
**THÈSE DE DOCTORAT DE SORBONNE UNIVERSITÉ**

Spécialité Informatique

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

**Marwan GHANEM**

Pour obtenir le grade de

**DOCTEUR de SORBONNE UNIVERSITÉ**

---

# Les centralités temporelles : étude de l'importance des nœuds dans les réseaux dynamiques

---

Soutenue le 5 octobre 2018 devant le jury composé de :

| | | |
|---|---|---|
| *Rapporteurs:* | Vania Conan | Responsable de recherche, HDR, THALES |
| | Jean-Philippe Cointet | Professeur, Sciences Po |
| *Examinateurs:* | Marcelo Dias de Amorim | Directeur de recherche, CNRS |
| | Nathalie Mitton | Directrice de recherche, HDR, INRIA |
| *Directeurs:* | Clémence Magnien | Directrice de recherche, CNRS |
| | Fabien Tarissan | Chargé de recherche, CNRS |

# Acknowledgments

First of all, I shall forever owe my gratitude to Clémence Magnien and Fabien Tarissan, whom without I would not have reached this point. They have always been there for me, giving me their time, knowledge, and support. I am deeply grateful for everything they have given me.

My sincere thanks go to Jean-Philippe Cointet and Vania Conan for reviewing my dissertation, and to Nathalie Mitton and Marcelo Dias de Amorim, for accepting the invitation to integrate the jury of my thesis defense, and for taking the time to assess my work.

Mr. Antoine and Mr. Frédéric whom I am beholden for helping me discover the world of research. For over five years they have both supported me and guided during my studies.

All my lab mates, Clémence, Fabien, Aurore, Audrey, Thibaud, Tiphaine, Noé, Hong-Lan, Frédéric, Raphaël, Keun-Woo Lim, Robin, Léonard, Remy C., Léo, Remy P., Matthieu, Pedro, Olivier, thank you all for the scientific and non scientific talks, board games, rendering this experience enjoyable. Special thanks to Lionel and Maximilien for sharing your scientific knowledge generously and all the laughs we shared.

Team APR, Steven, Matthieu D., Vincent, Ghiles, Alice, Matthieu J., Boubacar, Raphaël, Martin, Romain, Frédéric, Antoine, Emmanuel, thank you all for the many laughs, pauses, video games and jokes we shared during these three years.

Outside of work, Charles my first friend in France forever I shall be grateful for having this friendship. 8 years that impacted my life. Lousia B. for being always there, feeding me and hosting me. Louisa H. for the long talks, random walks, your generosity, and your kindness. Hana, my first friend, your support during 23 years of friendship. Nathalie all the talks, crepes, aimless walks.

Mom and Dad, without the inspiration, drive, love, and support that you have given me, I might not be the person I am today. Words cannot describe my love to you both. Forever I shall be grateful for having both of you.

# Contents

# Introduction

Interactions are a huge part of our daily life: phone calls, crossing paths, sending emails, gossiping, IP traffic, trading, and many other examples. As they belong to different types, they can have different properties: they can be instantaneous or last for a period of time, directed or undirected, involve a measurable amount of information that is exchanged or not.

Now consider several interactions between a group of entities. This group can be a group of friends, colleagues, computers, students, *etc*. Several interactions can then be a vector for a spread of rumor through a group of friends or colleagues. They can represent a disease epidemic in a school; a malicious software spreading in a computer network. Undeniably, all these examples and many others show that interactions are important.

Lets us consider a group of friends: Charlie, Lucy, Marcie, Patty, Franklin, and Sally. Charlie gossips with Marcie and Lucy, afterward Marcie repeats it to Patty and at the same moment Lucy repeats it to Sally and Franklin (see Figure 1 for illustration). It is easy to see that Charlie's gossip reaches everyone. Moreover, Lucy can spread a disease easier than the others as she is in direct contact with Charlie, Sally, and Franklin.

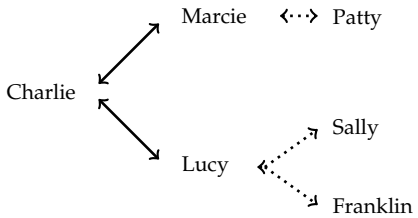


Figure 1: A schema representing the discussion between a group of friends. A line between 2 persons represents them talking to each other, with dashed lines occurring after the full ones.

Now consider a larger group of friends, or simply a large group of any type of entities. One can imagine many questions. Who can spread rumors faster? Can we detect the sub-groups of individuals? Does IP traffic travel via a specific machine? Can someone alter the news spreading? Can someone influence people easily? One would like to

answer these questions, which would give us a better understanding of how interactions affect our life.

To answer these questions and countless others, we can use graph theory. In the simplest form, a graph consists of a set of nodes that are connected to one another by links. The graph can be undirected, meaning a link has no orientation, or directed, in which case a link from a node $u$ to a node $v$ is not equal to a link from $v$ to $u$. This can easily describe interactions, the entities being represented by the nodes and the interactions by links. Many techniques exist to study the structure of these graphs to help us answer the questions above.

Consider again the group of friends. In the graph, each person is represented by a node, and the interactions are represented by the links between the nodes. Recall that Charlie gossips with Marcie, who in turn gossips with Patty. In a graph, these interactions can be represented as a sequence of links connecting two nodes, which is known as a path. Back to our example, we can see that the path Charlie $\rightarrow$ Marcie $\rightarrow$ Patty, represents the diffusion of the gossip.

Now if we want to detect that Charlie can diffuse information better than the others, or that Lucy can transmit a disease to the others better than them, several metrics exist that are based on the notion of links and paths, that we call centrality metrics. Numerous centrality metrics can be mentioned. One centrality method bases the importance on the number of interactions a node has; this can help us detect that Lucy can transmit a disease. Another centrality evaluates the importance of a node by how close it is to the others: it would help us detect that Charlie can diffuse a rumor better than the others.

Nevertheless, in practice graph theory does not answer these questions always in a correct manner. If Charlie speaks to Marcie before she talks to Patty, one can imagine that gossip or information that Charlie says will reach Patty. However, if Marcie and Patty meet before Charlie

speaks to Marcie, we can easily see the news can't reach Patty. Formally, the two interactions need to occur in a specific order for the gossip introduced by Charlie to reach Patty. The chronological order needs to be respected. In other words, there is a temporal aspect that exists and must be taken into account. In a real-world situation, interactions occur over time and the order of interactions is important. A graph in its simplest form, as described above, does not take into account this temporal aspect. Therefore, adaptations are required to take them into consideration.

How to achieve this has become a question of interest for many researchers. This introduces a lot of challenges such as: considering the temporal aspect correctly, considering it efficiently, *etc*. Many different manners to take into account the temporal aspect while addressing these challenges have been introduced.

In this thesis, we focus on centrality metrics. We study in Chapter 1 the different solutions that were proposed to adapt the centrality metrics to the dynamical context. In Chapter 2, we present the datasets used in this work. Afterward, in Chapter 3, to study and understand the different solutions, we developed a framework that compares and evaluates the different metrics. This is work is currently under minor revisions [Ghanem et al., 2018a].

We will observe that these metrics tend to be computationally expensive, and so naturally, we are not able to study certain networks that are too large. This brings us to the second part of this thesis. We concentrate on the computability of the centrality metrics. Our goal is to reduce the high computational demand and have the ability to analyze large datasets. To do so, we explore in Chapter 4 several possibilities such as computing the centrality for fewer instant or exploiting structural properties. We validate these propositions on several real-world networks. The first part of this chapter has been published [Ghanem et al., 2018b].

With a similar goal, however with a different approach,

we consider in Chapter 5 a real-world application: the diffusion of information in Delay Tolerant Networks. Considering a specific application allows us to introduce a centrality metric, tailored especially for this case, hence more efficient. The work of this chapter has been published [Ghanem et al., 2017].

Finally, we summarize our contributions and we discuss some of the possible perspectives that this work opens.

Chapter 1

# State of the Art

Finding the important nodes has been a question of interest for some time. This lead to the introduction of a large number of metrics, called centrality metrics. Each of these centrality metrics captures the importance in a different manner. In social network analysis, these metrics can be used to find the important individuals in each network. This can be useful in the case of message diffusion. In infrastructure networks, centrality metrics can find the critical points of the infrastructure, so they can be protected. Another case is movement networks: in an epidemic situation, when we lack the ability to vaccinate everyone, it is more interesting to find the individuals that are prone to transmitting the disease more than the others.

In this chapter, we present the principal centrality metrics that were introduced for static graphs. As these centrality metrics are not always adequate for dynamic networks, we study the existing adaptations for the temporal case. In a second part, we observe that computing these centralities, in both the temporal and static case, is not always computationally feasible. Thus, we investigate the existing approximation and estimation methods for existing centrality measures in both the temporal and static case.

## 1.1 Static Centralities

A network is represented in the form of a graph $G = (V, E)$, which consists of a set $V$ of nodes and a set $E$ of links in the form of $(u, v)$ where $u, v \in V$. These links can be directed or undirected, representing the transfer of information in one or both directions respectively. Figure 1.1 represents a simple graph consisting of six nodes and eight undirected links.

The first centrality to be introduced is the degree centrality. It defines the importance of a node by the number of links it has. From this centrality, the chances of an individual (represented by a node) to become infected by a virus or influence the nodes around it can be estimated. For example, the node $a$ in Figure 1.1 has a degree centrality equal to 4, as it is connected to nodes $b$, $c$, $d$ and $e$.

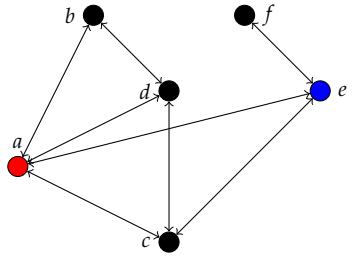


Figure 1.1: A toy example of a graph.

