, the steady state distribution $\pi$ of the random walk is given as follows. Let $\pi(G_k) = (\pi_1(G_k), \ldots, \pi_n(G_k))$ denote the steady state distribution of a random walk on the static graph $G_k$. In particular, assuming that $G_k$ is connected, we have that $\pi_i(G_k) = \deg(i,k)/\sum_j \deg(j,k)$ for $i = 1, \ldots n$ and $\deg(i,k)$ denotes the degree of node $i$ in graph $G_k$. Let $\Pi_k$ denote the steady state fraction of time spent in graph $G_k$. We then have that:*

$$\pi = \sum_{k=1}^{m} \Pi_k \pi(G_k) \tag{10}$$

PROOF. It follows trivially from the discussion above. We partition the state space as in Figure 2(a). Each partition corresponds to a given graph $G_k$. Since the random walker walks fast enough, the system is NCD and we can distributed the outgoing rates of each block in Figure 2(a) into the diagonal of each block. The $\boldsymbol{\pi}_i^*$ are then obtained from the solution of the static graph $G_i$ (the $\pi(G_k)$ in (10)). The $\boldsymbol{\alpha}$'s ($\Pi_k$ in (10)) are obtained from (6) and the final solution follows from (7)[1]. □

We now assume that the random walker rate is much slower than the rates of graph changes. For a sufficiently small $\gamma$, the state probabilities of the random walk $\pi$ is obtained from the solution of a MC that govern the changes between graphs.

THEOREM 3.2. *For a sufficiently small $\gamma$, the steady state distribution $\pi$ of the random walk is given by the steady state distribution of the following continuous time Markov chain with $n$ states corresponding to the vertex set. Let $R$ denote the rate matrix of this continuous time Markov chain and $r_{i,j}$ an element of $R$ with $i$ different from $j$. Define:*

$$r_{i,j} = \gamma \sum_{k=1}^{m} \mathbf{1}((i,j) \in E_k)\, \Pi_k\, 1/\deg(i,k) \tag{11}$$

*where $\mathbf{1}(\cdot)$ is the indicator function and $\deg(i,k)$ is the degree of vertex $i$ in graph $G_k$.*

[1]Note that the rate of moving from $G_l$ to $G_k$ is dependent on the particular node the walker is at the time of the move. This is why we need the conditional probabilities in each partition.

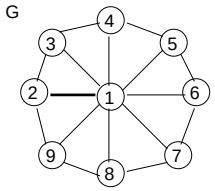


**Figure 4: The fixed but weighted graph seen by the slow walker on the example of Figure1. All edges have weight 0.5 with the execption of edge (1,2) which has weight one.**

PROOF. It follows from exactly the same principles as in Theorem 3.1 after partitioning the state space as in Figure 2(b). Each partition represents the dynamics of edge changes and the MC is NCD. Block $i$, corresponding to partition $\mathcal{S}_i$, represents a particular node in the graph. Since the edge dynamics are independent of the particular node the waker is in, and $\gamma$ is relatively small, then the conditional state probabilities are easily obtained by distributing the walker rates into the diagonal of each block. For instance, if the edge dynamics are modeled by independent on-off processes, then it is trivial to compute the conditional state probabilities for states in any partition. Likewise, for a general dynamic graph process, the conditional state probabilities will be given by the solution of the MC that represents the changes in the graphs with time. Once we obtain the conditional probabilities for the states in each partition, equations (6) and (7) are used to obtain the final solution. □

**Remarks**: From Simon-Ando theorems [39], the solutions outlined above can always meet a given level of accuracy for sufficiently large (or small) value of $\gamma$. For intermediate values of $\gamma$ the MC of the system is not NCD. However, one can resort on the special structure of the problem to obtain a solution efficiently from aggregation theory. We defer a discussion of these issues to a subsequent report, since our main focus in this paper is to show properties of dynamic graphs and, for this purpose, the above results suffice.

Returning to our example, consider the fast walker. Recall that the steady state distribution of a CTRW in a connected static graph $G_k$ is independent of the walker rate and is simply given by $\pi_i = \deg(i,k)/\sum_j \deg(j,k)$. Moreover, the fraction of time the process spends in each static graph is 0.5, since $\lambda_{12} = \lambda_{21} = 1$. Thus, the steady state probability of state $i$ is simply given by $\pi_i = 1/2(\deg(i,1)/16 + \deg(i,2)/16)$, which for state 2 is 3/32 (as can be confirmed in Figure 3).

As for the slow walker, Figure 4 shows the static but weighted graph over which the random walk will traverse. Recall that edges are not chosen uniformly at random, but instead proportional to the fraction of time they are present. In this example, all edges in the figure have weight 0.5 which correspond to the fraction of time they are present, with the exception of edge $(1,2)$ which has weight one, since it is present all the time. The steady state distribution of this process is very close to that of a slow walker shown in Figure 3.

