

The content correlation of multiple streaming edges

Michel de Rougemont¹ Guillaume Vimont¹

¹ University Paris II, CNRS-IRIF
December 27, 2018

Abstract

We study how to detect clusters in a graph defined by a stream of edges, without storing the entire graph. We extend the approach to dynamic graphs defined by the most recent edges of the stream and to several streams. The *content correlation* of two streams $\rho(t)$ is the Jaccard similarity of their clusters in the windows before time t . We propose a simple and efficient method to approximate this correlation online and show that for dynamic random graphs which follow a power law degree distribution, we can guarantee a good approximation. As an application, we follow Twitter streams and compute their content correlations online. We then propose a *search by correlation* where answers to sets of keywords are entirely based on the small correlations of the streams. Answers are ordered by the correlations, and explanations can be traced with the stored clusters.

Keywords: Streaming algorithms, Dynamic graphs, Clustering, Approximation

1 Introduction

Consider a stream of edges of a graph which follows a power law degree distribution. Sliding windows define dynamic graphs G_t and we select each edge with a uniform probability in each window. There are several techniques which generalize the original Reservoir sampling [12] for a fixed window, to dynamic windows. At any given time t , we have a Reservoir which keeps k edges among the possible m edges which have been read ($m \gg k$), and each edge has the same probability k/m to be chosen in the Reservoir. If the cluster S and the Reservoir size k are large enough, a large connected

component will appear in the Reservoir. The random edges of S are taken with the same probability $p = k/m$, i.e. follow the Erdős-Renyi model $G(n, p)$ and the giant component in S occurs if $p = k/m > 1/n = 1/|S|$, the classical phase transition probability. In the dynamic case, we will observe changes in the communities as new communities appear and old communities disappear. At discrete times t_i we store only the large connected components of the Reservoirs.

We study the correlation between multiple streams of graph edges and define the *content correlation* ρ of two streams of edges, based on the Jaccard similarity of their clusters and extend it to $\rho(t)$ on the dynamic graphs G_t . We provide an analysis of dynamic random graphs which follow a power law degree distribution, based on the Configuration Model [10]. We give sufficient conditions to detect clusters depending on their size, the Reservoir size k and the length of the observation. We then approximate the content correlation with an online algorithm, i.e. estimate the correlation $\rho(t_i)$ at discrete times t_i .

Streams of graph edges are ubiquitous, in particular in social networks where the graphs have a degree distribution close to a power law, a small diameter and clusters (communities) which evolve in time. Twitter streams are defined by some tags and generate a stream of edges of large dynamic graphs. We estimate the correlations between multiple streams and obtain a correlation matrix A , from which we build a phylogeny tree. We then introduce a *Search by correlation*: given some tags, we find the most correlated tags which depend on the history of the clusters and on the phylogeny tree. Our main results are:

- We propose an Algorithm which detects large static clusters in a graph which follows a power law degree distribution, with high probability, using only a few edge samples (theorem 1) and its extension to dynamic graphs (theorem 2),
- We present an online algorithm for $\rho(t)$. For two dynamic graphs with clusters S and S' whose Jaccard similarity is ρ^* , the correlation $\rho(t)$ is close to ρ^* (theorem 3).

We estimated the content correlations of 4 Twitter streams (approximately 2.10^6 edges) over 24h online and built the closest phylogeny tree. The *Search by correlation* illustrates the technique which uses the correlations, a phylogeny tree and the stored clusters. In the second section, we define the framework of dynamic graphs defined by a stream of edges and introduce the notion of content correlation of two streams. In the third section, we set a

model of random dynamic graphs where we can guarantee the approximation of the correlation. In the fourth section, we present our analysis of multiple Twitter channels and their correlations. In the fifth section, we introduce the Search by correlation, using a phylogeny tree built from the correlation matrix of the streams.

2 Dynamic graphs in a window of streaming edges

Let $e_1, e_2, \dots, e_i, \dots$ be a stream of edges where each $e_i = (u, v)$. It defines a graph $G = (V, E)$ where V is the set of nodes and $E \subseteq V^2$ is the set of edges: we allow self-loops and multi edges and assume that the graph is symmetric. In the *window model* we isolate the most recent edges at some discrete t_1, t_2, \dots . There are two models of sliding windows: the most recent τ edges or the most recent edges in some fixed time interval τ . If the rate (the number of edges per time unit) of the stream is fixed, both models coincide. It is not the case in practice, as the rate fluctuates within a factor 2 at any given time. We take the second model, i.e. keep the length of the window τ , hence $t_1 = \tau$ and each $t_i = \tau + \lambda \cdot (i - 1)$ for $i > 1$ and $\lambda < \tau$ determines a window of length τ and a graph G_i defined by the edges in the window or time interval $[t_i - \tau, t_i]$. The number of edges in a window may increase or decrease and reflects the increasing or decreasing rates of a stream. Consecutive windows overlap within a factor τ/λ , about 50% in the experiments. In practice, $\tau = 60$ mins and $\lambda = 30$ mins. The graphs G_{i+1} and G_i share many edges: old edges of G_i are removed and new edges are added to G_{i+1} . Social graphs have a specific structure, a specific degree distribution (power law), a small diameter and some dense clusters. The dynamic random graphs introduced in the next section satisfy these conditions.