### 1.1.1 Shortest path based centralities

Afterward, more subtle notions were explored, and using these notions the community introduced more sophisticated centrality metrics. One of these notions is the path. A path is a sequence of links which connects nodes, with the number of links representing the path's length. A shortest path between two nodes has the shortest length. The length of such a path is also known as the distance. For example, in the toy example in Figure 1.1, the distance between $c$ and $f$ is 2 as the shortest path contains two edges $c \to e$ and $e \to f$.

As one would expect information to travel via the shortest path, a class of centrality metrics was introduced to evaluate the importance of nodes in relation to this notion. One of the centrality based on the shortest path was introduced by [Bavelas, 1950] and is named the Closeness centrality. Informally, this centrality attributes more importance to the nodes that are closest to the others. Formally, this is calculated for a node $u$ as the sum of the inverse of distances between $u$ and all the other nodes. The definition

of this centrality for a node $u$ is:

$$Closeness(u) = \sum_{v \neq u} \frac{1}{d(u,v)}, \tag{1.1}$$

where $d(u,v)$ represents the distance between $u$ and $v$. One should note that when there is no path between $u$ and $v$, the distance between them is equal to infinity ($d(u,v) = \infty$), which translates to 0 in the centrality computation[1]. In the toy example in figure 1.1. The node $a$ (in red) can reach four nodes via one edge and one node via two edges. This is not the case for any other nodes, thus the node $a$ has the highest closeness centrality, that is equal to 4.5.

1. With the convention that $1/\infty = 0$.

Another centrality based on the shortest path was introduced by Freeman and is called the Betweenness Centrality [Freeman, 1977]. This centrality measures a different notion of importance than that of Closeness centrality. It measures the extent to which a node tends to be on the shortest paths between other nodes. The betweenness centrality of a node $u$ is defined as the sum over all pairs of nodes $s$ and $t$ of the fractions of shortest paths from $s$ to $t$ that go through $u$. Formally the centrality for a node $u$ is defined as:

$$Betweenness(u) = \sum_{s \neq u \neq t} \frac{\sigma_{st}(u)}{\sigma_{st}}, \tag{1.2}$$

where $\sigma_{st}$ represents the number of shortest paths between the nodes $s$ and $t$, and $\sigma_{st}(u)$ represents the number of those paths that go through $u$. One could expect the node with the highest closeness centrality to have the highest betweenness centrality, however this is not the case. In Figure 1.1, the node $e$ (blue) has a higher betweenness than $a$. The node $a$ is globally closer to all the nodes than $e$, however, for information to diffuse from $f$ to reach any of the nodes, it has to travel via $e$. This increases the importance of $e$, while in the case of $a$, we can observe that $d$ can act as a replacement in certain cases, thus lowering $a$'s importance as fewer shortest paths pass via this node. This simple example shows that both centralities are based

on the notion of shortest paths, yet they capture different notions of importance.

### 1.1.2 Spectral Centralities

A second class of centrality metrics, called spectral centralities, was introduced in parallel. In this class, we consider that information is more likely to go via a random path of any length, rather than through the shortest paths. Thus, in this centrality class paths of all lengths are taken into account. These centralities are based on the representation of the graph in the form of a matrix. This matrix is called *adjacency matrix*. The element $A_{uv}$ in the matrix is equal to 1 if there is a link between the nodes $u$ and $v$, and to 0 otherwise. Naturally, in the case of a undirected graph, this matrix is symmetric. Figure 1.2 represents the adjacency matrix for the toy graph previously observed.

$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

Figure 1.2: Adjacency matrix of the top graph.

Eigenvector centrality is one of the most known spectral centrality. It was introduced by [Bonacich, 1987]. This centrality is a natural extension of the degree centrality. In this centrality, connections to important nodes are more valuable than those with insignificant nodes. Thus, a node with numerous links to irrelevant nodes has a lower centrality than that of a node with few links to important nodes. For a node $u_i$, the eigenvector score is defined as:

$$\lambda c_{eigenvector}(u_i) = \sum_{j=1}^{n} a_{ij} c_{eigenvector}(u_j) \quad \forall i, \tag{1.3}$$

where $n$ is the number of nodes in the graph and $\lambda$ is a constant. This formula can be rewritten into the form of

$$Ac = \lambda c, \tag{1.4}$$

where $c$ is an eigenvector of $A$ and $\lambda$ is the associated eigenvalue. From the Perron-Frobenius theorem, we know that there is a unique and positive solution, if $\lambda$ is the largest eigenvalue associated with the eigenvector of the adjacency matrix A [Newman, 2010]. In this eigenvector $c$, the $v$th element of the vector gives the centrality score of the $vth$ node. To solve 1.4, a common method is the Power

iteration which can be used to find the dominant eigenvector, from which we have the centrality scores for all the nodes. If we apply this method to toy example, we get the following eigenvector: $[0.54, 0.33, 0.46, 0.45, 0.38, 0.12]$. Like in the case of closeness, the node $a$ has the highest centrality, however, we can observe that both nodes $c$ and $d$ have a higher centrality value than $e$, that had the highest betweenness centrality and second highest closeness centrality.

One of the problems of the previous centrality is that paths of all lengths are scored equally. To solve this, [Katz, 1953] proposed the Katz centrality with the purpose of giving less importance to long paths. This is done using a parameter $\alpha$, that acts as attenuation factor, where $\alpha < 1$. Paths of length $p$ are weighted by $p^{\alpha}$. Employing the fact that $A_{ij}^{p}$ gives the number of paths of length $p$ from $i$ to $j$, the following equations combined all the paths of all lengths:

$$(I - \alpha A)^{-1} = I + \alpha A + \alpha^2 A + \cdots , \tag{1.5}$$

where $I$ is the identity matrix. This equation combines all the paths, while penalizing each path of length $p$ by a factor of $p^{\alpha}$. Finally, using this result, the sum of each row of the matrix gives the centrality score of each node. One should note that, as the Katz centrality gives a lower score to long paths, this centrality can simulate the loss of information as it travels further. [Estrada and Hatano, 2008] introduced communicability centrality which is quite similar to Katz centrality. It also quantify the importance of a node in relation to all the paths in the network, while penalizing long paths. Several other spectral centralities exist such as PageRank [Page et al., 1999], which was specially designed for webgraphs. See [Newman, 2010] for more centrality metrics.

## 1.2 Temporal Centralities

Over time, it became obvious that the temporal aspect of networks should be taken into consideration [Lerman et al., 2010].

In dynamic networks, nodes interact at various times, thus the links between the nodes as well as the nodes themselves can appear and disappear over time. Consequently, there is a chronological order between the links that should be taken into consideration. Therefore, the simple solution consisting in: aggregating the graph and omitting the temporal information, to be able to use the classic centralities, is not adequate. For example, if we consider an aggregated graph and a shortest path based centrality the chronological ordering will not be respected and paths that do not respect time will be considered. In other words, a path could simulate information going from the future to the past. Figure 1.3 represents a toy dynamic graph with three nodes and two links. In the graph, the first link occurs between $a$ and $b$ at instant 1, and later on at instant 2, the second link occurs between $b$ and $c$. Naturally, information can pass only from $a$ to $c$ via $b$, but not from $c$ to $a$. A classic static centrality will consider the path from $c$ to $a$, which should normally not be considered. Several studies [Holme and Saramäki, 2012, Mantzaris and Higham, 2013, Liao et al., 2017] found that this can lead to the overestimation of the diffusion of information in the network. Indeed a static centrality will consider paths that are temporally meaningless and thus more transfer of information that is in reality possible.

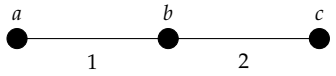


Figure 1.3: A toy example of a dynamic graph, with labels on each link representing the instant in which the links occurs.

Another obstacle exists: as the graph changes the nodes' centrality values evolve. As classic centralities attribute each node a single value representing its centrality, thus this dynamic change of a node's importance is completely omitted by studying the aggregated graph. Again if we consider the toy graph in Figure 1.3, at instant 1: the path $(a, b)$ exists while, at instant 2, the path $(a, b)$ disappears and $(b, c)$ appears. Therefore, it is expected that the centrality of each node to be different in each of these two instants.

### 1.2.1 Online Algorithms

Certain papers considered efficiently computing the static centrality whenever the network changes or evolves. For instance, Kas *et al.* [Kas et al., 2013] propose an algorithm that, given the distance between all pairs of nodes and a network change[2], computes the new value of the closeness centrality by updating the distance values rather than computing them from scratch. Another approach consists of finding the subgraph that was affected by the network's change [Lee et al., 2016]. From there, the betweenness centrality is recalculated only for the subgraph. Several other propositions [Singh et al., 2015, Nasre et al., 2014, Green et al., 2012] were introduced to update the betweenness centrality efficiently after each modifications in the graph. Santos *et al.* [Santos et al., 2016] introduced an efficient parallel and distributed algorithm to update the closeness centrality. However, this algorithm handles only the case of link deletion.

2. Links appearing or disappearing.

Finally, these methods are relevant only when the network evolves at a slower rate than that of the dissemination. For example, in a cellular network, phone calls are instant while the addition or removal of an antenna is quite slow, and the network can remain stable for years. Hence, considering a static centrality and taking into account paths that do not respect time is not an issue. In other words, simulating the spreading of information at any instant using classic methods remains relevant. On the other hand, in epidemic networks, individuals interact quite faster than the spreading of the diseases. The paths of between individual change during the propagation and thus, these changes should be taken into account.

### 1.2.2 Shortest path based centrality adaptations

One of the first works to adapt shortest path based centrality to take into account the evolution of temporal networks was introduced by Uddin *et al.* [Uddin et al., 2013]. This simple solution consists in considering the temporal graph

as several static graphs each representing an equal period of time. See figures 1.4 and 1.5 for illustration.

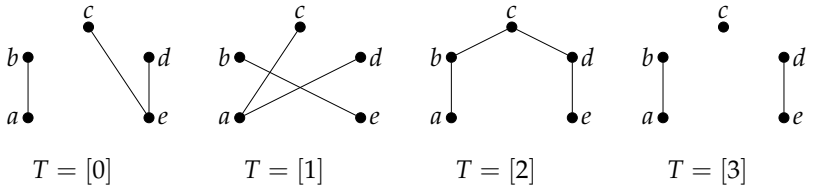



Figure 1.4: A representation of a dynamic graph in the form of 4 snapshots representing each 1 instant.