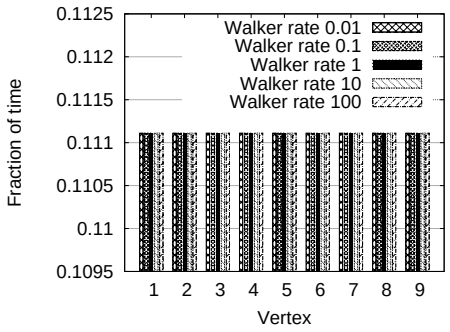


**Figure 5: As predicted by Theorem 3.3, a CTRW-D in steady state spends an equal amount of time on each vertex, regardless of the walker rate ($\gamma$), graph dynamics or structure of the each snapshot, $G_k$, $k = 1, \ldots, m$.**

## 3.3 CTRW-D steady state distribution

As before, we are interested in the fraction of time that the random walk spends at each vertex disregarding the graph dynamics. We start by considering the example presented above (illustrated in Figure 1), but with a CTRW-D walker. Figure 5 shows the steady state distribution for various base walker rates $\gamma$. Different from CTRW (compare with Figure 3), we observe no variation in $\pi$ as the walker rate changes. Moreover, the walker spends a fraction of time that is uniform across the vertices of the graph (the example has 9 vertices, which yields a fraction of time of $1/9 = 0.111$). This observation is quite remarkable, since it means we can traverse the graph at arbitrary speeds, irrespective of graph dynamics, while being able to characterize the steady state distribution that the walker will observe. In what follows, we prove this observation for all CTRW-D.

We start by noting that the steady state distribution of CTRW-D on static graphs $G_k$ is uniform over the set of vertices. Moreover, this result is also independent of $\gamma$, the walking rate. Intuitively, CTRW-D compensates the structural bias suffered by CTRW on by introducing a temporal bias. With CTRW-D, at the same time high degree nodes are more visited, less time is spent in them (higher walking rate). We can make this argument rigorous as follows.

Consider a connected graph $G_k$, $k = 1, \ldots, m$ and a CTRW-D walker. Let $v_i$ denote the fraction of visits to vertex $i \in V$ in steady state. Since fraction of visits is independent of time in a vertex, we have that $v_i = \deg(i)/\sum_j \deg(j)$, where $\deg(i)$ denotes the degree of vertex $i \in V$. Let $t_i$ denote the expected amount of time in vertex $i$ per visit, which is simply given by $t_i = 1/(\gamma \deg(i))$. Finally, the fraction of time spent in vertex $i$ is given by $\pi_i = (v_i t_i)/(\sum_j v_j t_j) = 1/n$, where $n = |V|$. Thus, CTRW-D in static graphs exhibit a uniform distribution over the set of vertices.

Interestingly, this same result follows for Markov dynamic graphs, as we show next. As before, let $\pi(s_{i,k})$ denote the fraction of time a CTRW-D in steady state spends at STnode $s_{i,k} \in U$. If the steady state distribution $\pi' = (\pi(s_{i,k}) : \forall s_{i,k} \in U)$ exists, it satisfies the set of equations $\pi' Q = \mathbf{0}$. In the following theorem we see that if the Markov dynamic graph is ergodic, then $\pi(s_{i,k})$ is the product of an uniform distribution over the vertices, $1/n$, with the fraction of time

that the Markov dynamic graph spends in graph $G_k$.

THEOREM 3.3. *If the Markov dynamic graph is ergodic $\pi(s_{i,k}) = \Pi_k/n$, $i \in V$, $k = 1, \ldots, m$, where $\Pi_k$ is the steady state fraction of time that the Markov dynamic graph spends in $G_k$.*

PROOF. Let $\pi' = (\pi'_1, \ldots, \pi'_{mn})$, where $\pi'_{(k-1)m+i} = c_k/n$, $i = 1, \ldots, n$ and $k = 1, \ldots, m$. We just need to find the values of $c_k$, $k = 1, \ldots, m$, that solve $\pi' Q = \mathbf{0}$. Note that $(c_k/n, \ldots, c_k/n)(A_k\gamma - D_k\gamma) = \mathbf{0}$, $\forall c_k \geq 0$, as $A_k$ is symmetric. Thus, $\pi' Q$ yields the set of equations:

$$-\lambda_k c_k + \sum_{\forall h \neq k} \lambda_{hk} c_h = 0\,, \quad k = 1, \ldots, m. \tag{12}$$

The Markov chain describing the dynamic graph is ergodic and thus $c_k = \Pi_k$ is a solution to equations (12), $k = 1, \ldots, m$. □

From the above result, we directly obtain the fraction of time CTRW-D spends in each vertex $i$ by simply summing over all $k$. Thus, $\pi_i = \sum_k \Pi_k/n = 1/n$. Thus, the CTRW-D in steady state experiences a uniform distribution over the set of vertices $V$.

# 4. SPATIOTEMPORAL CLUSTERING

In this section we propose an algorithm to perform clustering in dynamic graphs (spatiotemporal clustering). Section 4.1 introduces a well known and widely used RW-based clustering methods for static graphs. Then, in Section 4.2 we use the same formalism introduced in Section 4.1 to develop our spatiotemporal clustering method. Later in Section 5 we present two examples of applications.

## 4.1 Spatial (static graph) clustering

Spectral clustering techniques on static graphs have recently received considerable attention. A variety of spectral clustering methods can be found in the literature [24]. Random walk-based clustering [38] (a type of normalized graph Laplacian) is one of the most popular methods due to its nice theoretic properties and good performance on real world applications [24]. A justification to use random walks in graph partitioning can be found in a graph theoretic metric. We want to partition the graph into clusters (subsets) of nodes such that there are few edges between different clusters and many edges within clusters. In what follows we present an overview of this theoretical justification. Our primary goal here is not to present a tutorial on spectral clustering; a detailed explanation of the methods and their theoretic foundations can be found in [24]. Our interest is to use the following descriptions to guide us in developing a spatiotemporal clustering technique.