There are several definitions of a cluster or community or dense subgraph. We consider a cluster of domain S as a *maximal dense subgraph* which depends on a parameter γ . Let $\gamma \leq 1$ and let $E(S)$ be the multiset of internal edges i.e. edges $e = (u, v)$ where $u, v \in S$. A γ -cluster is maximal subset S such that $E(S) \geq \gamma \cdot |S|^2$. There are several other possible definitions of clusters which capture the high internal global density, and there are many algorithms to detect such clusters in a static graph. We are mainly interested in the approximate detection of clusters in the dynamic case, without storing the whole graph.

2.1 Detecting clusters in a stream of edges

If the entire graph is known, there are several classical techniques to approximate communities. For $\gamma = 1$, the problem is known as *Maxclique*, which is NP-hard. In our framework, we do not store the entire graph but only a few edges, and will only approximate the communities. We take a uniform sampling of the edges¹ for each window of the stream i which defines $G_i(t)$, and keep k samples in Reservoirs $R_i(t)$ for a fixed size k . There are several techniques to build such dynamic Reservoirs [4].

The sampling method we propose is not new, it is one of the sampling methods in [3]. Its analysis for dynamic graphs which follow a power law degree distribution is one of the central points of this paper. If a cluster is large, there will be a large connected component in the Reservoir as a witness. We ignore all the small connected components of the Reservoir and only store in a database the large ones. In a typical experiment $k = 400$ whereas we read 10^4 edges in a window. We ignore all the components of size less than 10, a threshold value. For a graph G_i , there could be several large clusters $C_{i,j}$ or there could be none. For a stream $G_i(t)$, we write $C_{i,j}(t)$ for the j -th cluster of the stream i at time t .

2.2 Correlations

The classical correlation, also called the *Pearson correlation* $p(X, Y)$ of two random variables X, Y of mean μ and standard deviation σ is $\frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \cdot \sigma_Y}$. How could it be extended to graphs? One could choose some statistics on the graphs and take the correlations between the statistical parameters. Social graphs have however very similar statistics and yet the core of the information seems to hide in the structure of their clusters. Given two graphs G_1 and G_2 , a first approach to their content correlation would be to consider the Jaccard similarity² $J(V_1, V_2)$ on the domains of the two graphs. It has several drawbacks: it is independent of the structures of the graphs, it is very sensitive to noise, nodes connected with one or few edges and it is not well adapted when the sizes are very different. It also requires to store the entire graphs.

We propose instead the following approach: we first estimate the dense components (clusters or communities) using the uniform sampling on the

¹Notice that such a uniform sampling on the edges is equivalent to a sampling of the nodes proportionally to their degrees.

²The Jaccard similarity or Index between two sets A and B is $J(A, B) = |A \cap B| / |A \cup B|$. The Jaccard distance is $1 - J(A, B)$.

edges in the sliding windows and apply the Jaccard similarity only to these large dense components. It exploits the structure of the graphs, is insensitive to noise and adapts well when the graphs have different sizes. Let $C_i = \bigcup_j C_{i,j}$ be the set of clusters of the graph G_i , for $i = 1$ or 2 . The *correlation of two graphs* $\rho = J(C_1, C_2)$. This definition is $\rho(t_1)$ for the first window or for two static graphs. For two dynamic streams $G_1(t)$ and $G_2(t)$ which share a time scale, we generalize C_i to $C_i(t) = \bigcup_{t' \leq t} \bigcup_j C_{i,j}(t')$. The *correlation of two graph streams* $G_1(t)$ and $G_2(t)$ is $\rho(t) = J(C_1(t), C_2(t))$. We can refine the correlation and define an *amortized correlation* $\rho_a(t)$, to give more importance to the recent components. In the next section, we give an algorithmic solution for graphs presented as streams of edges which scales when we consider dynamic graphs.