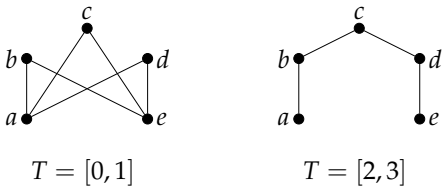



Figure 1.5: A representation of a dynamic graph in Figure 1.4 in the form of 2 snapshots representing each 2 instants.

Using this representation, the authors proposed a framework that allows the direct use of existing centrality metrics without the need of any modifications. For any given centrality metric, it is computed on each snapshot, attributing to each node a centrality value representing the importance of the node during this period. Thus, this gives several centrality values for each node (as many as snapshots). While this method proved to be more accurate than static analysis, it has several disadvantages. Firstly, as observed in the static case, paths that do not follow the chronological order of links are still considered inside each snapshot. For example, in Figure 1.5, in the snapshot $[0, 1]$, we can observe a path from $c$ to $b$, via the path $c \to a \to b$. However, this path is temporally wrong: if we look at Figure 1.4, we can see that the link $(c, a)$ occurs after $(a, b)$. Secondly, each centrality value represents a certain period rather than a specific instant. This means that the changes are centrality value is not completely represented. Thus, the dynamicity of the nodes' importance is not well described. Another issue is that to use this method, the size of the periods needs to be determined. The choice of this value has been investigated in [Krings et al., 2012, Léo et al., 2015]. If the time scale is too large, the temporal information is lost, thus the benefit of the approach is lost. When the time scale

is too small, each snapshot becomes too disconnected and contains paths and thus, the snapshot becomes meaningless. We can observe in Figures 1.4 and 1.5, that when the snapshot size is increased, a lot of temporal information is lost.

Finally, as each snapshot is analyzed separately, paths between the snapshots are not taken into account. We investigate the snapshot method more deeply in Chapter 3. Furthermore, [Uddin et al., 2016] propose two metrics which quantify the change in importance of a node in a dynamic network. This approach is subtly but really different from a study of the time evolution of a node's importance, as the authors quantify the total amount of dynamicity in the network. The snapshot method was also used by [Kim et al., 2012] to predict degree, closeness and betweenness centrality.

[Braha and Bar-Yam, 2008] also considered the snapshot method. They used the degree centrality to find local hubs in each snapshot. Additionally, they showed that between each snapshot there is a huge fluctuation in the results. The fact that they are capable to detect this fluctuation using a method that partly omits the network's dynamicity further confirms that this dynamicity should be taken completely into account. Finally, they compared the snapshot results with the aggregated results and showed that results were quite different. Additionally, they use the snapshot method to find cycles.

To take into account the transfer of information between snapshots, [Tang et al., 2010] redefined the notion of a temporal path. This definition considers paths between snapshots. Additionally, they limited the number of links that can be traversed each snapshot. Formally a path between nodes $i$ and $j$ is defined as:

$$p_{ij}^{h} = (n_1^{t1}, \cdots, n_k^{tk})$$

where $n_1$ is $i$, $n_k$ is $j$, $t_{k-1} \leq t_k$ and $h$ is the maximal number of consecutive nodes in the same snapshot. Thus, such a path allows transfers between snapshots, while limiting

the number of exchanges during the snapshot. The authors redefine both betweenness and closeness centralities with the introduced path definition. Therefore, this method is capable of considering long paths that go through the whole dataset. However, inside each snapshot, the chronological order of links is not respected. Thus, the paths that they consider could be misleading. Additionally, both centrality adaptations attribute each node a single centrality value that represents the global importance of a node in the whole dataset, rather than represent a specific instant.

Other approaches consist of defining a new notion of path. This rethinking would allows a complete representation of the dissemination of information. In [Pan and Saramäki, 2011], the notion of *temporal path* is introduced, however not defined formally. A temporal path consists of a sequence of links that are time-respecting. In figure 1.6 (left), there is a temporal path from node $B$ to node $E$ via the links ($B$,$A$), ($A$, $C$) and ($C$, $E$). From this, the authors define the notion of *temporal distance* between two nodes $u$ and $v$ as the shortest time to reach $v$ from $u$. Their study concentrated mostly on the paths, yet they introduced an adaptation of the closeness centrality. However, this adaptation represents the importance by a single figure, representing the global importance of nodes.

[Nicosia et al., 2013] introduce the notions of temporal betweenness centrality and closeness centrality by using the notion of temporal paths. However, their definitions consider only paths that start at the beginning of the dataset's time span. This can be misleading: for example, if two paths are equivalent, but one starts at the beginning and the other does not, only the one that starts at beginning is only considered. For this is reason, the paths in the dataset that occur after all nodes can be reached from each other are not taken into account. This underestimates the dissemination in the network. [Qiao et al., 2017] introduced a temporal closeness definition with the same notion of paths. However, their definition considered paths only in a certain time interval, and thus, the centrality value rep-

resents the importance of nodes for a certain time interval.

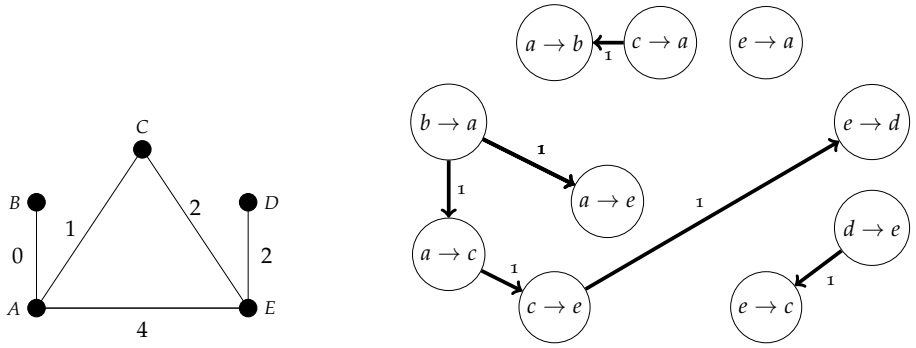



Figure 1.6: Left: A labeled graph, a link's label gives the instant at which the link has occurred. Right: The representation of the graph in the aggregated form proposed by [Scholtes et al., 2014].

[Scholtes et al., 2014, Scholtes et al., 2016] introduced another method to take into account the temporal aspect. They defined the *higher order aggregate network*, which aggregates the temporal network into a static network, in which each node represents a path. This model has different levels of aggregation. In the *kth* level, each node represents a path of length $k-1$, and a link between these two nodes represents a path of length $2k-2$. Each link is labeled by the number of times it appears. Thus this representation keeps information about the number of times each path appears, but not when. To observe this, in Figure 1.6 presents a temporal graph and the representation in the form of *higher order aggregate network* with $k=2$. We can observe how each node represents a path of length 1, and the links represent longer paths. Additionally, we observe how a temporal graph of 5 nodes and 5 links has to be represented using 10 nodes and 6 links, thus the memory need increases quite fast. They redefine both closeness and betweenness centrality for this model. Additionally, they introduced a novel centrality called *temporal reach centrality*. For a node, it measures the number of nodes that can be reached from this node via paths with a given maximum length. However, these centralities are too costly to be computed except for small graphs. Additionally, they represent the centrality of each node by one figure.

In a similar fashion using a static representation, [Kim and Anderson, 2012]

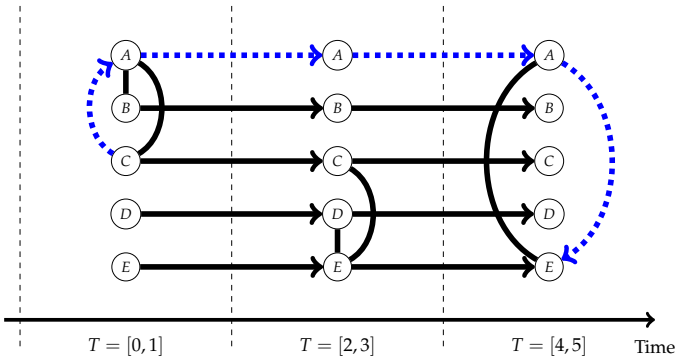



Figure 1.7: The toy example of Figure 1.6 (left) in the form of a *time-ordered graph* model.

proposed a method to transform temporal graphs into static graphs without loss of temporal information. They introduced the *time-ordered graph* model that represents a temporal graph in the form of a directed static graph. It decomposes the temporal graph into several static graphs each representing one instant. Each node is duplicated in each static graph and each node is linked to its duplicate in the following snapshot. However, as this is computational expensive, they use static graphs that represent periods larger than one instant. Figure 1.7 presents a temporal graph and the representation of this graph in the form of a *time-ordered graph*, with each static graph representing a period of 2 instants. In dotted blue we can observe one possible temporal path that would be considered by this method. From this model, they redefined the classic definition of betweenness and closeness. Nevertheless, these methods still evaluate the importance using a single value.

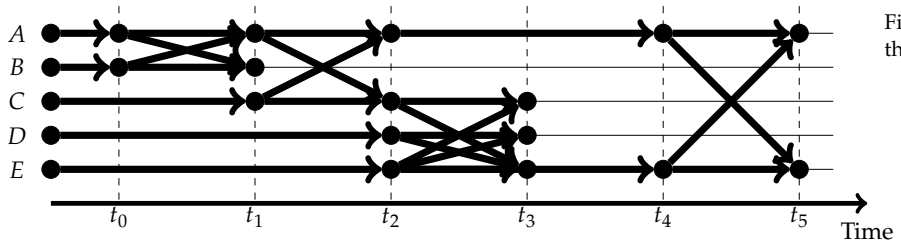



Figure 1.8: The toy example 1.6 (left) in the form of a *direct acyclic graph*.