Let $G = (V, E)$ be a static, connected, non bipartite, undirected, unweighted graph with $n$ nodes. It is trivial to extend the following formulas to weighted graphs [24]. Let $B \subset V$ and $\bar{B} = V \backslash B$. Let $\delta(B, \bar{B})$ be the number of edges between nodes in $B$ and $\bar{B}$, i.e., the number of edges that have an endpoint in $B$ and another endpoint in $\bar{B}$. The normalized cut (Ncut) of $B$ is given by

$$\text{Ncut}(B) = \frac{\delta(B, \bar{B})}{\text{vol}(B)} \tag{13}$$

The minimum Ncut problem consists of choosing a partition of $V$ into disjoint non-empty subsets $B_1, \ldots, B_k$ that minimizes

$$\text{Ncut}(B_1, \ldots, B_k) = \sum_{i=1}^{k} \text{Ncut}(B_i) \tag{14}$$

where $\text{vol}(B_i) = \sum_{\forall v \in B_i} d_v$ and $d_v$ is the degree of node $v$. The minimum Ncut can be approximated using the spectral decomposition of a RW on $G$. In order to facilitate our transition to the dynamic graph case, our exposition uses CTRWs instead of simple discrete-time RWs. Let's look at a CTRW at the times in which the walker takes a step (thus, it is immaterial to us if we are looking at a CTRW-C or CTRW-D) $\{X_i\}_{i \in \mathbb{N}}$, where $i$ is the $i$-th walker transition, given that the system starts in steady state according to distribution $\psi$. Denote $P(\bar{B}|B) = P(X_1 \in \bar{B} | X_0 \in B)$. A formal equivalence between Ncut and transition probabilities of the above random walk has been observed in [28]:

$$\text{Ncut}(B, \bar{B}) = P(\bar{B}|B) + P(B|\bar{B}). \tag{15}$$

Thus, minimizing $\text{Ncut}(B_1, \ldots, B_k)$, eq.(14), is equivalent to finding disjoint non-empty partitions of $V$, $B_1, \ldots, B_k$, that, in steady state, minimize the number of transitions that a RW makes between these partitions. We will revisit this equivalence in Section 4.2 when we look at spatiotemporal clustering. Note that the above clustering formulation is closely related to the block decompositions performed in Section 3.2.1, where our objective was to find blocks of states (nodes) in the Markov chain (graph) where more RW transitions are performed within the block (cluster) than outside the block (cluster).

In what follows we present a short description of the clustering technique proposed by [38]. Let $A = [a_{ij}]$, $i, j = 1, \ldots, n$ denote the adjacency matrix of $G$. Let $D = \text{diag}(d_i)$, $i = 1, \ldots, n$, be a diagonal matrix with the degrees $d_1, \ldots, d_n$ of the graph. Consider the infinitesimal generator matrix of a CTRW-D on $G$

$$Q'_s = A - D. \tag{16}$$

Now consider the transition probability matrix of its corresponding embedded Markov chain at the transition points of the CTRW-D,

$$P_s = -\text{diag}(Q'_s)^{-1} Q'_s + I,$$

where $\text{diag}(M)$ is a matrix with the diagonal elements of $M$. Note that the diagonal elements of $A$ are zero and, thus, $\text{diag}(Q'_s) = -D$. Let $P_s u_i = \alpha_i u_i$, where $u_i$ is the eigenvector of $Q'_s$ corresponding to the $i$-th largest eigenvalue $\alpha_i$ of $P_s$. As

$$(-\text{diag}(Q'_s)^{-1} Q'_s + I) u_i = -\text{diag}(Q'_s)^{-1} Q'_s u_i + u_i, \quad \forall i\,,$$

the $i$-th largest eigenvalue of $-\text{diag}(Q'_s)^{-1} Q'_s$ is $\alpha_i - 1$ and its corresponding eigenvector is $u_i$. Note that the largest eigenvalue is $\alpha_1 = 1$ and its corresponding eigenvector is $u_1 = \psi$ (recall that $\psi$ is the stationary distribution of the number of visits of the CTRW-D). The spectral decomposition used in [38] is in fact the spectral decomposition of $-\text{diag}(Q'_s)^{-1} Q'_s$ but, as we see below, we can instead use the spectral decomposition of $P_s$ without changing the algorithm.

The rest of the algorithm in [38] is as follows. Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the $k$ first eigenvectors of $P_s$ as columns. For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the $i$-th row of $U$. We cluster the

points $\{y_i\}_{i=1,\ldots,n} \in \mathbb{R}^k$ into $k$ subsets, $C_1, \ldots, C_k$, using $k$-means [38]. The $k$ clusters are $B_i = \{j \in V | y_j \in C_i\}$, $i = 1, \ldots, k$. The use of $k$-means to determine the clusters also has a justification: the Euclidean distance between points $y_i$, $i = 1, \ldots, k$, is closely related to a "random walk diffusion-based distance" metric [30].

