RWTC for Persistent Community Detection

Abby Leung
School of Computer Science
McGill University
Montreal, QC, Canada
oi.k.leung@mail.mcgill.ca

ABSTRACT

Community detection on graph structures is an important problem in network science and has many crucial applications in numerous fields such as sociology and epidemiology. Real world networks are dynamic, with nodes and edges appearing and disappearing across a timescale. However, common community detection algorithms aggregate these dynamic structures to static graphs. In doing so, vital temporal information is lost. In this project, I propose a clustering algorithm for temporal graphs using short random walk that finds persistent communities while able to preserve important temporal information.

CCS CONCEPTS

• **Networks** → **Network dynamics**; • **Theory of computation** → **Random walks and Markov chains**; • **Applied computing** → *Sociology*.

KEYWORDS

temporal graphs, clustering, random walk, social interaction

ACM Reference Format:
Abby Leung. 2020. RWTC for Persistent Community Detection. In *COMP 400 '20: Project in Computer Science, Winter 2020, Montreal, QC*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION AND MOTIVATION

In recent years, there has been a wide interest in studying and extracting information from temporal networks, such as the evolution of friendship networks and how infectious diseases are spread. From friendship network to web graphs, to the spreading of infectious diseases and information, real life networks are rarely static. Yet most currently existing community detection algorithms require a network to be static or aggregated to a static network. Few clustering algorithms are available for dynamic networks.[6] Inspired by the *WalkTrap* algorithm proposed by Pons and Lapaty[23], I explore the behaviour of a random walker on dynamic graphs and propose a clustering algorithm for finding persistent communities while retaining important temporal information in dynamic networks. This project is organized as follows: the next section outlines some of the previous research done in the areas of community detection and random walk. Section 3 recalls the fundamentals of random walks and temporal graphs, highlighting some important definitions and properties. Section 4 describes the details of the RWTC algorithm. Section 5 provides the setup of the experiments to be run and a brief description of the datasets used in the experiments. Section 6 details and interprets the results of experiments performed using different algorithms on the same datasets. Section 7 discusses the advantages and disadvantages of the RWTC algorithm, as well as future work to be done with RWTC. Lastly, section 10 summarizes the findings of this project and discusses further work that could be done in this area.

COMP 400 '20, Winter 2020, Montreal, QC

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 RELATED WORKS

Graph clustering is a widely studied problem with many previous literature[31]. Most graph clustering algorithms are defined to cluster a given graph into sets of closely connected nodes, commonly by maximizing a measure called modularity. [18], a quality function that measures how well a partition of a network is. These modularity maximizing methods include the *Kernighan–Lin algorithm*[10], *simple node-moving algorithm*[16][18] and *spectral modularity maximization*[16].Modularity maximization algorithms such as *Louvain algorithm*[3] are very popular, but this algorithm only applies to static graphs. Modularity has been extended for temporal graphs where nodes or edges change over time. These methods often assume the clusters also change over time and try to detect their evolution. In this work, we assume that the underlying clusters are persistent and will use the temporal information which is often present to better discover them. This is in between the well studied static community detection and dynamic community detection. The main hypothesis is that by incorporating the temporal information, we will be able to recover the static clusters more precisely. It is worth noting that there are recent efforts to use modularity maximization as a mean to perform community detection on temporal brain graphs by Garcia et al. [6]. Additionally, Mucha et al. [14] proposed a modularity optimization algorithm to detect temporal clusters in the US congress from the years 1789 to 2008.

Other temporal graph clustering approaches include aggregating temporal graphs into static graphs or performing community detection on given snapshots[9] to observe the evolution of clusters. However, by aggregating dynamic graphs into static graphs, important temporal information would be lost. On the other hand, performing community detection on individual snapshots fail to discover clusters that are persistent overtime. In this project, I will investigate the clusters that are consistent over time while retaining temporal information.

There are some existing method of persistent community detection. These methods often involve generalization of popular static clustering algorithms. For example, Jutla et al. [13][14] proposed a *genLouvain*, a generalized Louvain algorithm for persistent community detection on multilayer networks. Similarly, the Louvain algorithm can be applied on the supra-matrix representation of a multilayer network to find persistent communities. [26] Additionally, Li et al. [11] introduced the (θ, τ)-persistent k-core model which includes a temporal graph reduction algorithm to prune the original temporal graph to identify the (θ, τ)-persistent k-cores more efficiently. Other approaches to persistent community detection include statistical model-based algorithm such as the Persistent Community Detection (PCD) algorithm [12] proposed by Liu et al. In this project, I propose a random walk based algorithm.

Random walk, and Marchov process, itself is a widely studied subject in mathematics. Previous work have been done on deriving properties of random walks on temporal graphs and performing numerical simulations [27][2][21][28].

In addition, Rosvall et al. and Pons and Latapsy introduced the *InfoMap*[25] and *Walktrap*[22] algorithms respectively. Both are random walk based graph clustering algorithms. The former utilizes concepts from information theory while the latter uses the characteristics of short random walks. The *Walktrap* algorithm is a major influence for this project. It utilizes the intuition of random walks on a graph tend to get "trapped" into densely connected parts corresponding to communities. Therefore, two nodes from the same community tend to have an approximately equal probability of reaching all other nodes in a fixed number of stamps. Thus, a distance corresponding to this difference of probability is used to determine the similarity of two nodes and hierarchical clustering is performed on this distance.

According to an evaluation done by Orman et al. [19], random walk based algorithms performed better than other clustering algorithms in general. However, both *InfoMap* and *Walktrap* are only available for use on static graphs, therefore, this project aims to bridge the gap between random walk based clustering algorithms and temporal graph clustering.

3 PROBLEM DEFINITIONS

In this section, I will recall important definitions of random walk and temporal graph, define notations used in this project and provide background information needed for the clustering algorithm proposed.

3.1 Temporal Graphs and Random Walk

The following definitions are based on the definitions outlined in [28] and [22].