Another approach that represents also a dynamic network in the form of a static network was introduced in

[Takaguchi et al., 2016]. In this work, the authors create a copy of each node for each instant where it is active and link each node to its following duplicate using a directed link. Additionally, the links representing the interactions between nodes are kept. Figure 1.8 represents the toy example of Figure 1.7 (left) in the form of a *direct acyclic graph*. The number of links and nodes required to represent the dynamic graph is much larger than the original network. Thus, this representation requires a lot of memory, in particular for highly active datasets. This representations is quite similar to the *time-ordered graph* representation. The authors introduced two centralities: *temporal coverage centrality* and *temporal boundary coverage*. We discuss these centralities more deeply in Chapter 3.

[Kostakos, 2009] considered the same representation, but the study concentrated on structural metrics such as average geodesic proximity, which is the average number of edges that separate two nodes. From this, for a node $u$, the author quantifies the average geodesic distance from $u$ towards all the other nodes as well the average geodesic towards $u$ from all the nodes. In a similar manner, for two nodes $u$ and $v$, the author quantifies on average the time required to reach $v$ from $u$.

[Latapy et al., 2017] introduced lately a novel model called *link streams*. In this model, rather than representing a dataset as a graph, the dataset is represented as a flow of links, thus keeping all the temporal information. In addition to introducing all the basic notions found in graph theory, they introduced both closeness and betweenness centrality for the link streams.

Several other propositions acknowledge that the distance between nodes vary over time [Shamma et al., 2009, Alsayed and Higham, 2015, Williams and Musolesi, 2016]. In each of these propositions, the temporal paths were taken into account. However, the proposed methods still represented the importance of each node by a single figure that represents the global importance.

### 1.2.3 Spectral centrality adaptations

Several papers introduced adaptations and variants for the spectral centralities. [Laflin et al., 2013] considered a snapshot approach, and they are able to take into account the chronological order of the interactions. The authors do this by reconsidering the notion of path. In this case a path consists of a sequence of edges, and each link in the sequence occurs before or at the same instant as the following link. They then use the classic Katz centrality, hence they still represent the importance of a node by a single figure. [Mantzaris and Higham, 2013] applied this method to an email dataset, in which they simulated an epidemic spreading from a node. Their goal in this paper was to observe the infection rate in this dataset. They showed that the centrality method was a good estimation of the simulation results; in other words nodes with a high centrality diffused the disease better.

In the same manner, [Huang and Yu, 2017] proposed a spectral based centrality named *Dynamic-Sensitive centrality*. They use this centrality to detect the chances of each individual of becoming infected in three real-world datasets. They showed that this centrality performs better than static centralities. However, they also do not study the changes in the nodes' importance and represents the importance by a single value.

[Estrada, 2013, Grindrod et al., 2011] extended the communicability centrality, and introduced *Temporal communicability centrality*. This temporal adaptation can be computed in theory at any instant, however, in practice the authors use a snapshot method and compute the value for each snapshot. As this centrality regroups temporal and topological effects in graphs, [Colman and Charlton, 2016] proposed the splitting of temporal communicability centrality in two centrality metrics. They proposed two centralities: Broadcast centrality and Receive centrality. They showed using simple examples that a node does not have to be the most active to be the most important. Additionally, they performed several statistical studies on real-

world datasets.

[Wang et al., 2017] also use the snapshot method to redefine the degree centrality to the temporal case. Afterwards, they compute the deviation of degree centrality between each snapshot, *i.e.* that is how the degree centrality of each node differs between each snapshot. This centrality proved to be a good indicator to find the important nodes in epidemic models. Silvia *et al.* [Silva et al., 2017] used the same model for eigenvector centrality; additionally they detected communities.

Finally to get a better understanding of the snapshot method, Flores *et al.* [Flores and Romance, 2018] studied the difference between eigenvector-like centralities in discrete (snapshot) and continuous time scales. They show that real-world datasets were represented better by the continuous than discrete time centralities.

Another model known as *Temporal quantities* was introduced in [Praprotnik and Batagelj, 2015, Batagelj and Praprotnik, 2016]. They redefine the algebraic operations for the matrix computations, to take into account the temporal aspect. From this, they introduce a temporal version of both Katz and eigenvector centrality. However, this method is computationally demanding. In a similar manner, [Ghosh et al., 2014] redefine the algebraic operations to take into account the temporal aspect. From this, they propose a framework that computes a generalized spectral centrality. However, they represent the importance of each node by a single figure.

[Fenu and Higham, 2017] proposed an additional representation. The authors represent a dynamic network, in the form of a matrix where each column/row corresponds to a pair composed of a node and a time instant. [Taylor et al., 2017] introduced a similar representation known as *supra-centrality* matrix and study the eigenvector centrality on this matrix. We investigate this method deeper in Chapter 3.

This representation was used by [Prado et al., 2016] to analyze two books. The authors aggregated each chapter in a single graph. In other words, they used a snapshot

method, with each snapshot representing a chapter. Their study showed that aggregating the whole book fails, compared to aggregating per chapter, to capture the dynamicity of the book, for instance as the appearance and disappearance of characters in the books overtime. Additionally, the importance was attributed to the wrong character in the books when the books were aggregated.

[Lerman et al., 2010] proposed an adjacency matrix that represents the whole dynamic process. The adjacency matrix of each snapshot is computed and afterwards, these matrices are combined into a matrix of size $N \times N$, where $N$ is the number of nodes. In this matrix, the element $i, j$ represents the cumulative amount of information that reaches $j$ from $i$. Two types of matrices are generated, one where information can be kept within the node (*conservative diffusion*) and another corresponding to paths where the information is lost if it is not passed forward immediately to another node (*non conservative diffusion*). They introduced the *dynamic centrality*, which measures the importance a node $u$ by the total amount of information sent by $u$ that reaches all the other nodes.

In this work, their main goal is detecting influential papers in citation networks. They compared dynamic centrality with PageRank, which consists of aggregating the network and omitting all the temporal information. This comparison showed that the results given by dynamic centrality are different from that given by PageRank. This indicates that the temporal aspect should be taken in consideration

This method was further extended in [Ghosh et al., 2011, Ghosh and Lerman, 2012]. They studied citation networks, with each snapshot representing a period of one year. They showed that PageRank is better to find important actors in the conservative model than $\alpha$-centrality[Bonacich, 1987]. However, in the case of a non conservative model, $\alpha$-centrality performed better.

[Rozenshtein and Gionis, 2016] proposed a unique method to adapt PageRank to the temporal case that the authors

call Temporal PageRank. They redefined the notion of path, to respect the chronological order, rather than reconsidering the adjacency matrix as is commonly done with spectral centrality adaptations. Their study concentrated on the theoretical aspects of the classic PageRank[3] and temporal PageRank. Interestingly, they showed that in certain cases the temporal PageRank is an estimation of PageRank centrality in the aggregated graph.

3. Aggregating the network into a static graph.

### 1.2.4 Conclusion

We observe, for both shortest path based and spectral centralities that there are many adaptations taking into account the temporal aspect of the datasets. While these adaptations might appear to be quite different, in most cases they share many aspects. Several adaptations can be seen as snapshot based, aggregating certain periods of time for efficiency reasons. Another class of adaptations introduced similar path definitions with subtle differences.

Additionally, certain methods still represent the importance of a node by a single value and do not study the time evolution of this importance. Other methods are computationally demanding for highly active datasets, either because of the very large memory requirements or high algorithmic complicity. For a review, see [Holme and Saramäki, 2012, Holme, 2015]. To conclude, we can observe several properties that a good adaptation should possess:

1. Each node should have a centrality value at each instant;
2. The complexity should be low;
3. A notion of temporal path should be used.

## 1.3 Approximation

As we observed in the previous sections, computing both static and temporal centrality methods can be quite expensive. For example, to compute the static betweenness centrality for a node, all the shortest paths need to be com-

puted. [Brandes, 2001] proposed a method to compute betweenness centrality in $O(NM)$, where $N$ and $M$ are the number of nodes and edges respectively. This is much lower than the straightforward method in $O(N^3)$. Thus, the community designed methods estimate the centrality rather than efficiently compute the exact value.

### 1.3.1 Static graphs

We start by investigating the methods proposed for the static case. [Eppstein and Wang, 2001] introduced one of the first methods to estimate closeness centrality. They randomly select a number of nodes, from which they compute the shortest paths to all the other nodes of the graph. Afterward, for each node they compute the average distance to the sampled nodes, and use this value to estimate the node's centrality. [Brandes and Pich, 2007] used the same random strategy proposed in [Eppstein and Wang, 2001] and proposed several strategies to select nodes for the estimation, such as: randomly selecting nodes in proportion to their degree, to increase the chances to find the hubs; a strategy to increase the distance between the sampled nodes, to cover the whole network better; a strategy to select the periphery nodes; a strategy to decrease the distance between the sampled nodes, to avoid the overestimation of the distances. After comparing all these strategies, they concluded that the random strategy performs much better than the other sophisticated selection strategies.

Several adaptive sampling methods were introduced. In such methods, the number of nodes as well as the nature of nodes varies in relation to the network in question. [Bader et al., 2007] built on the sampling concept introduced [Eppstein and Wang, 2001] and proposed an adaptive sampling method. For a given node, they randomly sample a number of nodes to approximate their betweenness centrality. However, the number of sampled nodes is not fixed, thus the method is considered adaptive. The number of sampled nodes depends on the information obtained from each previously sampled node. In other terms,

after each sample, the number of required sampled nodes can increase or decrease. [Potamias et al., 2009] studied the estimation of the distance between nodes. They show that finding the optimal set of sampling nodes is a **NP**-hard problem. Thus, they study several heuristics to be able to select these nodes. Later on, [Tretyakov et al., 2011] improved this method to be able to estimate the distances between nodes in milliseconds for graphs with over three billion edges.