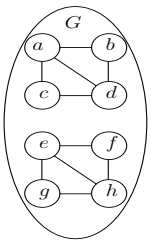


**Figure 6: A graph with two trivial clusters: $\{a, b, c, d\}$ and $\{e, f, g, h\}$**

Note that the above algorithm requires the number of clusters, $k$, to be known ahead of time. A simple way to obtain $k$ from the data uses the notion of a spectral gap [24, Section 8.3]. The principle behind this idea is to choose $k$ such that $\alpha_1, \ldots, \alpha_k$ are relatively large but $\alpha_{k+1}$ is relatively small. A justification for this procedure is based on the decomposition principles introduced in Section 2.5 (based on perturbation theory, see [24, Section 7.1] for more details). In the ideal case of $k$ completely disconnected clusters $\alpha_1 = \cdots = \alpha_k = 1$ with a gap to the $(k+1)$-th eigenvalue $\alpha_{k+1} < 1$. Consider the static graph shown in Figure 6. For the sake of simplicity we relax our assumption that $G$ is connected and allow subgraphs $\{a, b, c, d\}$ and $\{e, f, g, h\}$ to be disconnected. Arrange the adjacency matrix of $G$ in lexicographical order, i.e., the first row (column) represents $a$ and the last row (column) represents $h$. Note that $P_s$ has a trivial block decomposition

$$P_s = \left[\begin{array}{c|c} P_1 & \mathbf{0} \\ \hline \mathbf{0} & P_2 \end{array}\right]$$

and the spectral decomposition of $P_s$ shows eigenvectors $u_1$ and $u_2$ to be (up to a constant multiplier)

$$u_1 = \begin{bmatrix} 0.59 \\ 0.39 \\ 0.59 \\ 0.39 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \text{and} \quad u_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.59 \\ 0.39 \\ 0.59 \\ 0.39 \end{bmatrix},$$

respectively. Both $u_1$ and $u_2$ display two clear clusters; the elements in $u_1$ associated with $\{a, b, c, d\}$ are non-zero while its elements associated with $\{e, f, g, h\}$ are all zero; $u_2$ is the opposite.

In the above we reviewed a random walk-based technique to cluster static graphs. In what follows we extend the above ideas to dynamic graphs.

## 4.2 Spatiotemporal clustering

A dynamic graph clustering algorithm must classify nodes not only according to their spatial separation, like the two clusters in Figure 6, but also according to how well they are connected in time. Let $G(t)$ be a dynamic graph as described in Section 2. In what follows we extend the definition of an Ncut, eq.(13), to dynamic graphs.

Following the technique presented in Section 4.1, we look at the transition points of a CTRW-D but this time on the dynamic graph $G(t)$. Consider the sequence of visited STnodes, $\{X_i\}_{i\in\mathbb{N}}$, where $i$ is the $i$-th walker transition and $X_i \in U$, starting with $X_0$ taken from the stationary distribution $\psi$ of the number of visits to STnodes. Let $C \subset U$ be a subset of STnodes and denote $P(\bar{C}|C) = P(X_1 \in \bar{C} | X_0 \in C)$. The static graph Ncut, eq.(15), definition can be extended as follows.

DEFINITION 4.1. *Let $C \subset U$ be a set of STnodes. The dynamic graph Ncut is defined as*

$$DNcut(C, \bar{C}) = P(\bar{C}|C) + P(C|\bar{C}) .$$

The algorithm in Section 4.1 uses the spectral decomposition of $P_s$ to approximate the partition of $V$ into disjoint non-empty subsets $B_1, \ldots, B_k$ which minimize the total Ncut value of the partition (eq.(13)). We also presented an heuristic to find a value for $k$. In this section we apply the same techniques to the problem of finding the spatiotemporal clusters of $G(t)$.

Let $C_1, \ldots, C_k$, $C_i \subset U$, $i = 1, \ldots, k$, be a partition of $U$ into $k$ disjoint non-empty subsets that minimizes

$$\sum_{i=1}^{k} \mathrm{DNcut}(C_i, \bar{C}) .$$

We denote subsets $C_1, \ldots, C_k$ *spatiotemporal clusters* (or *STclusters*). Let $Q'$, eq.(3), be the infinitesimal generator matrix of a CTRW-D with base walker rate $\gamma$ on $G(t)$. The transition probability matrix of the corresponding embedded Markov chain at the transition points of the walker is

$$P = -\mathrm{diag}(Q')^{-1} Q' + I.$$

Then the rest of the algorithm is a straightforward application of the spectral clustering technique presented in Section 4.1. A trivial consequence of the NCD argument presented in Section 3.2.1 is that, distinct from their static graph counterparts, STclusters are affected by the base walker rate, $\gamma$, with respect to the graph transition rates.

In what follows we present some examples of applications of the methods presented in this work.

# 5. EXAMPLES

In this section we study CTRWs, more specifically CTRW-D, on synthetic and real-world examples. In Section 5.1 we present an example showing the average residence time of CTRW-D when walking a real-world mobile network. We simulate multiple CTRW-D walkers on real-world mobile device traces [14]. In Section 5.2 we look at two examples using our algorithm to perform spatiotemporal clustering. In the first example, a simple ad-hoc network scenario, mobile wireless devices move around in a grid. In the second example we cluster U.S. states in the dynamic graph span by vote roll calls of elected state representatives in the U.S. House of Representatives from 1959 to 2009.