Definition 3.1. Temporal Graph. Let $G = (G_1, G_2, ..., G_T)$ be a temporal graph of a sequence of snapshots where G_t is a snapshot of G at timestamp t, with $t = 1, ..., T$. Each $G_t = (V_t, E_t)$ consists of a set of nodes V_t and of edges E_t that are present at that timestep.

Let us consider an undirected, unweighted static graph $\Gamma = (V, E)$ with adjacency matrix A, where $A_{ij} = 1$ if an edge e_{ij} exists between i and j, 0 otherwise. A random walk process is defined by a walker that, located on a given vertex i at timestep t, chooses uniformly at random to move to one of its neighbours j at the next timestep[28]. At each timestep, this transition probability is defined as:

$$P_{ij} = \frac{A_{ij}}{d(i)} \tag{1}$$

, where $d(i) = \sum_{j=1}^{n} A_{ij}$. Hence, the transition probability for unweighted graph is

$$P_{ij} = \frac{1}{d(i)}$$

In the case of weighted graphs, the adjacency matrix is replaced with the weight matrix w. Thus the transition probability becomes

$$P_{ij} = \frac{w_{ij}}{k(i)} \tag{2}$$

, where $k(i) = \sum_{j=1}^{n} w_{ij}$. P is defined to be the *probability transition matrix* for the random walk process. Alternatively, P can be obtained from the degree matrix D

$$D_{ij} = \begin{cases} d(i) & i = j \\ 0 & otherwise \end{cases}$$

and the adjacency matrix A,

$$P = D^{-1}A \tag{3}$$

Similarly, in temporal graphs, the *probability transition matrix* is defined to be $P = (p_1, ..., p_n)$, with snapshots $p_t = D_t^{-1}A_t$, $1 \le t \le T$ corresponding to the *probability transition matrix* at timestep t.

Likewise, in weighted graphs, the *probability transition matrix* P can be obtained from the weight matrix W

$$W_{ij} = \begin{cases} k(i) & i = j \\ 0 & otherwise \end{cases}$$

and the adjacency matrix A,

$$P = W^{-1}A \tag{4}$$

In temporal graphs, the *probability transition matrices* $P = (p_1, ..., p_n)$ becomes $p_t = W_t^{-1}A_t$, $1 \le t \le T$ corresponding to the *probability transition matrix* at timestep t.

In this project, I will focus on undirected graphs, both weighted and unweighted.

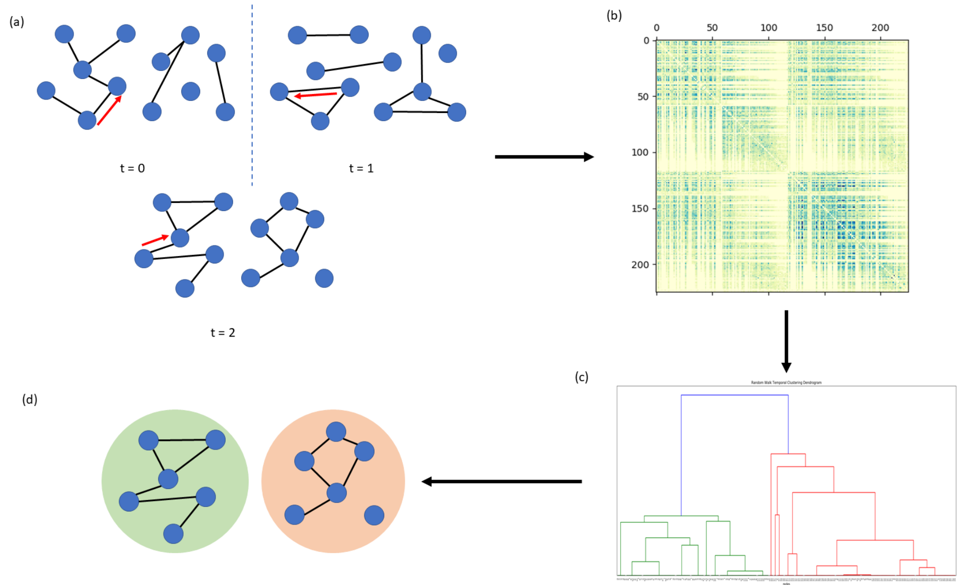

Figure 1: Flowchart detailing the sequence of RWTC algorithm: (a) a random walker walking on a temporal graph with edges appearing and dissapearing, (b) the reachability matrix R is formed, (c) hierarchical clustering is performed on the reachability matrix, (d) two clusters are formed by cutting the dendrogram at the point where multislice modularity is maximized

4 METHODOLOGY

In this section, I outline the method used for community detection in this project. RWTC investigates the "reachability" of node j from a random walker starting at node i for every pair of nodes (i, j). Since communities in graph can be seen as each community having more connections within the community than with the rest of the graph[24], each node is more likely to be reachable by nodes within its own community before being reachable by nodes outside in short random walks. The main intuition is that a random walker starting from an arbitrary node will visit nodes from the same community more often than nodes in a different community. In this project, simulations are used to determine the reachability between nodes.

4.1 Setup and Initialization

For the purpose of this project, it is assumed that the walker walks at the same speed as the rate of the temporal graph evolves. However, it is possible for a random walker to walk faster or slower than the rate of the graph evolution, or at an arbitrary rate.

In particular, three different adjacency matrices are tested for the Congress co-sponsorship network in this project, (1) un-normalizaed adjacency matrix tabulating the number of times each pair of legislators co-sponsored bills,

$$A_{ijt} = \sum_k \gamma_{ijk}$$

where $\gamma_{ijk} = 1$ if legislator i and j cosponsored bill k at Congress t, (2) adjacency matrix normalized by the sum of the total number of bills a pair of legislators co-sponsored,

$$A_{ijt} = \frac{1}{b_{it} + b_{jt}} \sum_k \gamma_{ijk}$$

where b_{it} is the number of bills legislator i co-sponsored at Congress t, and (3) adjacency matrix normalized by the product of the total number of bills a pair of legislators co-sponsored,

$$A_{ijt} = \frac{1}{b_{it} b_{jt}} \sum_k \gamma_{ijk}$$

where b_{it} is the number of bills legislator i co-sponsored at Congress t.