[Maiya and Berger-Wolf, 2010] used concepts from graph expanders [Hoory et al., 2006] to sample the nodes for the approximation. This method consists in randomly selecting a node, and afterwards all following sampled nodes are selected from the neighborhood of the already selected nodes. Each time they sample a node, they select the node that would give the largest number of possible nodes to be sampled for the next turn. Thus, this slowly expands the graph. They show that this method can approximate betweenness, closeness and eigenvector centrality. They compared this method to random-walk (**RW**) and breadth-first search (**BFS**) sampling methods that were introduced in [Brandes, 2001, Eppstein and Wang, 2001]. They showed that their proposed method performs better than **RW** and **BFS**. Later on, [Lim et al., 2011] modified this method and presented an extensive study, from which they showed that taking into account the nodes' degree centrality helps in estimating other centrality metrics.

[Borassi et al., 2015] proposed a method to detect the top $k$ nodes: the $k$ most important nodes. To do, the authors modified the BFS procedure used in computation of closeness centrality. When computing the closeness for a node $u$, the proposed BFS procedure stops once it is sure that the node $u$ is not in the top $k$ nodes. Later on, [Bergamini et al., 2016] built on this method and introduced a faster algorithm for unweighted graphs. They keep track of the lower bound of the farness of each node. The farness is the opposite of closeness, it measures how far a node is to the others. A node's lower bound farness

indicates if the node is in the top $k$ nodes.

In a different approach, [Okamoto et al., 2008] suggests estimating the ranking of the nodes rather than estimating the centrality values. To identify an important node, the centrality value is not as useful as simply knowing how well a node performs in comparison to the others. Thus, Okamoto *et al.* introduced a ranking method for closeness centrality that uses both the estimation method of [Eppstein and Wang, 2001] as well as the exact method of [Fredman and Tarjan, 1987]. Their method simply evaluates how each node compares to the others rather than computing the centrality value. This is much more efficient than estimating the centrality and ranking the nodes afterwards. However, again this method only estimates the ranking rather the centrality value. Thus for example, distinguishing two nodes with quite similar centrality values is not possible. More over, [Saxena et al., 2017] showed that ranking versus closeness centrality always follows a sigmoid pattern; using randomly selected nodes, they estimate the centrality for nodes of average importance. Afterwards, with this estimated value and the sigmoid form, they estimate the ranks and values of centrality of all the nodes.

### 1.3.2 Evolving graphs

Finally, we are not aware of any estimation methods that exists for temporal graphs. Nevertheless, certain methods have been proposed for evolving graphs. Bergamini *et al.* [Bergamini et al., 2014] introduced one of the first methods to estimate betweenness centrality for weighted and unweighted evolving graphs. Their method proves to be much faster than existing methods that recompute the estimation from scratch, however, they estimate the centrality after a batch of modification and not after every single modification. Other methods were introduced in [Kourtellis et al., 2015, Riondato and Upfal, 2016], however, they do not take into account the totality of the temporal aspect. In other terms, they consider chronologically meaningless paths, and thus their methods are only relevant for evolving graphs and not temporal ones.

# Chapter 2

# Datasets

In this thesis we are interested in datasets that represent a set of interactions occurring between entities at different instants over a fixed duration. Depending on the context, the entities for instance can be individuals and an interaction is the simple act of being near another individual. Another example is the exchange of emails between email accounts; in this case, the entities are the email accounts and sending an email is an interaction between the sender and receiver. Social networks such as Twitter, can also be considered, where the Twitter accounts are the entities, and retweeting a tweet of another account is the interaction.

In this chapter, we split the datasets that we analyzed in this work by groups depending on their nature. For each dataset, we describe its basic properties and study certain aspects such as the activity over time. All the datasets are anonymised.

## 2.1 Email-exchange Networks

The datasets in this group are based on the exchange of emails. In this group, the entities are individuals or more precisely email accounts. The interactions correspond to one email account sending an email to another account.

When a single email is sent to several accounts, this is represented by several interactions.

*Enron*

The Enron dataset represents the emails exchanged by 150 employees of the Enron company [Shetty and Adibi, 2005]. The dataset was obtained by the Federal Energy Regulatory Commission during its investigation in Enron's collapse. The 150 employees exchanged over 47,000 emails during three years. For each email exchanged, information on the senders, receivers, and the moment it was sent is available.

*Radoslaw*

Similarly to the Enron dataset, the Radoslaw dataset represents the exchange of emails between the employees of a mid-size company [Michalski et al., 2011]. The company contained 168 employees, who exchanged over 80,000 emails during a period of 9 months in 2010. Like Enron, for each email, it records information on the senders, receivers, and the moment they were sent.

*Democratic National Committee (DNC)*

This dataset comes from a leak of the emails exchanged between the members of the Democratic National Committee, which is the formal governing body for the United States Democratic Party [Kunegis, 2013]. The leak occurred in 2016 and the email dump contained all email data exchanged by the 1891 members. This corresponds to around 40,000 email exchanges from September 2013 to May 2016. Here, we only kept the sender, receivers, and the moment each email was sent.

*UC Irvine messages network (UC)*

These are the messages exchanged by the students of the University of California, Irvine [Opsahl and Panzarasa, 2009]. This dataset contains just under 60,000 messages that were

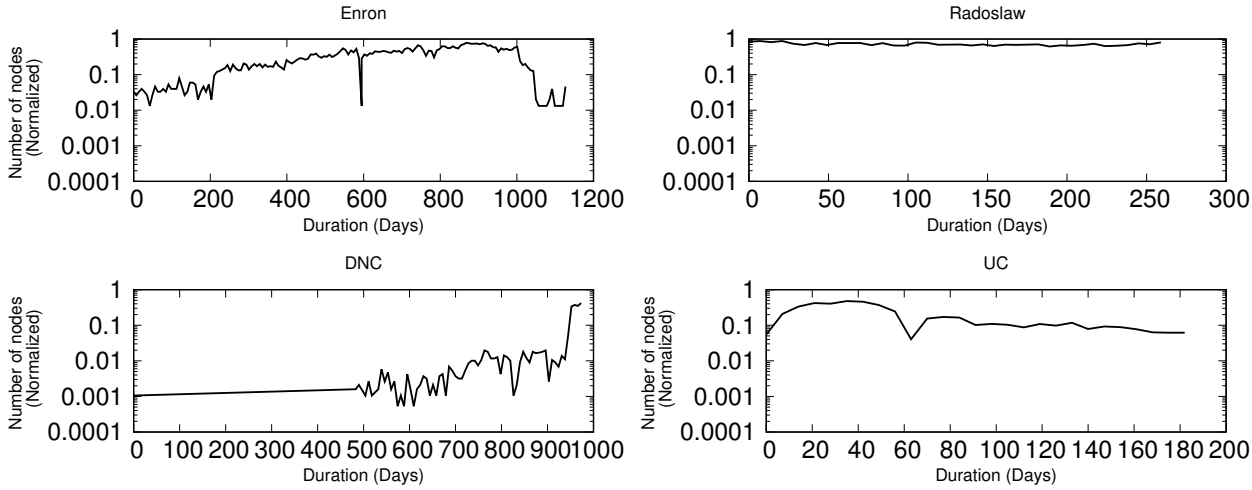



Figure 2.1: Evolution of the number of active nodes as a function of time for the four email based datasets.

exchanged during 6 months. In this case also, the messages can be sent to several students, which corresponds to multiple interactions that have a common source and timestamp.

For each dataset, we study the evolution of the number of active nodes over time. We employ the snapshot method: the network is considered as a sequence of static graphs considered separately. We study the number of active nodes for each period[1]. Figure 2.1 presents the evolution of the activity for the four email-exchange datasets. In three datasets (Enron, DNC, UC-Email), we observe some peaks of activity, however most of the time the activity can be considered low. For example, this is quite obvious in DNC, where the activity is quite low for most of the duration of the dataset, except for the peak around the $900th$ day. Additionally, at most 700 members are active, yet the dataset contains a total of 1891 nodes. Finally, Radoslaw is the exception, at any given time, around 80% of the nodes are active.

1. See table 2.1 for the size of snapshot for each dataset in page 47.

Additionally, for each dataset, we study for each node the amount of time it is active[2] and the number of interactions that involve it. Figure 2.2 presents the correlation between the duration of activity and the number of interactions. Each dot in the figure represents a node. In En-

2. The difference between first and last interaction that involves the node.

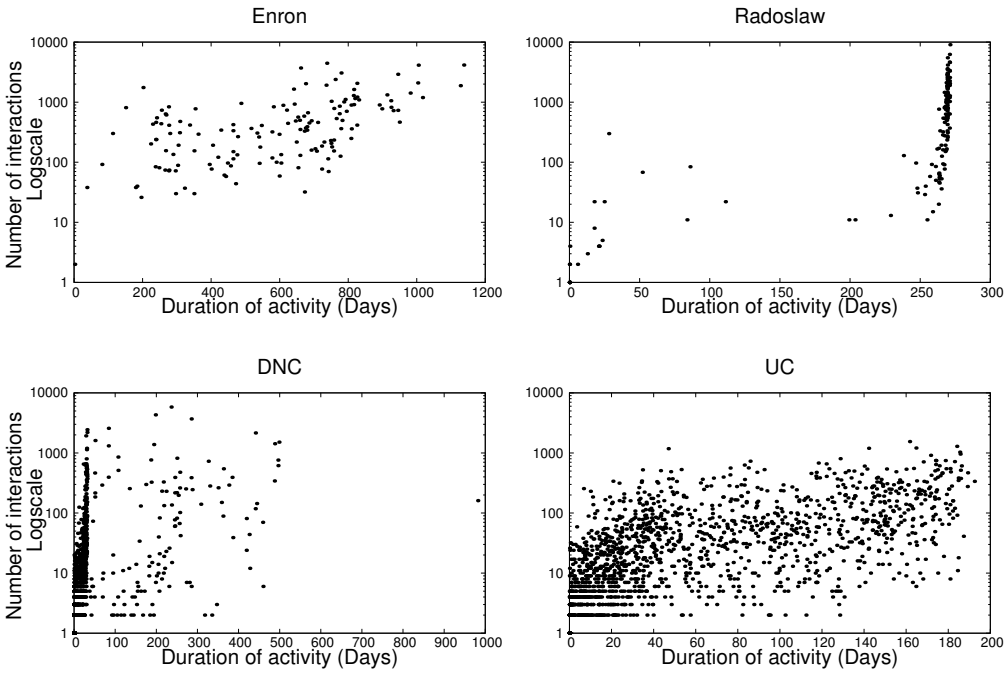



Figure 2.2: Duration of activity *v.s.* number of interactions for the four email based datasets.

ron, the values are quite distributed, additionally, the same node has the the largest number of interactions and longest duration of activity. However, globally certain nodes are active for a long period of time, without participating in a lot of interactions. On the opposite, certain nodes are active for much shorter times, yet they interact much more during this short period. In Radoslaw, most nodes have almost the same duration; however, they do not interact the same amount of times. The rest of the nodes have a smaller duration: at most one third of the dataset's duration. In DNC, we observe the same phenomenon as in Radoslaw, but most of the nodes are active for a short period of time and only one node has a high duration. In UC we observe values all over the spectrum.