2.3 What is stored over time

At some discrete times t_1, t_2, \dots , we store the large connected components of the Reservoirs R_t . There could be none. We use a NoSQL database, with 4 (Key, Values) tables where the key is always a tag (@x or #y) and the Values store the clusters nodes. Notice that a stream is identified by a tag (or a set of tags) and a cluster is also identified by a tag, its node of highest degree.

- *Stream(tag, list(cluster, timestamp))* is the table which provides the most recent clusters of a stream,
- *Cluster(tag, list(stream, timestamp, list(high-degree nodes), list(nodes, degree)))* is the table which provides the list of high-degree nodes and the list of nodes with their degree, in a given cluster,
- *Nodes(tag, list(stream, cluster, timestamp))* is the table which provides for each node the list of streams, clusters and timestamps where the node appears,
- *Correlation((tag1, tag2), list(value, timestamp))* is the table which provides for each pair of streams (tag1, tag2) the different correlation values $\rho(t)$.

2.4 Other approaches

There are many other approaches to detect clusters in streams of graphs edges. The *dynamic graphs algorithms* community studies the compromise between update and query time in the worst case. The *graph streaming approach* [8] emphasizes the space complexity in the worst case and in particular for the window model. The *network sampling approach* such as [3] does consider the uniform sampling on the edges but there is no analysis for the detection of clusters in dynamic graphs. The detection of a *planted clique* is a classical problem [6], hard when the clique size is for example $O(\sqrt{n}/2)$ in the worst

case. The *graph mining community* [2, 1, 13] studies the detection of clusters when the graphs are entirely known.

In our approach, we only consider classes of graphs which follow a power law degree distribution, and study approximate algorithms for the detection of dynamic γ -cliques in the window model using only small Reservoirs.

3 Deciding properties and correlations in dynamic random models

We define a model of dynamic random graphs $G(t)$ which may or may not have clusters and follow a degree distribution using the Configuration Model. Temporal Logic is a framework to decide temporal properties of the dynamic graphs G_t . Let P be a graph property such as Connectivity, or the existence of a γ -cluster of size at least 10. A typical temporal property is $\diamond P$ stating that there exists a t such that $G_t \models P$ or $\square P$ stating that for all t , $G_t \models P$. In this section, we show that the algorithmic approach guarantees a good approximation of the correlation $\rho(t)$ with high probability.

3.1 Dynamic Random graphs

The classical Erdős-Renyi model $G(n, p)$ [5], generates random graphs with n nodes and edges are taken independently with probability p where $0 < p < 1$. The degree distribution is close to a gaussian centered on $n.p$. Most of the social graphs have a degree distribution \mathcal{D} close to a power law, such as a Zipfian distribution where $Prob[d = j] = c/j^2$, where d is the degree of a node. In this case, the maximum degree is $d_{max} = O(\sqrt{n})$. The Configuration Model for \mathcal{D} and a graph with n nodes enumerates each node u with d half-edges (stubs) and takes a symmetric random matching π between two stubs, for example with a uniform permutation such that $\pi(i) \neq i$. All the possible graphs are obtained with a distribution close to the uniform distribution.

A classical study is to find sufficient conditions so that a random graph has a *giant component*, i.e. of size $O(n)$ for a graph of size n . In the Erdős-Renyi model $G(n, p)$, it requires that $p > 1/n$, and in the Configuration Model it requires that $\mathbb{E}[\mathcal{D}^2] - 2\mathbb{E}[\mathcal{D}] > 0$ as proved in [9], which is realized for the Zipfian distribution. There is a phase transition for both models. There are several possible extensions to dynamic random graphs. In our model, the Dynamics is exogenous and at any time chooses between the Uniform and the Concentrated Dynamics.

3.1.1 Uniform Dynamics

we generalize the Configuration Model in a dynamic setting. Remove $q \geq 2$ random edges, uniformly on the set of edges of G , freeing $2 \cdot q$ stubs. Generate a new uniform matching on these hubs to obtain G' . The distribution of random graphs stays uniform.

3.1.2 Concentrated Dynamics

a typical graph generated by the Uniform Dynamics is not likely to have a large cluster. The **S -concentrated Dynamics** fixes some a subset S among the nodes of high degree. Remove $q \geq 2$ edges, uniformly on the set of edges of G , freeing $2 \cdot q$ stubs, as before. A stub is in S if its origin or extremity is in S . With probability 80%, match the stubs in S uniformly in S . With probability 20%, match the stubs in S uniformly in $V - S$. This dynamics will concentrate edges in S and will create a γ -cluster after a few iterations with high probability, assuming the degree distribution is a power law. The distribution of graphs with a γ -cluster stays also uniform.