4.2 Computing Reachability

The walker first starts at an arbitrary node i at time $t = 0$. The walker then chooses uniformly at random to move to any of its neighbours, or stay at its current position, at the next timestep. This process is then repeated for a fixed number of steps s. Each visit to node j is recorded in the n x n reachability matrix R. Each R_{ij} records the number of visits to node j a random walker starting from node i made. This process is repeated a fixed number of time.

$$R_{ij} = \sum_k \gamma_{ijk}$$

where $\gamma_{ijk} = 1$ if a walker starting from node i reaches node j at walk k, 0 otherwise. To simplify calculations, the matrix is made to be symmetrical, so a visit to node j from node i would also be recorded as a visit from node i to node j. Therefore making

$$R_{ij} = R_{ji}$$

Because the graph is evolving, it is necessary to consider all cases where the random walker starts at each timestep. Otherwise, vital information regarding the interactions might be lost. For example, nodes that only appear later in the graph will not be captured if the walker always starts at $t = 0$. Thus the reachability matrix then becomes

$$R_{ij} = \sum_t \sum_k \gamma_{ijkt}$$

where $\gamma_{ijkt} = 1$ if a walker starting from node i at time t reaches node j at walk k, 0 otherwise.

4.3 Clustering

After the reachability matrix is computed, the Congress Co- sponsorship network is then normalized in two ways, (1) by the sum of the number of times each legislators is in Congress, the reachability matrix becomes

$$R_{ij} = \frac{1}{b_i + b_j} \sum_t \sum_k \gamma_{ijkt}$$

where b_i is the number of times legislator i in Congress, and (2) the product of the number of times each legislators is in Congress and the reachability matrix becomes

$$R_{ij} = \frac{1}{b_i \dot{b}_j} \sum_t \sum_k \gamma_{ijkt}$$

where b_i is the number of times legislator i in Congress.

Hierarchical clustering will then be performed on the noramlized and un-normalized reachability matrices. The reachability metric is converted to a distance matrix D by taking its reciprocal.

$$D_{ij} = \frac{1}{R_{ij}}$$

for all nodes i, j. After the full dendrogram is formed, the quality of the clusters is evaluated by computing the temporal modularity [14] [15],

$$Q = \frac{1}{2\mu} \sum_{ijlr} \left\{ \left(A_{ijl} - \gamma_l P_{ijl} \right) \delta_{lr} + \delta_{ij} C_{jlr} \right\} \delta \left(g_{il}, g_{jr} \right) \qquad (5)$$

where

- A_{ijl} is the adjacency matrix A for node i and j at time l,
- γ_l is the resolution parameter for module size at time l (default $\gamma_l = 1$),
- $P_{ijl} = \frac{k_{il} k_{jl}}{2m_l}, k_{il} = \sum_j A_{ijl}$ is the null model adjacency matrix at time l, $m_l = \sum_{ij} A_{ijl}$ is the total number of edges at time l,
- C_{jlr} is the resolution parameter for module dynamics that takes binary values $\{0, \omega\}$ indicating the absence (0) or presence (ω) of links between timestamps
- g_{il} is the community of node i at timestamp l
- δ is the Kronecker delta function ($\delta_{ij} = 1$ if $i = j$, 0 otherwise)
- $2\mu = \sum_{il}(k_{il} + c_{il})$, measures the strengths of each node individually in each timestamp by k_{il} and across timestamps by $c_{il} = \sum_r C_{ilr}$

A more detailed discussion of the derivation of the temporal modularity and the effects of changing the parameters γ_l and ω can be found in [14] and [15]. The best partition with the maximal modularity is chosen to be the clusters returned by the algorithm.

5 EXPERIMENT SETUP

In the section, I give a brief description of the three datasets examined and the outline the experiments performed.

5.1 Dataset Descriptions

This project examines 3 datasets, (1) the *primary school temporal network data* [29][7], a un-weighted network, (2) the *US Senate Co-sponsorship network*, a weighted network, and (3) the *US Senate Voting Similarity Nework*, a weighted network of over 10000 nodes across 110 timesteps.

5.1.1 *Primary School Contact Network.*

This dataset contains a temporal contact network between 242 students and teachers in 5 grades and 10 classes over the course of 2 school days. It was collected using RFID proximity-sensing devices worn by participants to record their interactions in 20s intervals.

$$A_{ijt} = \begin{cases} 1 & \text{if student } i, j \text{interacted at time } t \\ 0 & \text{otherwise} \end{cases}$$

The dataset is undirected, unweighted and was originally collected by Stehlé et al.[29] to study how infecious diseases spread within a school setting. In this dataset, the class to which each student belongs is used as the ground truth.

5.1.2 *Legislative Co-Sponsorship Network.*

The Legislative Co- sponsorship network dataset [29] contains a temporal network of the number of times 225 US senators co-sponsored pieces of legislature across 12 Congresses. Each congress is made up of the House of Commons and the Senate. Compared to other social networks, the co-soponsorship network is much more densely connected. The the average distances for both the House and Senate across all Congresses are quite short, ranging from 1.58 to 1.95 for the House and 1.17 to 1.51 for the Senate compared to 4.0 for scientific collaboration network.[17] In the House of Commons, majority of the legislators receive co-sponsorships from 25% of the their fellow legislators while the majority of Senators receives co-sponsorships from 75% of the other Senators, possibly due to its smaller size and the tendency for Senators to be more skilled at making connections. In this dataset, the party affiliation of each legislator is used as the ground truth. Additionally, clusters found in the co-sponsorship datset is also compared to the geographic region corresponding to each lesgislator.