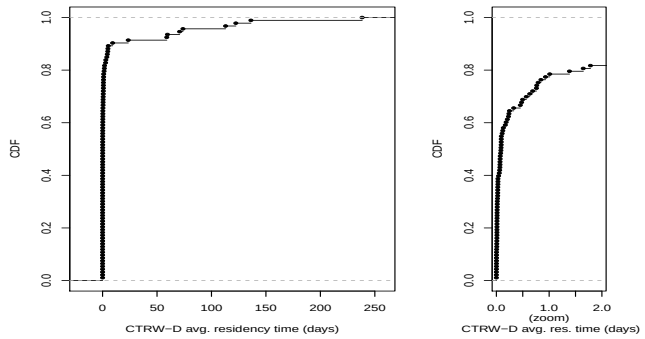


**Figure 7: CDF of average residence times (in days) at each mobile (average over $10,000$ runs). Base CTRW-D walker rate $\gamma = 10^2$.**

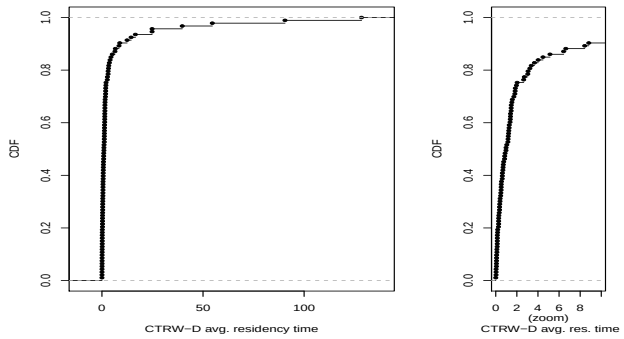


**Figure 8: CDF of average residence times (in days) at each mobile (average over $10,000$ runs). Base CTRW-D walker rate $\gamma = 1$.**

## 5.1 The time a CTRW-D walker spends on nodes: A DTN real-world example

In Section 3 we proved that, given some ergodicity conditions, an infinitely long lived CTRW-D spends the same amount of time at each node of the graph, independent of the graph dynamics. We call the amount of time a walker spends on a node the walker *residence time* on that node. In what follows we study the average *residence time* in a real-world scenario. In our experiment we simulate a CTRW-D on a real-world bluetooth mobility trace (the Reality Mining Dataset [14]) and look at the the average residence times. The trace [14] has bluetooth connectivity data of mobile phones of 94 subjects during 246 days. Note that the trace-driven dynamic graph representing the mobile connections is "highly" disconnected and transient (bluetooth connectivity needs the two mobiles to be within approximately five meters of each other). The average number of contacts per mobile per day is 0.024. As expected, the snapshots of the dynamic graph are mostly disconnected. The median, the mean, and the variance of the time between two consecutive changes in the graph are 6.3 minutes, 2 hours, and 1.8 days, respectively. We also observe that some mobiles become permanently disconnected after interacting with other mobiles for a short period of time.

We simulate a CTRW-D on the trace-driven dynamic graph where we start the CTRW-D at randomly chosen node at time $t = 0$. The base walker rates are $\gamma \in \{10^{-2}, 1, 10^2\}$ ($1/\gamma$ is the average number of days between the times that the walker decides to take a step). Figure 7 plots the CDF of the average walker residence times observed on the graph when $\gamma = 10^2$. The CDF shows that for 70% of nodes the walker spends between zero and $1/2$ day at the node. Although $1/2$ day seem like a long time, recall that the average number of contact per mobile per day is only 0.024. Note that the CDF shows that we are close to the ideal case for most of the nodes: the CDF shaped like a step function (which would happen if the average walker residence times was the same for all nodes). Some nodes have large average residence times becausse the walker tends to get stuck inside nodes (vertices) that get permanently disconnected. In a slow walker ($\gamma = 10^{-2}$) we observe that the CTRW-D gets too few opportunities to sample the dynamic graph and the average residence times are not similar. Figure 8 plots the CDF of the average walker residence times when $\gamma = 1$. Despite the transient and disconnectedness of the dynamic graph, our results, depicted in Figure 8, show that the average time that CTRW-D spends on each mobile is close to constant (averaged over $10,000$ runs). Moreover, the difference in the average residence times of 60% of the nodes is $1/20$th of the average inter-contact time. These are promising results that point to the possibility of applying CTRW-D in scenarios where the graph dynamics are "heavily" transient. Choosing the appropriate value of $\gamma$ for a given graph dynamic is an important open problem and is topic of future work.

In what follows we perform spatiotemporal clustering experiments.

## 5.2 Spatiotemporal clustering

Now we turn our attention to the task of finding spatiotemporal clusters in dynamic graphs. Our main concern is to present examples in which the ground truth is (somewhat) obvious (to the best we can). In Section 5.2.1 we present a naive DTN example where mobiles walk on a lattice. Two mobiles nodes are connected (have an edge between them) if they occupy the same square in the lattice. Mobiles are colored red and blue and spend more time communicating with other mobiles that have the same color. We see that our algorithm, which is oblivious to color and mobile dynamics, can group mobiles by color and dynamics (more details in Section 5.2.1). In Section 5.2.2 we cluster U.S. states according to the evolution of how their state representatives voted in the U.S. House of Representatives from 1959 to 2009. Our algorithm is oblivious to the parties of the state representatives and to the number of political parties in the U.S. system. Our algorithm finds that that U.S. states should be divided into two clusters (which we interpret to represent those states that the two political parties are dominant). The division of states into clusters appear to follow the state political inclinations. For instance, Massachusetts and Kansas are in different clusters.

### *5.2.1 Naive DTN example*

In this section we perform spatiotemporal clustering on a naive DTN model, represented by a dynamic Markov graph.

The objective of this experiment is to see how our technique works on a DTN toy scenario. The DTN is modeled as follows. Consider $N$ wireless mobile agents moving on a $r \times r$ lattice. Agents are colored red or blue and can only move horizontally or vertically. Two or more agents are allowed to occupy the same lattice square at the same time. An agent with color $x$ spends an exponentially distributed amount of time, $1/\mu$ in average, on each square before it moves to an adjacent square. The rate $\mu$ is determined by the existence of other agents with color $x$ in the same square: $\mu = 1/100$ if there other agents with color $x$; $\mu = 1$ otherwise. We are modeling the following behavior: only agents of same color can transmit data to each other but only if they are in the same square. An agent spends more time in a square when transmitting data.