3.1.3 General Dynamics

a general Dynamics is a function which chooses at any given time, one of the two strategies: either a Uniform Dynamics or some S -concentrated Dynamics for a fixed S . An example is the **Step Dynamics**: apply the Uniform Dynamics first, then switch to the S -dynamics for a time period Δ , and switch back to the Uniform Dynamics. In our setting, the Dynamics depends on some external information, which we try to approximately recover. Notice that during the Uniform Dynamics phase, there are no large components and we store nothing. For the step phase, we store some components which will approximate S . More complex strategies could involve several clusters S_1 and S_2 which may or may not intersect.

3.2 Deciding a static property: there is a large γ -cluster

Let R be the Reservoir of size k after we read m edges e_1, e_2, \dots, e_m . In this simple case, we first fill the Reservoir with e_1, e_2, \dots, e_k . For $i > k$, we decide to keep e_i with probability k/i and if we keep e_i , we remove one of the edges (with probability $1/k$) to make room for e_i . Each edge e_i has then probability k/m to be in the Reservoir, i.e. uniform.

The probabilistic space Ω is determined by the choices taken at every step by the Reservoir sampling. Consider a clique S in the graph: its image in the

Reservoir is the set G_S of internal edges $e = (u, v)$ in the Reservoir, where $u, v \in S$. Each edge of the clique S is selected with constant probability k/m , so we are in the case of the Erdős-Renyi model $G(n, p)$ where $n = |S|$ and $p = k/m$. We know that the phase transition occurs at $p = 1/n$, i.e. there is a giant component if $p > 1/n$ and the graph is connected if $p \geq \log n/n$.

In the case of a γ -clusters S associated with the S -concentrated Dynamics, the phase transition occurs at $p = 1/\gamma.n$. Let V_S be the set of nodes of the giant component G_S whose nodes are in S . As it is customary for approximate algorithms, we write $\text{Prob}_\Omega[\text{Condition}] \geq 1 - \delta$ to say that the Condition is true with high probability.

Lemma 1 *For m large enough, there exists $\alpha = O(\log n)$ and δ such that if $|S| \geq m/\gamma.k$ in the concentrated Dynamics, then $\text{Prob}_\Omega[|V_S| > \alpha] \geq 1 - \delta$.*

Proof : If S is almost a clique, i.e. a γ -cluster, then the phase transition occurs at $p = 1/\gamma.|S|$. Hence if $p > 1/\gamma.|S|$, there is a giant component of size larger than a constant times $|S|$, say $|S|/2$ with high probability $1 - \delta$. As the probability of the edges is k/m , it occurs if $|S| \geq m/\gamma.k$. Hence for m large enough, there exists $\alpha = O(\log n)$ such that $\text{Prob}_\Omega[|V_S| > \alpha] \geq 1 - \delta$.

In order to decide the graph property P : *there is a large γ -cluster*, consider this simple algorithm.

Static Cluster detection Algorithm 1: let C be the largest connected component of the Reservoir R . If $|C| \geq \alpha$ then Accept, else Reject.

Theorem 1 *If $|S| \geq m/\gamma.k$ for the concentrated Dynamics, then:*

$$\text{Prob}_\Omega[\text{Algorithm 1 Accepts}] \geq 1 - \delta$$

and for the uniform Dynamics:

$$\text{Prob}_\Omega[\text{Algorithm 1 Rejects}] \geq 1 - \delta.$$

Proof : If $|S| \geq m/\gamma.k$ for the concentrated Dynamics, Lemma 1 states that $|V_S| > \alpha$ with high probability, hence as $V_S \subseteq C$, the condition $|C| \geq \alpha$ is true with high probability hence $\text{Prob}_\Omega[\text{Algorithm 1 Accepts}] \geq 1 - \delta$. For the Uniform Dynamics ($|S| = 0$), [9] shows that the largest connected component has size $O(\log n)$. Hence $\text{Prob}_\Omega[\text{Algorithm 1 Rejects}] \geq 1 - \delta$.

Notice that $m = c_1.n.\log n$, as the average degree in a power law is $c_1.\log n$. If $k = \sqrt{c_1.n}.\log n$ and $|S| \geq m/\gamma.k = \sqrt{c_1.n}/\gamma$, it satisfies the condition and it can be realized with the nodes of high degree.

3.3 Deciding a dynamic property: $\diamond P$

Let P be the previous property: is there a γ -cluster? How do we decide $\diamond P$? Consider the step strategy of length $\Delta > \tau$. When we switch strategy

at time t_1 there is a delay until S is a γ -cluster and symmetrically the same delay when we switch again at time $t_2 > t_1$. The probabilistic space Ω_t is now much larger.

Dynamic Cluster detection Algorithm 2: let C_i be the largest connected component of a dynamic Reservoir R_i at time t_i . If there is an i such that $|C_i| \geq \alpha$, then Accept, else Reject.