5.1.3 US Senate Voting Similarity Network. The US Senate voting similarity network[30], generated from the House and Senate roll-call data from Voteview.com [1], measures the similarity in voting between individual Senators across 110 2-year Congress sessions. The similarity in voting between each pair of Senators in a given Congress is described in the adjacency matrix, each

$$A_{ij} = \frac{1}{b_{ij}} \sum_k \gamma_{ijk}$$

where γ_{ijk} equals 1 if Senators i and j voted the same on bill k and 0 otherwise, and b_{ij} is the total number of bills on which both legislators voted. In this dataset, the party affiliation of each Senator is used as the ground truth. However, this dataset is quite incomplete and party information of 20% of Senators are missing. When computing the accuracy of the clustering results, nodes that are missing ground truth are removed.

5.2 Baseline

The quality of the RWTC algorithm will be evaluated against the following baseline clustering algorithms,

- Clauset-Newman-Moore greedy modularity maximization algorithm as static graph [5]
- Louvain algorithm as static graph [3]

using metrics such as NMI and ARI. For the purpose of these experiments, the class in which each student belongs is to be treated as the ground truth for the primary school dataset and party affiliation for the co-sponsorhsip and voting similarity datasets. The greedy modularity maximization algorithm and Louvain algorithm are computed on a static graph formed by aggregating the temporal graph at each timestamp.

For the primary school network, each short random walk is of length 5 and the parameters $\gamma_l = 1$ and $\omega = 1$ are used. For the co-sponsorship network, $\gamma_l = \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$ and $\omega = \{0, 0.1, 0.5, 1\}$ are used and only the highest score is reported. Additionally, for all networks, $C_{jlr} = \omega$ for all timesteps r where $r = l + 1$, 0 otherwise, as it is impossible to travel back in time nor is it possible to skip through timesteps. The entire random walk process is repeated 100 times.

5.3 Normalizations

The RWTC will be performed on the two normalized adjacency matrices and the un-noramlized adjacency matrix. In addition, the generated reachability matrix will be normalized as described in Section 4.3. Results from these noramlizations will be compared against each other and the un-noramlized matrices to determine the best suited configurations.

5.4 Different Clustering Algorithms on the RWTC Distance Matrix

Finally, clustering results of hierachical clustering is compared with those of k-medoid clustering and agglomerative complete linkage clustering.

6 RESULTS

In this section, results from the above mentioned experiments are presented. Certain clustering algorithms requires setting various parameters, only the highest score of from the tested combination of parameters is shown.

6.1 Performance of the RWTC Algorithm

The resulting communities detected by random walk temporal clustering, Clauset-Newman-Moore greedy modularity maximization and the Louvain algorithm are evaluated using Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI). [20]

Additionally, a number of normalizations and different configurations of the RWTC algorithm are also evaluated: (1) normalization on the adjacency matrix, (2) normalization on the reachability matrix, and (3) clustering step of RWTC compared to agglomerative clustering and k-medoid clustering.

Shown in *Table 1*, the RWTC algorithm has outperforms all static clustering algorithms, such as the greedy modularity maximization algorithm and the louvain algorithm on the primary school contact network. On the Congress bill Co-sponsorship network, however, the advantage of RWTC is less obvious. Although the performance of the RWTC algorithm is generally on par with the hightest performing clustering algorithms in terms of NMI and ARI. Due the highly collaborative nature of the co-sponsorship network, none of the evaluated perform well in the task of uncovering party information in this network. Because of the size of the voting similarity network, only static clustering methods and RWTC with agglomerative and k-medoid clustering are performed.

In an attempt to further understand the community structure in the co-sponsorship network, the clustering results of the co-sponsorship network is compared to the geographic region of the riding of each legislator.

Comparing the clusters found from RWTC to the geographic region of each legislator, shown in Table 2, shows a lower accuracy. This suggests that party affiliation is still a better indicator of legislator co-sponsorship.

6.2 Normalization

In these sets of experiments, normalizations of the adjacency and reachability matrices of the co-sponsorship network are evaluated against the ground truth of this network. Silhouette score is used to determine how well each of these normalized and un-normalized matrices correspond to the ground truth.

Shown in Table 3, the adjacency matrix normalized by the product of the number of co-sponsored bills is more similar to the ground truth than the un-normalized adjacency matrix. Similarily, the reachability matrix normalized by the product of the number of times in congress is more similar to the un-normalized reachability matrix. However, likely to due the fact that the co-sponsorship network is more densely connected than other social networks,

Dataset	Normalization* Adj.	Reachability	Clustering Method	Evaluation NMI	ARI	no. clusters
Primary School contact	None	None	RWTC	**0.8316**	**0.6991**	8
	None	—	Greedy modularity maximization	0.6582	0.1614	3
	None	—	Louvain	0.8115	0.5732	6
Co-sponsorship	None	None	RWTC	0.3325	0.1046	56
	None	sum	RWTC	0.2802	0.0893	38
	None	prod	RWTC	0.2570	0.1158	237
	sum	None	RWTC	0.3505	0.1184	51
	prod	None	RWTC	0.2955	0.1075	50
	None	None	RWTC + agglomerative	0.0912	0.0194	4
	None	None	RWTC + k-medoid	0.0388	0.0158	3
	None	sum	RWTC + agglomerative	0.0984	0.0159	4
	None	sum	RWTC + k-medoid	0.0445	0.0163	4
	None	prod	RWTC + agglomerative	0.0953	0.0269	4
	None	prod	RWTC + k-medoid	0.0274	0.0183	4
	None	—	Greedy modularity maximization	0.0	0.0	1
	None	—	Louvain	0.0025	-0.0013	2
	sum	—	Greedy modularity maximization	0.3658	-7.9364E-5	224
	sum	—	Louvain	0.0025	-0.0014	3
	prod	—	Greedy modularity maximization	0.3687	0.0	225
	prod	—	Louvain	0.1586	0.1178	3
Voting Similarity	None	None	RWTC + agglomerative	0.0177	0.0003	2
	None	None	RWTC + k-medoid	0.0330	0.0246	43
	None	—	Greedy modularity maximization	0.0542	-0.0403	2
	None	—	Louvain	0.1601	0.0311	6

Table 1: Results from experiments. *prod in adj. indicates normalization of the adjacency matrix by the product of the number of times each pair of legislator co-sponsored bills, and sum indicates normalization by the sum of the number of times each pair of legislator co-sponsored bills. Prod in reachability indicates a noramlization of the reachability matrix by the product of the number of Congress sessions each legislator is present and sum indicates a normalization by the sum of the number of Congress sessions.