As observers we are oblivious to agent colors or data transmissions. We would like to: (1) identify the same colored agents and (2) determine when data transmissions are taking place. Although these tasks are straightforward given prior knowledge, we are interested in the quality of our spatiotemporal clustering approach when there is no prior knowledge. Using the above model we build a dynamic graph $G(t)$. Agents are represented by nodes. Two nodes have an edge at time $t$ if their corresponding agents are in the same square in the lattice at time $t$. We see that $G(t)$ is a Markov dynamic graph. Because the number of graph snapshots grows exponentially in $N$, we consider an example of $N = 3$ agents in a $3 \times 3$ lattice. Two mobiles are blue, denoted $b_1$ and $b_2$, and one mobile is red, denoted $r$.

Our results show the following spatiotemporal clusters (cluster numbers are arbitrary):

- slow walker ($\gamma = 10^{-4}$): finds three clusters: (cluster 1) the walker is at the agents $b_1$ or $b_2$, (cluster 2) the walker is at agent $r$ and $r$ shares its square with either agents $b_1$ or $b_2$, and (cluster 3) encompasses all other cases.

- fast walker ($\gamma = 10^{4}$): finds nine clusters: (clusters 1 to 7) these clusters contain all STnodes where the walker is not at an isolated node and (clusters 8 and 9) contain all STnodes where the walker is at an isolated node. The difference between clusters 8 in 9 lies in that in cluster 8 $b_1$ and $b_2$ need at least two moves to be in the same square (to get connected) while in cluster 9 $b_1$ and $b_2$ are within one move to occupy the same square.

- medium paced walker ($\gamma = 10^{-1}$): finds two clusters: (cluster 1) this cluster contain all STnodes where at least two agents are within few moves from a corner of the lattice; (cluster 2) contains all STnodes where the agents are within few moves from the opposite corner of the lattice.

We observe that the slow walker clusters STnodes according to the color of the agents, rather than how they move. A fast walker clusters STnodes mostly according to how the agents move in the graph (however agent color still helps to define clusters). A medium paced walker clusters STnodes according to a reference point in the lattice (e.g., one of the corners).

### 5.2.2 *U.S. House of Representatives example*

This second example uses a dataset from roll call voting in the U.S. House of Representative [1]. Our dataset includes all Congresses from the 86th Congress (elected in 1959 during Eseinhower's second term; this is also the first Congress with all 50 states) to the 111th Congress (elected in 2009). Each Congress is a graph snapshot, denoted $G_k$, $k = 86, \ldots, 111$, where $k$ is the Congress number. A node in $G_k$ represents a U.S. state. An edges is added to $G_k$ as follows. Let $a$ and $b$ be two nodes (states) in $G_k$. Consider all possible pairs of tuples between a state representative of $a$ and a state representative of $b$. We compare each tuple by counting the fraction of "Yay" and "Nay" votes of each bill that both state representatives voted the same way. A tuple of state representatives match if they have voted the same way in at least 70% of the bills. Two nodes (states) $a$ and $b$ have an edge if more than 50% of its tuples are a match.

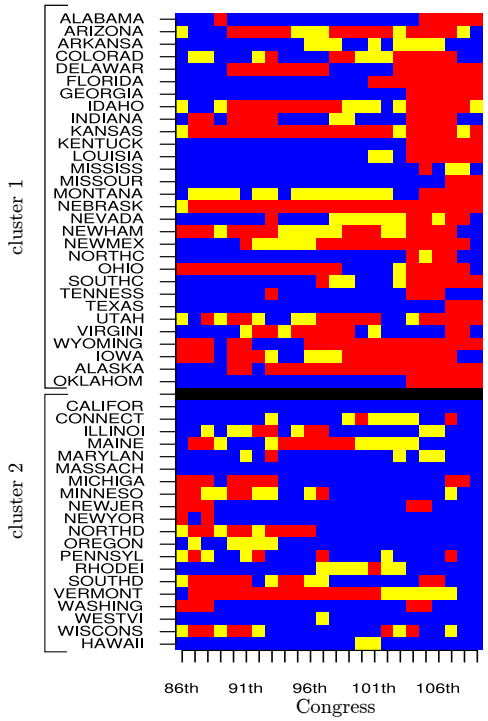


**Figure 9: U.S. states clustered by our algorithm using House roll call votes from 1959 to 2009. The figure shows a $50 \times 24$ matrix with states (nodes) as rows and Congress number (graph snapshot) as columns. Let $R_{ij}$ ($D_{ij}$) be the number of Republican (Democrat) representatives from state $i$ elected in Congress $j$. Element $(i, j)$ of the matrix is colored either red, blue or yellow depending whether $R_{ij} > D_{ij}$, $D_{ij} > R_{ij}$, or $D_{ij} = R_{ij}$, respectively.**

We model the residence time of each Congress is an exponential random variable with rate one (the value of this rate is arbitrary). A graph snapshot, $G_k$, $k = 86, \ldots, 110$, can transition to snapshots $G_{k+1}$ and $G_{k-1}$. Snapshot $G_{87}$ transitions only to $G_{88}$; similarly $G_{111}$ transitions only to