We can still distinguish between the Uniform and the S -concentrated Dynamics, if S is large enough. Let $m(t)$ be the number of edges in the window at time t . Let $G(t)$ be a graph defined by a stream of $m(t)$ edges following a power law \mathcal{D} .

Theorem 2 *For the step Dynamics of length Δ and $t > t_2$, if $|S| \geq m(t)/\gamma.k$, then $\text{Prob}_\Omega[A_2 \text{ Accepts}] \geq 1 - \delta^{\Delta/\tau}$ For the Uniform Dynamics $\text{Prob}_\Omega[A_2 \text{ Rejects}] \geq (1 - \delta)^{\Delta/\lambda}$.*

Proof : For each window, we can apply theorem 1 and there are Δ/τ independent windows. If $|S| \geq m/\gamma.k$ for the concentrated Dynamics, the error probability is smaller than the error made for Δ/τ independent windows, which is $\delta^{\Delta/\tau}$. Hence $\text{Prob}_\Omega[\text{Algorithm 2 Accepts}] \geq 1 - \delta^{\Delta/\tau}$. For the Uniform Dynamics (equivalent to $|S| = 0$), the algorithm needs to be correct at each Δ/λ step. Hence $\text{Prob}_\Omega[\text{Algorithm 2 Rejects}] \geq (1 - \delta)^{\Delta/\lambda}$.

The probability to accept for the S concentrated Dynamics is amplified whereas the probability to reject for the Uniform Dynamics decreases. One single error generates a global error. Clearly, we could also estimate Δ , for step strategies with similar techniques.

3.4 Correlation between two streams

Suppose we have two streams $G_1(t)$ and $G_2(t)$ which share the same clock. Suppose that $G_1(t)$ is a step strategy Δ_1 on a cluster S_1 and $G_2(t)$ is a step strategy Δ_2 on a cluster S_2 . Let $\rho^* = J(S_1, S_2)$. How good is the estimation of their correlation? Let $C_i(t) = \bigcup_{t' \leq t} \bigcup_j C_{i,j}(t')$ be the set of large clusters $C_{i,j}(t')$ at time $t' \leq t$ of the graph G_i , for $i = 1$ or 2 . Consider the following online algorithm to compute $\rho(t)$:

Online Algorithm 3 for $\rho(t)$. At time $t + \lambda$, compute the increase δ_i in size of $C_i(t + \lambda)$ for $i = 1, 2$ from $C_i(t)$, and δ' the increase in size of $C_1(t + \lambda) \cap C_2(t + \lambda)$. Suppose $\rho(t) = I/U$ where $I = |C_1(t) \cap C_2(t)|$ and $U = |C_1(t) \cup C_2(t)|$. Then: $\rho(t + \lambda) = \rho(t) + \frac{U \cdot \delta' - I \cdot (\delta_1 + \delta_2)}{U \cdot (U + \delta_1 + \delta_2)}$.

A simple computation shows that $\rho(t + \lambda) = \frac{I + \delta'}{U + \delta_1 + \delta_2}$, i.e. the correct definition. The δ_i, δ' are computed by standard operations on sets.