Normalization* Adj.	Reachability	Clustering Method	Evaluation NMI	ARI	no. clusters
None	sum	RWTC	0.2065	0.0050	38
None	prod	RWTC	0.1528	0.0089	20

Table 2: Results from using geographic region as the ground truth for Legislative co-sponsorhip network

none of these matrices are highly similar to the ground truth.

As previous results suggest, Table 4 confirms that both adjacency and reachability matrices are more similar to the party affiliations than to the geographic regions of each legislator indicating that party affiliation is a better indicator of bill co-sponsorship than geographic locations.

	Normalization	Silhouette Score**
adjacency	None	-0.1262
	Sum	-0.1247
	Prod	0.0438
reachability	None	-0.0624
	Sum	-0.5641
	Prod	-0.0513

Table 3: Silhouette score of normalized and non-normalized matrices. **Silhouette score for reachability matrices are calculated for the all random walk step sizes, only the the highest score is shown. the adjacency matrices are divided by the sum and product of the number of congress sessions each pairs of legislators are present, only the highest score is shown.

	Normalization	Silhouette Score**
adjacency	None	-0.9421
	Sum	-0.9409
	Prod	-0.2888
reachability	None	-0.0624
	Sum	-0.5641
	Prod	-0.8867

Table 4: Silhouette score of normalized and non-normalized matrices compared to geographic region.

7 DISCUSSION

The RWTC algorithm works very well on the primary school contact network where the partition in the nodes are clear. It has outperformed all static community detection algorithms tested. This confirms the earlier hypothesis that aggregating dynamics graphs into static graphs results in a loss of temporal information vital to detecting persistent communities. However, in the co-sponsorship and voting similarity network where the partition of nodes are less clear, the advantage of preserving temporal information of networks is less apparent. To further investing the advantages and disadvantages of the RWTC algorithm, future work can include experimenting on real world networks that have good community structures and on benchmark graphs.

7.1 Temporal Walktrap Algorithm

One of the influences of this project is the *Walktrap* algorithm proposed by Pons and Lapaty. The *Walktrap* algorithm uses a distance that measures how likely a random walker starting from node i will end up in j at a random walk of a given length. The main idea of this project can be extended to compute temporal communities in the style of the *Walktrap* algorithm. Instead of performing random walk simulations, one can perform hierarchical clustering on a temporal probability matrix [4]

$$P = p^0 \cdot p^1 \cdot p^2 \cdot \ldots \cdot p^k \tag{6}$$

where P^t is the probability transition matrix at each timestamp. With a similar intuition applied, a random walker will more likely to explore its own neighbourhood before moving to another, thus resulting in a higher probability in the temporal probability matrix for short random walks.

7.2 Time Complexity

This temporal random walk algorithm is divided into two parts, the random walk simulation and the dendrogram cut. The temporal random walk part runs in $O(ntsk)$ where n is the number of nodes in the graph, t is the number of timesteps, s is the length of the short random walk and k is the number of times the random walk process is repeated. The dendrogram cut part involves the computation of temporal modularity at every possible dendrogram cut, which equals the number of nodes n. The modularity computation itself is a summation over all pairs of timesteps and all pairs of nodes. The dendrogram cut part runs in $O(n^3t^2)$. However, modularity can be calculated individually within each cluster and then summed up, thus reducing the number of calculations performed.

8 CONCLUSIONS

In this project, I have proposed a new method of temporal community detection algorithm using short random walks. This method is able to capture temporal community information and uses hierarchical clustering to detect persistent communities in networks. Experiments on a real world temporal network demonstrates that this algorithm performs at least as well as popular static community detection algorithms with potential of improved accuracy. However, the temporal nature of this algorithm makes it slow to run which poses a problem if used on large datasets. In addition, the random nature of simulations might lead non-deterministic results. Further work can be done on performing more experiments on datasets with better community structures and extending the *Walktrap* algorithm which produces unique results and finding a more efficient way of computing temporal modularity.

ACKNOWLEDGMENTS

I would like to thank Prof. Rabbany for her guidance throughout this project.

REFERENCES

[1] [n.d.].

[2] Chen Avin, Michal Koucký, and Zvi Lotker. 2018. Cover time and mixing time of random walks on dynamic graphs. *Random Structures & Algorithms* 52, 4 (2018), 576–596. https://doi.org/10.1002/rsa.20752 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/rsa.20752

[3] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008, 10 (oct 2008), P10008. https://doi.org/10.1088/1742-5468/2008/10/p10008

[4] Cheng chi Huang. 1977. Non-homogeneous Markov chains and their applications.

[5] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. 2004. Finding community structure in very large networks. *Physical Review E* 70, 6 (Dec 2004). https://doi.org/10.1103/physreve.70.066111

[6] J. O. Garcia, A. Ashourvan, S. Muldoon, J. M. Vettel, and D. S. Bassett. 2018. Applications of Community Detection Techniques to Brain Graphs: Algorithmic Considerations and Implications for Neural Function. *Proc. IEEE* 106, 5 (May 2018), 846–867. https://doi.org/10.1109/JPROC.2017.2786710

[7] Valerio Gemmetto, Alain Barrat, and Ciro Cattuto. 2014. Mitigation of infectious disease at school: targeted class closure vs school closure. *BMC infectious diseases* 14, 1 (Dec. 2014), 695. https://doi.org/10.1186/PREACCEPT-6851518521414365

[8] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. 2008. Exploring Network Structure, Dynamics, and Function using NetworkX. In *Proceedings of the 7th Python in Science Conference*, Gaël Varoquaux, Travis Vaught, and Jarrod Millman (Eds.). Pasadena, CA USA, 11 – 15.