Finally, for each node we study its degree in the aggregated graph. Figure 2.3 presents the inverse cumulative distribution of the degree for the four datasets in logscale. In Enron, one node comes in direct contact with half of nodes, while 90% of the nodes are in contact with less than 30% of the nodes during the whole dataset. In Radoslaw,

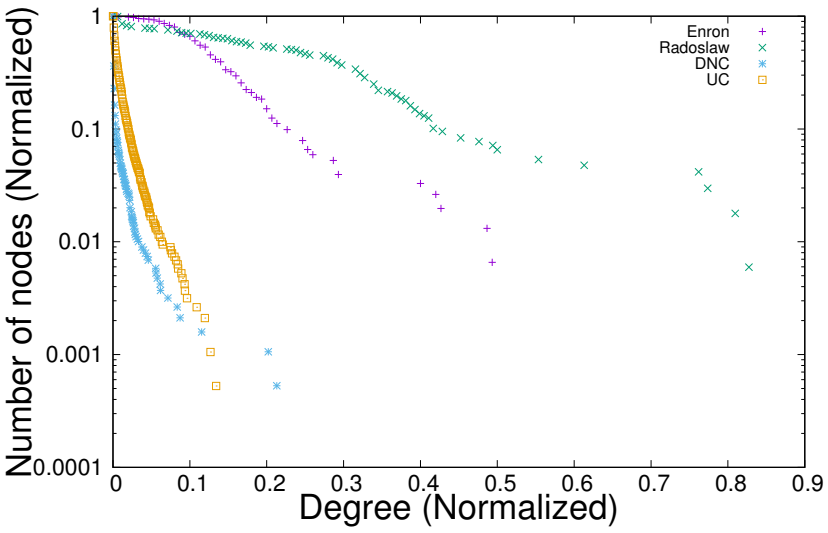



Figure 2.3: Inverse cumulative distribution of degree centrality normalized by total number of nodes for the email based datasets.

nodes have higher relative degrees: one node is in contact with over 90% of the nodes and 10% are in contact with at least 50% of the nodes. In DNC and UC, we observe low values of the degree, in both datasets the maximum degree does not reach 25% of the nodes.

## 2.2 Co-occurrence networks

The networks in this group do not necessarily come from the same source, however they share the same notion of link. A link here represents two entities being present together in a specific context, like for example two hashtags in the same tweet.

*HashTags*

This is a 22 day long twitter dataset[3] generated by twitter accounts known to be associated with terrorist groups. Each node represents a hashtag. When two hashtags are used in the same tweet, a link between both exists. Thus, a tweet with several hashtags generates several links. The dataset contains 3048 hashtags and 100,429 links.

3. This dataset is not publicly available.

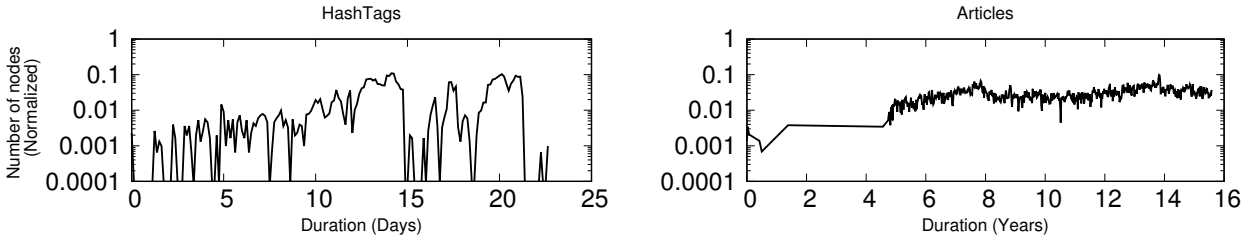



Figure 2.4: Evolution of the number of active nodes as a function of time.

*Articles*

A dataset provided by the Guardian newspaper[3]. Each article published on the guardian website has a set of keywords, for example, *Australia*. The keywords here represent the nodes of the graph. Each link $(u, v, t)$ represents two keywords $u$ and $v$ appearing on the same article at time $t$. During the 15 years, 2902 keywords were used, producing over $500,000$ links.

We start by studying the evolution of activity for these two datasets in figure 2.4. We should first note that due to the log scale, when the number of active nodes is zero, the values are not presented and we observe empty periods. The first observable phenomenon in both datasets is the low activity: the number of active nodes at any given instant is much lower than the total number of nodes. This is expected as keywords or hashtags are not constant and vary depending on the current situation (example: news of the day). Thus, here the nodes appear for a certain period and then disappear completely, which explains why the all the nodes are never active at the same moment.

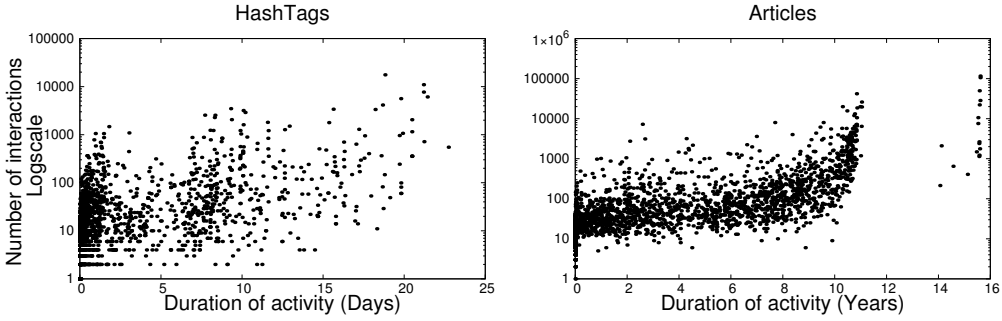



Figure 2.5: Duration of activity *v.s.* number of interactions for the two co-occurrence datasets.

Now, we concentrate on the duration of activity and number of interactions for each node in figure 2.5. In Hashtags, we can observe values all over the spectrum. Certain nodes are present for really short periods (less than 1 day), however they have a lot of interactions. These nodes are probably related to a specific event that everyone tweets about for a short period of time. In Articles, we observe the same distribution except for certain nodes that are active for almost all the dataset, and at least three years more than the other nodes. Manual investigation showed that these nodes correspond to general keywords, *e.g.* India, Plants, Animals. From the activity profile (figure 2.4) and duration of activity (figure 2.5), we can conclude that nodes become active for the first time at different times. For example, in Articles, only a small number of nodes are active for the whole duration of the datasets, yet the global activity remains quite constant.

Figure 2.6 presents the inverse distribution of the degree centrality. For HashTags, the nodes are rarely in direct contact with many others. The node with highest degree is in contact with around 20% of the nodes. This is not the case in Articles, the node with highest degree is almost in contact with all the others, and globally all the nodes have a higher degree than that observed in HashTags. We estimate that this due to fact that hashtags can be written in different ways, however in Articles as the keywords are chosen by the editors they are more likely to be written in the same manner and be more general.

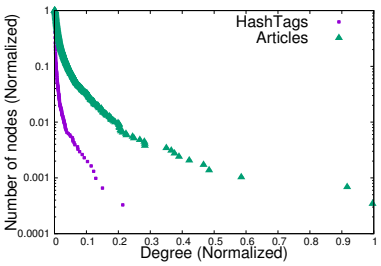



Figure 2.6: Inverse cumulative distribution of degree centrality for the co-occurrence based datasets.

## 2.3 Social Networks

All networks in this group come from similar sources, such as Twitter and Facebook. Here the nodes of the datasets can represent social accounts, such as Facebook account or Twitter accounts.

*Retweets*

From the same twitter dataset used in HashTags, we extracted the subset of 27,919 re-tweets generated by the 10,484 twitter accounts. Each link $(u, v, t)$ represents a user $v$ re-tweeting a tweet of user $u$ at time $t$.

*Facebook*

This dataset is a 1 year long record of the activity of Facebook users between 31st of December 2015 and 31st of December 2016 [Viswanath et al., 2009]. The dataset contains 8 977 Facebook users and their 66 153 posts on each other's wall on Facebook. The nodes of the network are users, and each link represents a user writing on another user's wall.

*Bitcoins*

Bitcoin is a cryptocurrency that is used to trade anonymously over the web [Kumar et al., 2016]. This anonymity can lead to fraud quite easily, as finding the person physically or his identity after a transaction is near impossible. Thus after transactions, user's can rate one another evaluating their trust. Each link $(u, v, t)$ represents $u$ rating $v$ at time $t$. This dataset contains 24,186 interactions between 3783 accounts during a period of 5 years.

Now, we study the evolution of activity for the three datasets in figure 2.8. As observed with the co-occurrence networks, the number of active nodes is always low compared to the total number of nodes. Figure 2.7 presents the inverse cumulative distribution of the duration of activity of nodes for the three datasets. In Retweets (resp. Bitcoins) just under 90% (resp. 80%) of the nodes are active for only 20% of the total duration of the dataset or less. The most plausible reason is likely people changing accounts in twitter or trading with the different accounts in Bitcoin.

Additionally, we can observe that in Retweets the activity starts extremely low and increases slowly, with drops at certain moments. In Facebook, the activity continues to increase over time, as new nodes are constantly appearing.