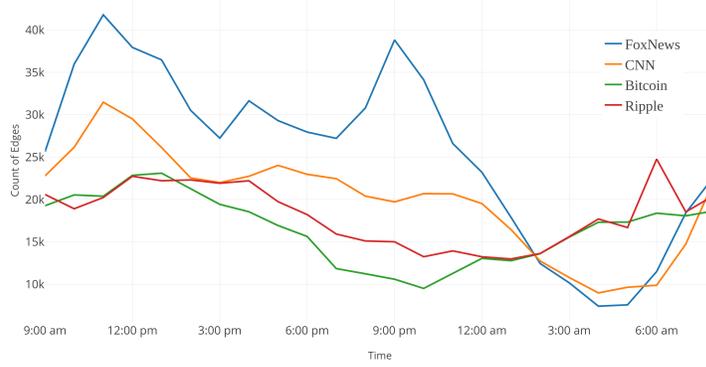


Figure 1: Number of edges in 1h windows, for 4 streams during 24h

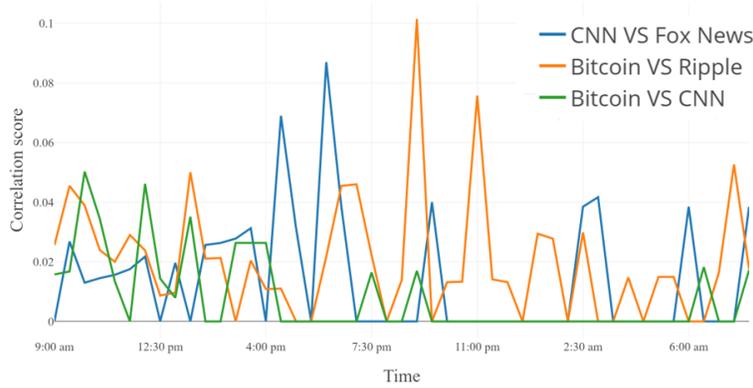


Figure 2: Online content correlation for 24h

Theorem 3 Let $G_1(t)$ and $G_2(t)$ be two step strategies before time t on two clusters such that $|S_i| \geq m/\gamma.k$ for $i = 1, 2$. Then $\text{Prob}_{\Omega_t}[|\rho(t) - \rho^*| \leq \varepsilon] \geq 1 - \delta$.

Proof : After the first observed step, for example on S_1 , Lemma 1 indicates that V_{S_1} is already some approximation of S_1 . After Δ_1/τ independent trials, $V_1 = \bigcup_i V_{S_1,i}$ will be a good approximation of S_1 . Similarly for S_2 and therefore $\rho(t) = J(V_1, V_2)$ will (ε, δ) approximate ρ^* .

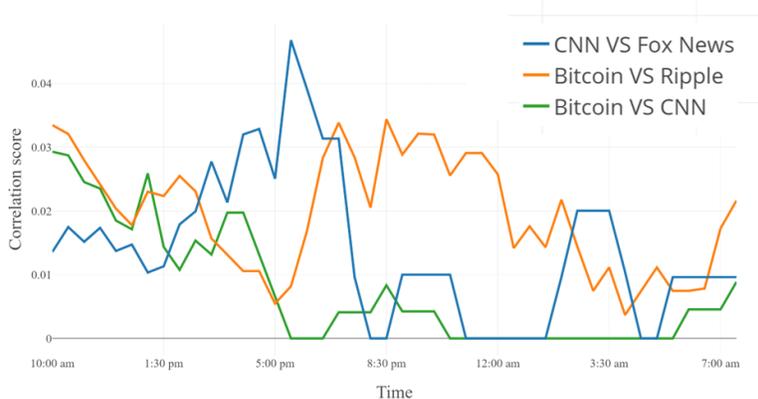


Figure 3: Online averaged content correlation for 24h

4 Twitter streams

Given a set of tags such as #CNN or #Bitcoin, Twitter provides a stream of tweets represented as Json trees whose content contains at least one of these tags. The *Twitter Graph* of the stream, is the graph $G = (V, E)$ with multiple edges E where V is the set of tags $\#x$ or $@y$ seen and for each tweet sent by $@y$ which contains tags $\#x, @z$ we construct the edges $(@y, \#x)$ and $(@y, @z)$ in E . The URL's which appear in the tweet can also be considered as nodes but we ignore them for simplicity. A stream of tweets is then transformed into a stream of edges e_1, \dots, e_m, \dots

We simultaneously captured 4 twitter streams³ on the tags #CNN, #FoxNews, #Bitcoin, and #Xrp (Ripple) during 24 hours with a window size of $\tau = 1h$ and a time interval $\lambda = 30\text{mins}$, using a standard PC. Figure 1 indicates the number of edges seen in a window, approximately $m = 20 \cdot 10^3$ per stream, on 48 points. For 24 independent windows, we read approximately $48 \cdot 10^4$ edges, and globally approximately $2 \cdot 10^6$ edges. The Reservoirs size $k = 400$ and on the average we save 100 nodes and edges, i.e. $4 \cdot 48 \cdot 100 \simeq 2 \cdot 10^4$ edges, i.e. a compression of 100. For $\gamma = 0.8$, the minimum size of a cluster is $m/\gamma \cdot k \simeq 60$. Notice that k is close to \sqrt{m} .

Figure 2 gives the three mains correlations $\rho(t)$ out of the possible 6 and

³Using a program available on <https://github.com/twitterUP2/stream> which takes some tags, a Reservoir size k , a window size τ , a step λ and saves the large connected components of the Reservoirs R_t .

the averaged correlation:

$$\rho'(t) = (\rho(t-1) + \rho(t) + \rho(t+1))/3$$

The correlation is highly discontinuous, as it can be expected, but the averaged version is smooth. The maximum value is 1% for the correlation and 0.5% for the averaged version. It is always small as witnessed by the correlation matrix. We experienced very small changes in the correlations and $\rho'(t)$, also computed online, witnessed it. The spectrum of the Reservoirs, i.e. the sizes of the large connected components is another interesting indicator. For the #Bitcoin stream, there is a unique very large component.