[9] Jialin He and Duanbing Chen. 2015. A fast algorithm for community detection in temporal network. *Physica A: Statistical Mechanics and its Applications* 429 (2015), 87 – 94. https://doi.org/10.1016/j.physa.2015.02.069

[10] B. W. Kernighan and S. Lin. 1970. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal* 49, 2 (Feb 1970), 291–307. https://doi.org/10.1002/j.1538-7305.1970.tb01770.x

[11] R. Li, J. Su, L. Qin, J. X. Yu, and Q. Dai. 2018. Persistent Community Search in Temporal Networks. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. 797–808.

[12] Siyuan Liu, Shuhui Wang, and Ramayya Krishnan. 2014. Persistent Community Detection in Dynamic Social Networks. In *Advances in Knowledge Discovery and Data Mining*, Vincent S. Tseng, Tu Bao Ho, Zhi-Hua Zhou, Arbee L. P. Chen, and Hung-Yu Kao (Eds.). Springer International Publishing, Cham, 78–89.

[13] Inderjit S. Jutla Lucas G. S. Jeub, Marya Bazzi and Peter J. Mucha. [n.d.]. A generalized Louvain method for community detection implemented in MATLAB.

[14] Peter J. Mucha, Thomas Richardson, Kevin Macon, Mason A. Porter, and Jukka-Pekka Onnela. 2010. Community Structure in Time-Dependent, Multiscale, and Multiplex Networks. *Science* 328, 5980 (2010), 876–878. https://doi.org/10.1126/science.1184819 arXiv:https://science.sciencemag.org/content/328/5980/876.full.pdf

[15] Sarah F. Muldoon. 2017. Modularity in static and dynamic networks. OHBM – Brain Graphs Workshop.

[16] Mark Newman. 2018. *Networks* (2nd ed.). Oxford University Press, Inc., New York, NY, USA.

[17] M. E. J. Newman. 2001. Scientific collaboration networks. I. Network construction and fundamental results. *Phys. Rev. E* 64 (Jun 2001), 016131. Issue 1. https://doi.org/10.1103/PhysRevE.64.016131

[18] M. E. J. Newman. 2006. From the Cover: Modularity and community structure in networks. *Proceedings of the National Academy of Science* 103, 23 (Jun 2006), 8577–8582. https://doi.org/10.1073/pnas.0601602103 arXiv:physics.data-an/physics/0602124

[19] Günce Keziban Orman, Vincent Labatut, and Hocine Cherifi. 2012. Comparative evaluation of community detection algorithms: a topological approach. *Journal of Statistical Mechanics: Theory and Experiment* 2012, 08 (aug 2012), P08001. https://doi.org/10.1088/1742-5468/2012/08/p08001

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[21] Julien Petit, Martin Gueuning, Timoteo Carletti, Ben Lauwens, and Renaud Lambiotte. 2018. Random walk on temporal networks with lasting edges. *Phys. Rev. E* 98 (Nov 2018), 052307. Issue 5. https://doi.org/10.1103/PhysRevE.98.052307

[22] Pascal Pons and Matthieu Latapy. 2005. Computing Communities in Large Networks Using Random Walks. In *Computer and Information Sciences - ISCIS 2005*, pInar Yolum, Tunga Güngör, Fikret Gürgen, and Can Özturan (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 284–293.

[23] Pascal Pons and Matthieu Latapy. 2006. Computing Communities in Large Networks Using Random Walks. *Journal of Graph Algorithms and Applications* 10, 2 (2006), 191–218. https://doi.org/10.7155/jgaa.00124

[24] Filippo Radicchi, Claudio Castellano, Federico Cecconi, Vittorio Loreto, and Domenico Parisi. 2004. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences* 101, 9 (2004), 2658–2663. https://doi.org/10.1073/pnas.0400054101 arXiv:https://www.pnas.org/content/101/9/2658.full.pdf

[25] M. Rosvall, D. Axelsson, and C. T. Bergstrom. 2009. The map equation. *The European Physical Journal Special Topics* 178, 1 (01 Nov 2009), 13–23. https://doi.org/10.1140/epjst/e2010-01179-1

[26] Marta Sarzynska, Elizabeth A. Leicht, Gerardo Chowell, and Mason A. Porter. 2015. Null models for community detection in spatially embedded, temporal networks. *Journal of Complex Networks* 4, 3 (11 2015), 363–406. https://doi.org/10.1093/comnet/cnv027 arXiv:https://academic.oup.com/comnet/article-pdf/4/3/363/6716398/cnv027.pdf

[27] Thomas Sauerwald and Luca Zanetti. 2019. Random Walks on Dynamic Graphs: Mixing Times, HittingTimes, and Return Probabilities. *arXiv e-prints*, Article arXiv:1903.01342 (Mar 2019), arXiv:1903.01342 pages. arXiv:math.PR/1903.01342

[28] Michele Starnini, Andrea Baronchelli, Alain Barrat, and Romualdo Pastor-Satorras. 2012. Random walks on temporal networks. *Phys. Rev. E* 85 (May 2012), 056115. Issue 5. https://doi.org/10.1103/PhysRevE.85.056115

[29] Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-François Pinton, Marco Quaggiotto, Wouter Van den Broeck, Corinne Régis, Bruno Lina, and Philippe Vanhems. 2011. High-Resolution Measurements of Face-to-Face Contact Patterns in a Primary School. *PLOS ONE* 6, 8 (08 2011), e23176. https://doi.org/10.1371/journal.pone.0023176

[30] Andrew Scott Waugh, Liuyi Pei, James H. Fowler, Peter J. Mucha, and Mason A. Porter. 2009. Party Polarization in Congress: A Network Science Approach. arXiv:physics.soc-ph/0907.3509

[31] Zhao Yang, René Algesheimer, and Claudio J. Tessone. 2016. A Comparative Analysis of Community Detection Algorithms on Artificial Networks. *Scientific Reports (Nature Publisher Group)* 6 (08 2016), 30750. https://proxy.library.mcgill.ca/login?url=https://search.proquest.com/docview/1901687755?accountid=12339 Copyright - Copyright Nature Publishing Group Aug 2016; Last updated - 2019-10-21.