$G_{110}$. The choice of transition rates and the structure of the graph MC is arbitrary, enough just to give a sense of time passing between two consecutive Congresses. In this example we are just interested in the long run clustering of voting patters *per* state. We set the base random walker rate $\gamma = 10^{-4}$. Our algorithm is oblivious to: (1) state representative political party affiliation, (2) the number of political parties, and (3) state political inclinations. Figure 9 show that our algorithm divided the states into two clusters (in agreement with the U.S. political system where there are only two main parties). The results shows that our algorithm seem to have divided states according to their political tendencies. This result may shed light in our political process. States may have Republican or Democratic majorities but the state's true conservative or liberal nature is reflected in how their representatives vote over time. For instance, Arkansas (Bill Clinton's home state) has no Republican majority in the House from 1959 to 2009. However, our algorithm places Arkansas in "cluster 1" (the "Republican" cluster). Arkansas is an interesting example. Since 1960 Republican presidential candidates have best their Democratic opponent 8 times in Arkansas (there were 13 presidential elections between 1959 and 2009 including two races where Bill Clinton won Arkansas). Vermont, placed in "cluster 2", has had a number of Republican majorities in the House but is considered a liberal state according to Wikipedia (and is one of the few states with same-sex marriages, for instance).

# 6. RELATED WORK

Random walks have been widely used to understand and characterize graphs due to its well understood steady state behavior. By leveraging the steady state distribution of random walks, principled mechanisms for characterizing and estimating vertex-related properties can be devised [32, 16, 26, 35, 36].

It follows that random walks can potentially be used to understand and characterize dynamic graphs. In fact, efforts in this direction concerning time-independent dynamic graphs (i.e., each snapshot is independent of the previous) have appeared in the literature [10, 20, 22, 34], mainly in the context of determining upper and lower bounds for the cover time of random walks. More recently, proposals to define time-dependent dynamic graph models as well as characterize random walks in them have also appeared in the literature [3, 11, 2]. However, these efforts have focused on discrete-time dynamic graph models and random walks with the goal of computing the cover time of random walks (in special graph structures [3], in specific dynamic graph models [11], or through numerical evaluations [2]). Our work differs from these in the sense that we consider continuous-time dynamic graph models and continuous-time random walks with the goal of analytically characterizing the steady state behavior of the walker.

Besides understanding their general behavior on dynamic graphs, random walks have also been recently used to characterize properties of vertices in these contexts. One particular application is vertex clustering. Early work in this direction define clusters in a dynamic graph to be connected components in the graph snapshots [5] or assume a static hypergraph [9]. Clustering of truly dynamic graphs models has only recently being investigated [29]. This first approach is based on extending the notion of *community structure* (introduced in [31]) to dynamic graph models and is rather pragmatic, without a principled theoretical foundation. In fact, it is non-trivial to grasp the meaning of spatiotemporal clusters, let alone define metrics that yield intuitive results. Our work focuses on applying continuous-time random walks (for which we know the steady state behavior) to clustering in dynamic graphs in an attempt to better understand clustering in this context.

Finally, random walks have also been used as a mechanism to measure and estimate characteristics of vertices (e.g., fraction of vertices of a particular kind) in large static graphs [16, 26, 36]. More recently, efforts to measure characteristics of vertices in dynamic graphs have also appeared in the literature [40, 35]. However, these are mostly preliminary and exploratory work, indicating potential pitfalls and biases introduced by fast changing dynamic graphs. In contrast, our works is a first step at providing a theoretical foundation for measuring and characterizing vertices using random-walks in a class of dynamic graphs.

# 7. DISCUSSION AND CONCLUSION

Technological progress has brought dynamic networks to play a major hole in society and, as such raising recent attention to understand the fundamental principles that govern their behavior. Despite their importance, very little is known about them. Our work is a first step towards understanding a few basic properties of dynamic networks.

Our work indicates that CTRW yields a principled mechanism for characterizing topological and vertex-related features of dynamic graphs. By considering Markov dynamic graph models we derive closed-form analytical solution to the steady state of CTRW and CTRW-D, which have strikingly different behavior. While the steady state of CTRW is dependent on the walker rate, CTRW-D has predictable behavior which is independent not only of walker, but also graph structure and dynamics. These findings have profound implications for CTRW-based methods for characterizing dynamic graph properties (e.g., sampling). Using CTRW-D we devised a metric for characterizing spatiotemporal clusters, a concept that is still on infancy.

# 8. REFERENCES


[1] `http://voteview.com`.

[2] Utku Günay Acer, Petros Drineas, and Alhussein A. Abouzeid. Random walks in time-graphs. In *Proceedings of the Second International Workshop on Mobile Opportunistic Networking*, MobiOpp'10, pages 93–100, 2010.

[3] C. Avin, M. Koucky, and Z. Lotker. How to explore a fast-changing world. (on the cover time of dynamic graphs). In *Proc. ACM ICALP*, pages 121–132, 2008.

[4] A. Azran and Z. Ghahramani. A new approach to data driven clustering. In *Proc. International Conference on Machine Learning*, 2006.

[5] S. Banerjee and S. Khuller. A clustering scheme for hierarchical control in multi-hop wireless networks. In *IEEE INFOCOM'01*, 2001.

[6] A. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, pages 509–512, October 1999.

[7] C.C. Bilgin and B. Yener. Dynamic network evolution: Models, clustering, anomaly detection. Technical report, RPI Technical Report 08-08, 2008.

[8] N. Bisnik and A.A. Abouzeid. Optimizing random walk search algorithms in p2p networks. *Computer Networks*, 51(6):1499–1514, 2007.