5 Search by correlation

We stored the history of the large clusters for each stream, i.e. the set of nodes of the clusters. Given a search query defined by a set of tags, the answer to the query is the set of the most correlated tags. We first need a definition of the correlation between a tag and a set of tags. Given a stream, we need to find some other close streams and use the standard Phylogeny method.

5.1 Phylogeny

Given a correlation matrix A_t between streams, a standard approach, such as the Neighbor Joining method [11], constructs a tree T with valued edges such that each stream appears as a leaf in T and the distance $d(i, j)$ between two streams in the tree is approximately the distance defined by the correlation matrix, i.e. $d(i, j) \simeq 1 - A(i, j)$. This construction assumes an additive property of the distances, but there is always an approximate solution.

In a learning phase ended at time t , we construct the tree T . Later on, we have a different matrix A' and a different tree T' as in the Figure 4. The *Tree Edit distance with moves* is a standard distance between trees where the basic operations are: edition of a label, insertion/deletion of an edge, move of a subtree. This distance is very easy to approximate [7], although the exact distance is *NP*-hard to compute. We just have to compare the k -grams of the subtrees at depth k . For the unordered trees of Figure 4, $\text{dist}(T, T') = 2$.

We can hence easily detect small or large changes in the tree T . Given a stream, the neighbors of a leaf in T are the closest streams in T , which we use in the Search by correlation.

We have a dynamic representation of the distances between streams, which we use in the next section.

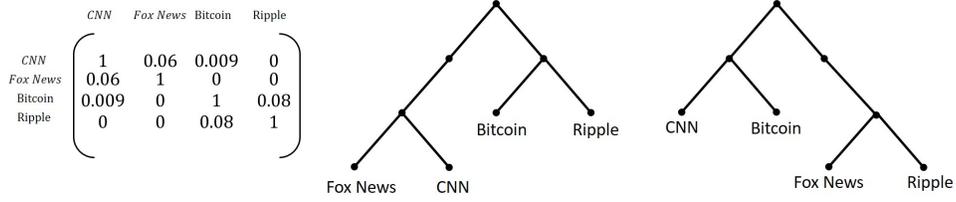


Figure 4: Phylogeny tree T (center) from the correlation matrix, and another close tree T' (right)

Input: CNN		
Ranking	t = 12	t = 24
1	#ODonnell	#POTUS
2	#NRA	#NRA
3	#POTUS	#ODonnell
4	#Obamacare	#Tucker
5	#MondayMotivaton	#2A

Input: CNN POTUS		
Ranking	t = 12	t = 24
1	#MondayMotivaton	#Tucker
2	#Hannity	#2A
3	#NRA	#NRA
4	#1A	#1A
5	#Tucker	#Hannity

Input: CNN POTUS NRA	
Ranking	t = 12
1	#Hannity
2	#2A
3	#1A
4	#Clinton
5	#ClintonFoundation

Table 1: Search results on inputs: σ =CNN, σ =CNN, POTUS and σ =CNN, POTUS, NRA

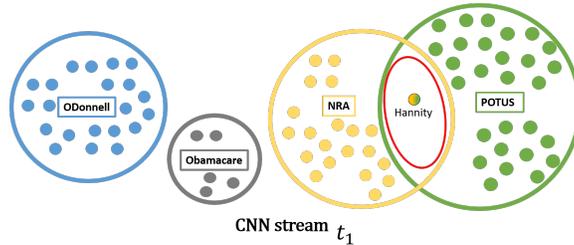


Figure 5: Nodes in an intersection of clusters for the input CNN, POTUS, NRA

5.2 Correlation between tags at time t

We extend the amortized correlation of section 2 to tags, i.e. node labels. Given a tag σ , we first check if it is a stream, the name of a cluster, or a simple node, using the tables *Stream*, *Clusters*, *Nodes*. In each case we recover the most recent clusters C_{t_i} of a stream σ' , or of the nodes. By construction, all the tags have at least one component they belong to.

Given two tags σ_1 and σ_2 , we retrieve their most recent components, C_{1,t_i} and C_{2,t_j} of the streams σ'_1 and σ'_2 and suppose $t_i > t_j$ and $\Delta(u) = t_i - t_j$. Assume dist is the distance between the streams σ'_1 and σ'_2 in the phylogeny tree. If one tag belongs to another component, we output the nodes of high-degree of the components. If these components intersect, then the tags of the intersection have a correlation coefficient of $(1 - \frac{\Delta}{t}) \cdot (\frac{t_i}{t})(1 - \text{dist})$. If a tag is in several intersections, we add the correlations.