A ADDITIONAL INFORMATION ON PRIMARY SCHOOL DATA

This dataset contains a temporal contact network between students and teachers in 5 grades and 10 classes over the course of 2 school days. The dataset is undirected, unweighted and was originally collected by Stehlé et al. to study how infecious diseases spread within a school setting. An overview of the dataset can be found in *Table 5* [29].

Table 5: Distribution of Primary School Participants

Class	No. Participants (Day 1)	No. Participants (Day 2)
1A	22	23
1B	25	25
2A	22	23
2B	25	26
3A	23	23
3B	21	21
4A	21	21
4B	22	22
5A	22	21
5B	23	23
Teachers	10	10
Total	242	

Table 6: Summary of Primary School Student and Teacher Interactions

	No. Interactions
Average	1,039
Same class	91,265 (72.6%)
Different class	34,508 (27.4%)

Most interactions, 72.6%, are between individuals from the same class, as shown in *Table 6* with upper year students (4th and 5th grade) more likely to interact with each other than with lower year (1st to 3rd grade) students, and vice versa, as shown in *Figure 2*[29]. It is worth noting that though teachers are considered to be in their own "class", they interact mostly with students of various classes and rarely with each other.

Figure 3 [29] displays the distribution of the total amount of time, w_{ij}, as well the duration of contacts. Stehlé et al. found that most interactions between individuals are short, with an average duration of 33 seconds. 88% of the contacts last less than one minute while 0.2% of the contacts last more than 5 minutes.

Similarly, most cumulated contact durations between two individuals are short, but each student interacts with more than half of their peers. 64% of the pairs of students interacted for less than 2 minutes on the same day, 9% interacted for more than 10 minutes and 0.38% more than 1 hour. The average amount of time spent by

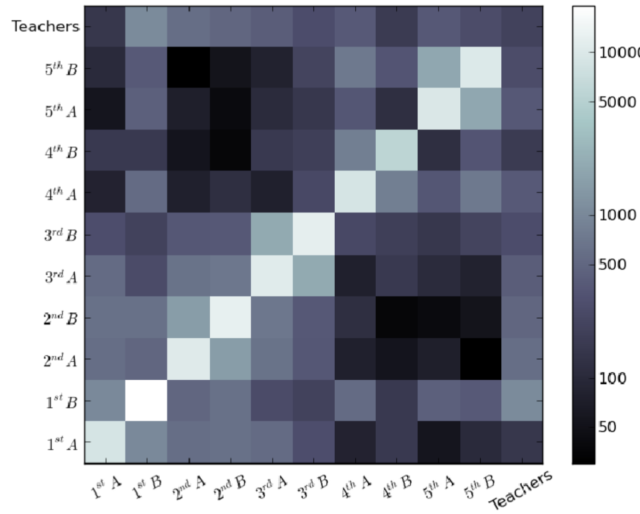

Figure 2: Contact matrix of students and teachers between classes

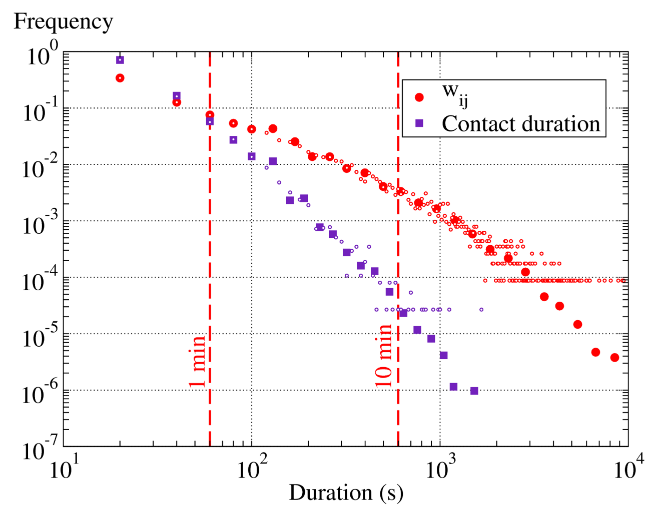

Figure 3: Contact duration between primary school students and teachers

two students in face-to-face proximity in one day is 207 seconds (3 min 27 s) for day 1, and 236 seconds (3 min 56 s) for day 2. This is important as highly infectious diseases can be spread easily even if the students only interacted for a short period of time.

The degree distribution of the primary school dataset does not appear to follow a power law distribution as shown in *Figure 4*

B ADDITIONAL RESULTS ON THE PRIMARY SCHOOL DATASET

In this section, results from the the experiments will be presented along with a visualization of the optimal communities discovered

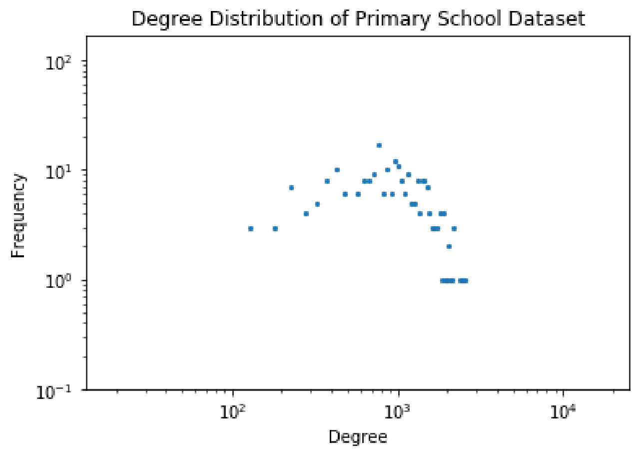

Figure 4: Degree Distribution of primary school dataset

by each algorithm.