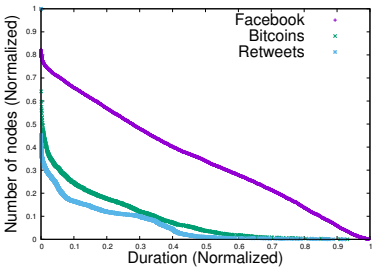



Figure 2.7: Inverse cumulative distribution of duration of activity of nodes for the social networks datasets.

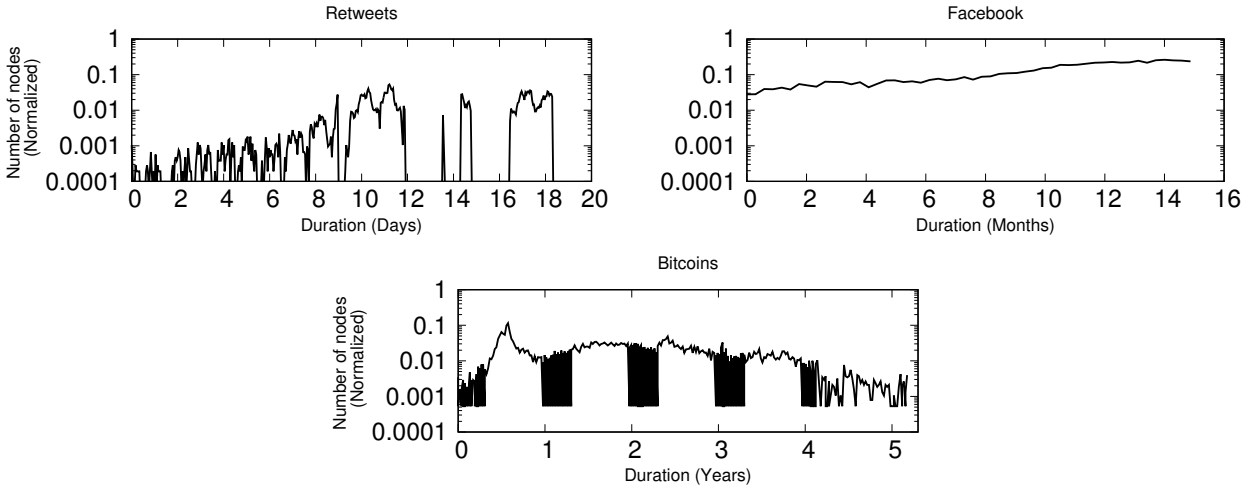



Figure 2.8: Evolution of the number of active nodes as a function of time for the social networks datasets.

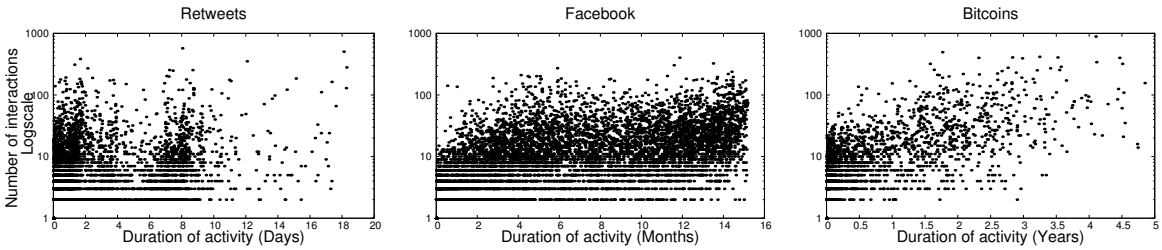



Figure 2.9: Duration of activity *v.s.* number of interactions for the three social networks.

Interestingly, in Bitcoins, the activity at the beginning of each year fluctuates quite strongly then becomes stable till the following year. This is likely due to a seasonal transaction period.

Finally, we study the duration of activity and number of links for each node in Figure 2.9. Again, as for like co-occurrence networks we observed values all over the spectrum. We are not able to point out a specific profile: some nodes are active for a short period, but they have a lot of interactions; some nodes are active for long periods with a small number of interactions; and nodes are in between. Also, we observe that the node with highest number of interaction is not the node with the longest duration and vice-versa.

We finally study the inverse cumulative distribution of the degree in figure 2.10. We can observe that the values

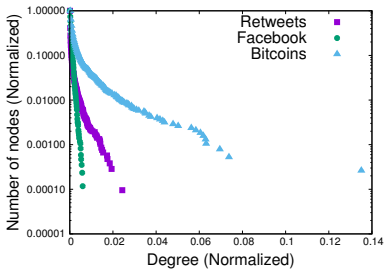



Figure 2.10: Inverse cumulative distribution of degree centrality for the social based datasets.

are much lower than those observed in the other groups of dataset. Again this is not surprising: in Facebook, individuals communicate only with their friends; for a node to have a degree close to 100% of the total number of nodes, it would need to write on the facebook wall of 8977 different individuals. In Retweets and Bitcoins the values are higher but remain quite low; again this can be related to the nature of the datasets and how users behave.

## 2.4 Motion networks

In this group of networks, all the nodes represent individuals, participating in an event or being present at the same location for a certain period of time. Two individuals at close proximity at a certain time are represented by a link here.

*RollerNet*

This dataset was collected during a rollerblading tour in Paris in August 2006 [Tournoux et al., 2009]. The tour is a weekly event and gathers approximately 2500 participants. It takes place in the streets which implies acceleration and speed reductions due to traffic lights for example. Among the participants, 62 were equipped with wireless sensors recording when they were at a communication distance from one another. These sensors use bluetooth technology to look for neighbors every 15 seconds. The dataset therefore contains the proximity links between the individuals carrying the sensors. The total dataset duration is approximately 2 hours and 45 minutes (note that there is a break of approximately 30 minutes during the tour).

*Reality*

During a period of 9 months, 96 students from the Massachusetts Institute of Technology (MIT) [Opsahl and Panzarasa, 2009] participated in a data collection experiment. The students' phones registered each time they came in contact with another student. From September 2014 to *May* 2015, the 96

students came into contact over 1,000,000 times.

*Taxi*

For 30 days, 320 Taxi in the city of Roma had a tablet device that sent the GPS location every 7 seconds [Bracciale et al., 2014]. From this dataset, we produced a network in which every link represents two taxis coming into near proximity, 30 meters from one another. In the analysis we only considered an 8-hour interval, the duration of a taxi driver's shift. During this 8 hour interval, 131 taxis were active and had over 80,000 contacts.

*Primary*

242 students and teachers in a primary school participated, from the beginning of a working day till the end of the next working day [Gemmetto et al., 2014]. They wore a wireless sensor that records the proximity between each other. A link between any two individuals is registered if the individuals are in proximity. The capture is made every 20 seconds. Thus, a 1 minute encounter produces three links. We should note that as this dataset was recorded over 2 days, it includes the evening and night. As during this time each individual returns to his home, a period of total inactivity exists. Thus, we produced a second dataset that represents the first working day, that we call **Primary-day1**. The total dataset contains over 120,000 interactions, from which around 60,000 occurred in the first day.

*HyperText*

During the ACM Hypertext 2009 conference, the conference attendees volunteered to participate in a data collection experiment [Isella et al., 2011]. Around 100 participants wore radio badges that monitored their proximity over about 2.5 days. This produced around 20,000 links.

*Infocom*

Similarly to HyperText dataset, a 4 days an experiment was conducted during Infocom 2006 in Barcelona [Chaintreau et al., 2007].

98 devices were deployed among which 78 were participants, 17 were placed at fixed points and 3 have been placed in the elevators. They scanned their surroundings every 120 seconds, to detect the other devices in proximity. This produced around 100,000 links.

Finally, we study the evolution of activity for these six datasets in figure 2.11. We shall study each dataset in order. First of all, we can observe in RollerNet and Reality, that most of the nodes are constantly in contact with the others during the whole duration of the datasets. The one exception being the drop in activity in Reality, that represents the end of year holidays. In Taxi, the nodes are never all active at the same instant; this is due to the fact that the drivers' shifts do not necessarily start at the same time, hence the nodes are not active at the same moment. In Primary and Hypertext, the evolution can be considered similar to Rollernet; almost all the nodes are active during the working hours and then become completely inactive during the evening. Finally, in Infocom, we can observe a similar evolution, where the activity decreases overnight, but never reaches zero.

Now we study the duration of activity and number of interactions each node has for the six datasets in figure 2.12. In RollerNet, all the nodes have a similar duration and number of links, except for two nodes. In other datasets, we can observe the same phenomenon to different extents. In each dataset, groups of nodes share the same duration and then have different values of number of interactions. The only exception to this is Taxi, where the values are quite distributed all over the spectrum.