If the components do not intersect, we look for another component C' of another stream close to σ'_1 and σ'_2 (using the phylogeny tree) which intersects C_{1,t_i} and C_{2,t_j} , or we look for older components C_{1,t'_i} and C_{2,t'_j} where $t'_i < t_i$ or $t'_j < t_j$ which do intersect. We generalize this definition for more than 2 tags. We can synthesize the *Search Algorithm* as follows:

Search by correlation Algorithm $A_4(\sigma_1, \dots, \sigma_l)$:

- For each tag σ_i , find the most recent components C_{i,t_i} . Output the nodes with the highest correlation coefficient,
- If there are no tags in the intersections, look at close streams (using the phylogeny tree) and their components.

The nodes in some intersection of recent components will have the highest correlation and will be the in the top answers. The *explanation* of the search will be these clusters C_{i,t_i} , which are associated with the given tags. In the example for Figure 5, the first answer #Hannity belongs to two CNN clusters, NRA and POTUS.

5.3 Experimental results

In the first example, we are given the tag $\sigma_1 = \text{CNN}$. As the table 1 indicates, the answers are the nodes of high-degree of the most recent component of the stream CNN. Notice that the answers depend on t , for the two examples $t = 12h$ and $t = 24h$. For the tags $\sigma_2 = \text{CNN}$, POTUS, we retrieve the most recent components and in this case, POTUS belongs a component of CNN: we output the nodes of high degree of that component.

For the tag, $\sigma_3 = \text{CNN}$, POTUS, NRA, we retrieve the three most recent components and in this case, the component of POTUS intersects the component of NRA and are both components of CNN. We output the nodes of the intersection, ordered by degree.

6 Conclusion

We introduced *the content correlation ρ between two graphs* and its extension $\rho(t)$ between two *dynamical graphs* based on the Jaccard similarity of their clusters. In the model of Uniform and Concentrated Dynamics for graphs with a power law degree distribution, we showed that the detection of a large connected component in a Reservoir built from uniform edge samples is a good method to distinguish a Uniform Dynamics from a Concentrated Dynamics, when S is large enough. This method generalizes to dynamic graphs and we can compute the content correlation of two streams with an online algorithm (Algorithm 3).

As we read different streams of edges in the window model, we only store the large connected components at some times t_1, t_2, \dots . In our experiments, we followed 4 Twitter streams for 24h, reading 2.10^6 edges, but kept approximately 2.10^4 nodes, i.e. 1% of the data. From the correlation matrix, we obtained the closest phylogeny tree. We defined the *Search by correlation* on some given tags, where the answers are tags ordered by correlation. The witnessed used as explanations of the correlations are the stored clusters.

References

- [1] Charu C. Aggarwal. An introduction to cluster analysis. In *Data Clustering: Algorithms and Applications*, pages 1–28. CRC Press, 2013.
- [2] Charu C. Aggarwal, Yuchen Zhao, and Philip S. Yu. On clustering graph streams. In *Proceedings of the SIAM International Conference on Data Mining, SDM 2010, April 29 - May 1, 2010, Columbus, Ohio, USA*, pages 478–489, 2010.
- [3] Nesreen K. Ahmed, Jennifer Neville, and Ramana Kompella. Network sampling: From static to streaming graphs. *ACM Trans. Knowl. Discov. Data*, 8(2):7:1–7:56, June 2013.
- [4] Brian Babcock, Mayur Datar, and Rajeev Motwani. Sampling from a moving window over streaming data. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '02*, pages 633–634, 2002.
- [5] P. Erdős and A Renyi. On the evolution of random graphs. In *Publication of the mathematical institute of the Hungarian Academy of Sciences*, pages 17–61, 1960.

- [6] Uriel Feige and Robert Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Struct. Algorithms*, 16(2):195–208, March 2000.
- [7] E. Fischer, F. Magniez, and M. de Rougemont. Approximate Satisfiability and Equivalence. *SIAM Journal of Computing*, 39(6):421–430, 2010.
- [8] Andrew McGregor. Graph stream algorithms: A survey. *SIGMOD Rec.*, 43(1), 2014.
- [9] Michael Molloy and Bruce Reed. The size of the giant component of a random graph with a given degree sequence. *Comb. Probab. Comput.*, 7(3):295–305, 1998.
- [10] Mark Newman. *Networks: An Introduction*. Oxford University Press, Inc., 2010.
- [11] N Saitou and M Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987.
- [12] Jeffrey S. Vitter. Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1):37–57, March 1985.
- [13] Reza Zafarani, Mohammad Ali Abbasi, and Huan Liu. *Social Media Mining: An Introduction*. Cambridge University Press, New York, NY, USA, 2014.