B.1 Random Walk Temporal Clustering

After running the simulation and performing hierarchical clustering, the dendrogram shown in *Figure* ?? is formed. The cut of the dendrogram is determined by computing the temporal modularity defined in *Equation 5* and finding the cut that results in the maximal modularity in the graph. The temporal modularity value for clusters formed at each cut of the dendrogram shown in *Figure 5*.

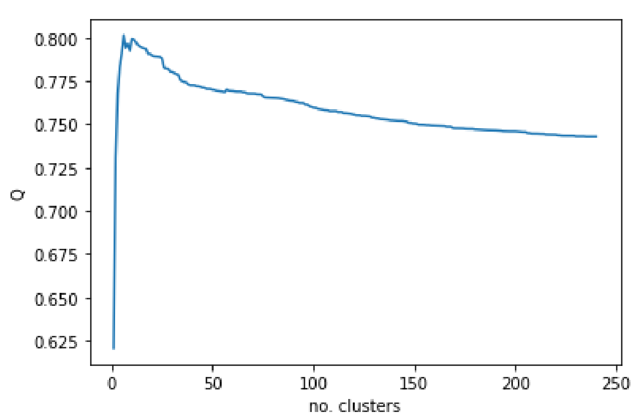

Figure 5: Modularity corresponding to number of communities

Figure 6 indicates the best partition consists of six distinct communities.

Figure 7 illustrates the optimal community structure found by the random walk clustering algorithm.

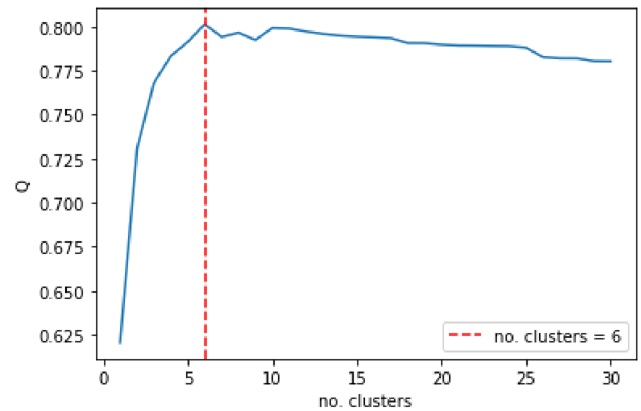

Figure 6: Enlarged portion of *Figure 5* from one community to thirty communities. A partition of 6 communities result in the highest temporal modularity.

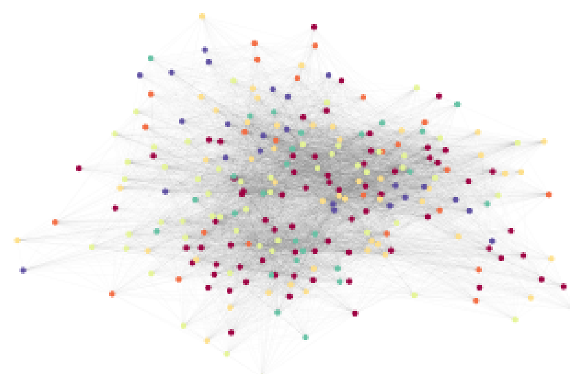

Figure 7: Visualization of random walk clustering results. Random walk clustering found 6 clusters with a NMI score of 0.8115 and ARI score of 0.5732.

B.2 Clauset-Newman-Moore Greedy Modularity Maximization Algorithm

The Clauset-Newman-Moore Greedy Modularity Maximization Algorithm is a hierarchical agglomeration algorithm for detecting community structure in static graphs that begins with each node in its own community and joins the pair of communities that most increases modularity until no such pair exists.[5][8] Applying the Greedy Modularity Maximization algorithm on the Primary School dataset produces three distinct communities, a visual representation of the clusters found is shown in *Figure 8*.

B.3 Louvain Algorithm

The Louvain algorithm is a community detection algorithm on static graphs. It consists of two phases: phase one where modularity is optimized by allowing only local changes of communities;

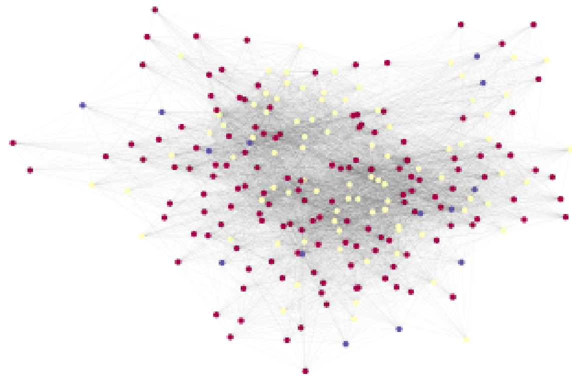

Figure 8: Visualization of Clauset-Newman-Moore greedy modularity maximization clustering results. The greedy modularity maximization clustering algorithm found 3 clusters with a NMI score of 0.4894 and ARI score of 0.1614.

phase two where the found communities are aggregated in order to build a new network of communities. The process is then repeated iteratively until no increase of modularity is possible.[3] A visualization of the clusters found by the Louvain algorithm can be found in *Figure 9*.

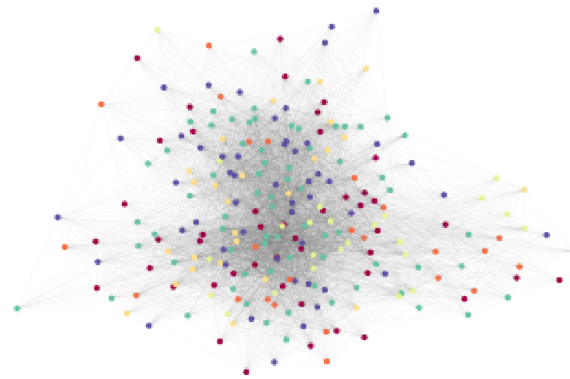

Figure 9: Visualization of Louvain clustering results. Louvain algorithm found 6 clusters with a NMI score of 0.8115 and ARI score of 0.5732.