[9] U.V. Catalyurek and C. Aykanat. Hypergraph-partitioning-based decomposition for parallel sparse-matrix vector multiplication. *Parallel and Distributed Systems, IEEE Transactions on*, 10(7):673–693, jul 1999.

[10] Andrea E. F. Clementi, Francesco Pasquale, Angelo Monti, and Riccardo Silvestri. Communication in dynamic radio networks. In *Proc. ACM PODC'07*, pages 205–214. ACM, 2007.

[11] Andrea E.F. Clementi, Claudio Macci, Angelo Monti, Francesco Pasquale, and Riccardo Silvestri. Flooding time in edge-markovian dynamic graphs. In *Proc. ACM PODC'08*, pages 213–222. ACM, 2008.

[12] Pierre-Jacques Courtois. *Decomposability, Queueing and Computer Systems Applications*. Academic Press, 1976.

[13] E. de Souza e Silva and H.R. Gail. Transient Solutions for Markov Chains. In W. Grassmann, editor, *Computational Probability*, pages 43–79. Kluwer Academic Publishers, 2000.

[14] Nathan Eagle, Alex (Sandy) Pentland, and David Lazer. Inferring friendship network structure by using mobile phone data. *PNAS*, 106(36):15274–15278, Sept. 2009.

[15] A. Gautreau, A. Barrat, and M. Barthelemy. Microdynamics in stationary complex networks. *PNAS*, 106:8847–8852, 2009.

[16] M. Gjoka, M. Kurant, C. Butts, and A. Markopoulou. A walk in Facebook: Uniform sampling of users in online social networks. In *Proc. IEEE INFOCOM*, 2010.

[17] C. Gkantsidis and M. Mihail. Hybrid search schemes for unstructured peer-to-peer networks. In *Proc. IEEE INFOCOM*, 2005.

[18] C. Gkantsidis, M. Mihail, and A. Saberi. Random walks in peer-to-peer networks: algorithms and evaluation. *Perform. Eval.*, 63(3):241–263, March 2006.

[19] P. Grindrod and D.J. Higham. Evolving graphs: dynamical models, inverse problems and propagation. *Proc. of the Royal Society A*, 2010.

[20] S.M. Hedetniemi, S.T. Hedetniemi, and A.L.Liestman. A survey of gossiping and broadcasting in communication networks. *Networks*, 18(4):319–349, 1988.

[21] Stratis Ioannidis and Peter Marbach. Absence of evidence as evidence of absence: A simple mechanism for scalable p2p search. In *Proc. of the IEEE INFOCOM'09*, 2009.

[22] D. Kempe and J. Kleinberg. Protocols and impossibility results for gossip-based communication mechanisms. In *Proc. of Symposium on Foundations of Computer Science (FOCS)*, pages 471–480, 2002.

[23] R. Lambiotte. Multi-scale modularity in complex networks. In *Proc. of 8th WiOpt*, April 2010.

[24] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[25] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *Proc. of the 16th international Conference on Supercomputing*, 2002.

[26] L. Massoulié, E. Le Merrer, A.-M. Kermarrec, and A. Ganesh. Peer counting and sampling in overlay networks: random walk methods. In *Proc. of the PODC*, pages 123–132, 2006.

[27] M. Meila and J. Shi. A random walks view of spectral segmentation. In *10th International Workshop on Artificial Intelligence and Statistics*, 2001.

[28] M. Meila and J. Shi. A random walks view of spectral segmentation. In *International Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2001.

[29] Peter J. Mucha, Thomas Richardson, Kevin Macon, Mason A. Porter, and Jukka-Pekka Onnela. Community Structure in Time-Dependent, Multiscale, and Multiplex Networks. *Science*, 328(5980):876–878, 2010.

[30] Boaz Nadler, Stéphane Lafon, Ronald R. Coifman, and Ioannis G. Kevrekidis. Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators. In *in Advances in Neural Information Processing Systems 18*, pages 955–962. MIT Press, 2005.

[31] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

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

[33] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proc. of NIPS*, 2001.

[34] B. Pittel. On spreading a rumor. *SIAM Journal on Applied Mathematics*, 47(1):213–223, 1987.

[35] A. H. Rasti, M. Torkjazi, R. Rejaie, N. Duffield, W. Willinger, and D. Stutzbach. Respondent-driven sampling for characterizing unstructured overlays. In *Proc. of IEEE INFOCOM'09 Mini-Conference*, 2009.

[36] B. Ribeiro and D. Towsley. Estimating and sampling graphs with multidimensional random walks. In *Proc. of the ACM SIGCOMM Internet Measurement Conference*, Nov 2010.

[37] A. Scherrer, P. Borgnat, E. Fleury, J.-L. Guillaume, and C. Robardet. Description and simulation of dynamic mobility networks. *Computer Networks*, 52:2842–2858, Oct. 2008.

[38] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22:888–905, 2000.

[39] Herbert A. Simon and Albert Ando. Aggregation of variables in dynamic systems. *Econometrica*, 29:111–138, 1961.

[40] Daniel Stutzbach, Reza Rejaie, Nick Duffield, Subhabrata Sen, and Walter Willinger. On unbiased sampling for unstructured peer-to-peer networks. *IEEE/ACM Trans. Netw.*, 17:377–390, April 2009.

[41] Marco Zuniga, Chen Avin, and Manfred Hauswirth. Querying dynamic wireless sensor networks with non-revisiting random walks. In *Proc. EWSN 2010*, pages 49–64, 2010.