Finally, figure 2.13 presents the inverse cumulative distribution of the degree for the six datasets. We observe several different profiles. For example for RollerNet most of the nodes are in almost in direct contact with all others nodes, while in the case of Taxi, the highest value is 30% of the total number of nodes. However, globally the values are much higher than those observed in the other classes of datasets. This is expected from the nature of the datasets,

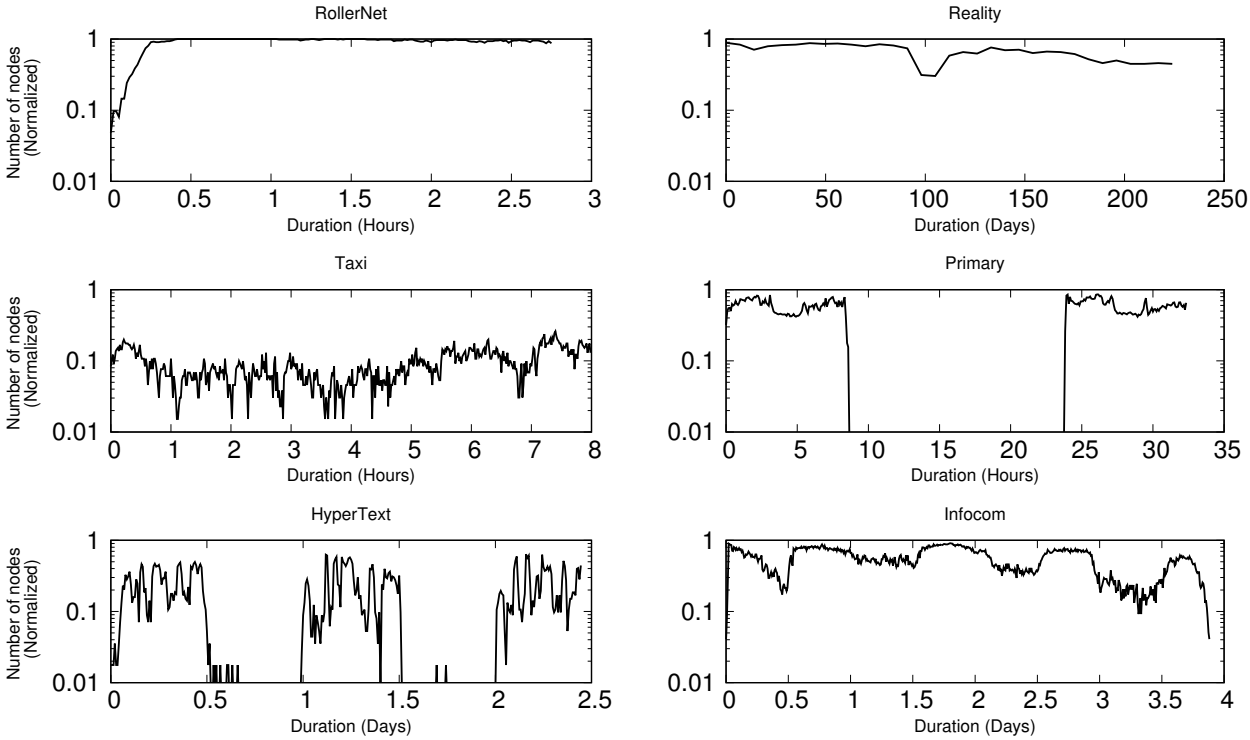



Figure 2.11: Evolution of the number of active nodes in function of time for the motion datasets.

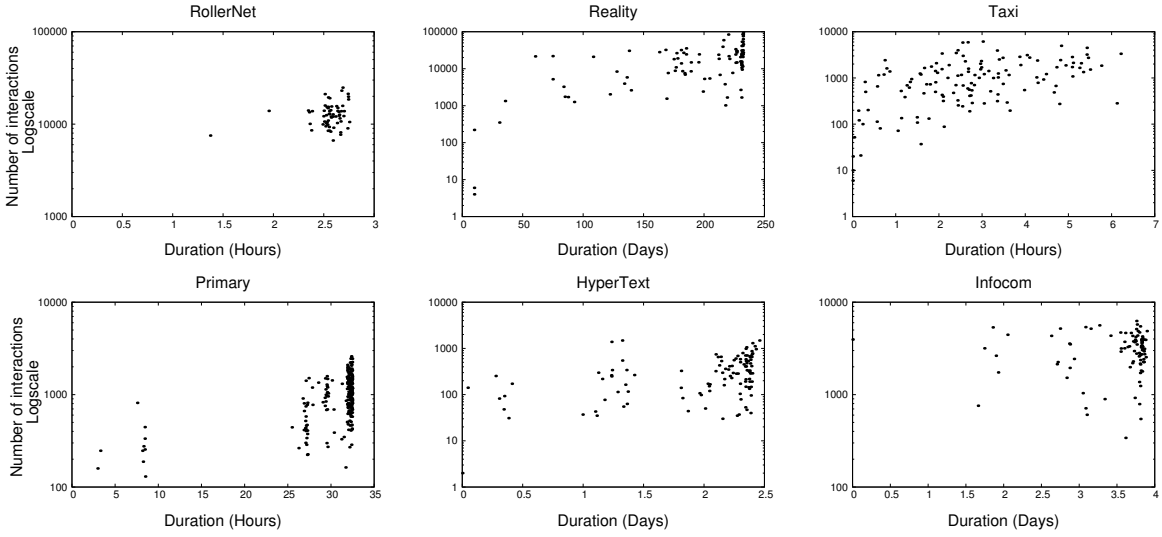



Figure 2.12: Duration of activity *v.s.* number of interactions for the motion datasets.

which usually consist of a group of individuals remaining in the same area for a period of time. This naturally increases the chances of interacting with the others.

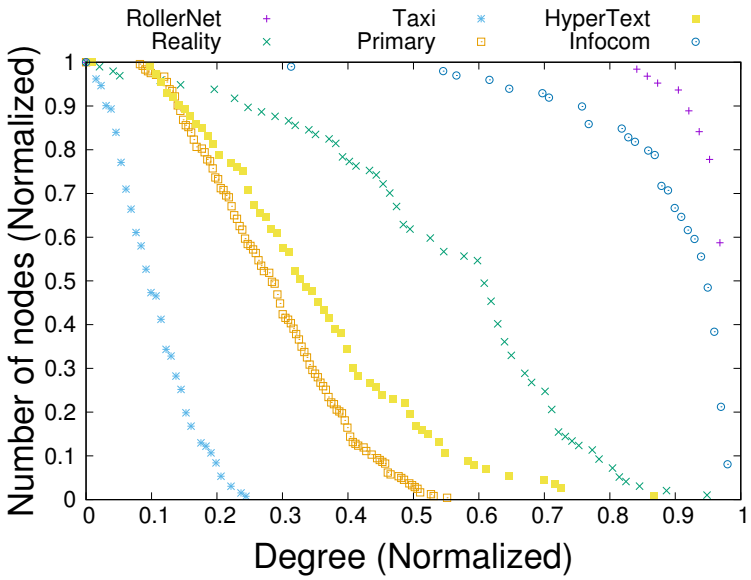



Figure 2.13: Inverse cumulative distribution of the degree for the motion based datasets.

## 2.5 Conclusion

To conclude, a total of fifteen datasets were presented. Four datasets have an email exchange nature, two have a co-occurrence nature, three came from social networks and the remaining six are motion datasets. They all vary in the number of nodes, number of interactions and duration. Certain datasets are quite dense such as RollerNet and some are quite sparse such as Retweets. For each dataset, we had to compute the median inter contact time and snapshot size. These values as well as the basic properties of each dataset are present in Table 2.1.

| Datasets | $\|V\|$ | $\|E\|$ | Duration | Nature | Median inter contact time (Seconds) | Snapshot duration |
|---|---|---|---|---|---|---|
| **Enron** | 151 | 47 088 | 3 years | Email | 960 | 1 week |
| **Radoslaw** | 168 | 82 876 | 9 months | Email | 53 | 1 week |
| **DNC** | 1891 | 39 264 | 2 5 years | Email | 57 | 1 week |
| **UC** | 1899 | 59 835 | 6 months | Email | 38 | 1 week |
| **HashTags** | 3 048 | 100 429 | 22 days | Social | 16 | 3 hours |
| **Retweets** | 10 484 | 27 919 | 20 days | Social | 18 | 1 hours |
| **Facebook** | 8 977 | 66 153 | 1 year | Social | 278 | 1 week |
| **Articles** | 2 902 | 571 877 | 15 year | Social | 9134 | 1 week |
| **Bitcoins** | 3 873 | 24 186 | 5.2 year | Social | 86400 | 1 week |
| **RollerNet** | 62 | 403 834 | 3 hours | Motion | 4 | 60 seconds |
| **Reality** | 96 | 1 063 063 | 9 months | Motion | 600 | 1 week |
| **Taxi** | 131 | 85 732 | 8 hours | Motion | 1 | 60 seconds |
| **Primary** | 242 | 125 773 | 32 hours | Motion | 20 | 60 seconds |
| **Primary-day1** | 242 | 60 611 | 8.6 hours | Motion | 20 | 60 seconds |
| **HyperText** | 113 | 20 818 | 2.5 days | Motion | 20 | 60 seconds |
| **Infocom** | 98 | 155 524 | 4 days | Motion | 20 | 60 seconds |

Table 2.1: number of nodes $|V|$, number of links $|E|$, dataset duration, median of inter-link time and snapshot duration for each dataset.

# Chapter 3

# Temporal Centralities under scrutiny

As we observed in Chapter 1, several methods were introduced to adapt existing centrality measures for the temporal case. Some adaptations were introduced to modify specific centrality metrics. Others introduced global solutions that can apply to any centrality metrics. The range of techniques used in those adaptations are wide and there is a lack of understanding on how these techniques differ and perform against one another.

In this chapter, we introduce a framework to compare the centrality metrics. We apply it to four centrality adaptations. We investigate how these four methods differ on both the global and node level on several real world datasets.

## 3.1 Centralities

In this section, we present the four centrality methods that we consider. These four centralities are emblematic of the adaptations that can be found in the current literature. These adaptations consist of either redefining the notion of path, representing the graph in an aggregated form or splitting the graph into several snapshots. We focused on centralities that respect several aspects: they have to attribute each node a centrality value at each instant; they

must be computationally feasible; they must take into account all the temporal information; notice also that we are not interested in updating a classic centrality metric as the graph evolves, but in metrics aiming at describing the evolution of nodes centrality.

### 3.1.1 Temporal Closeness

The first method was presented by [Magnien and Tarissan, 2015]. A dynamic network $G = (V, E)$ consists of a set $V$ of nodes and a set $E$ of timed links of the form $(u, v, t)$ where $u, v \in V$ and $t$ is a timestamp. In this formulation, the set of nodes does not evolve over time. We should note that for this method and all the others, we consider the networks as undirected. For example a link $(u, v, t)$ is equivalent to a link$(v, u, t)$.

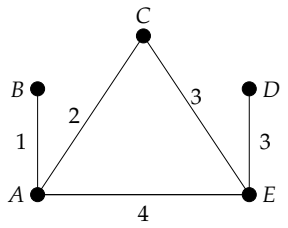



Figure 3.1: A toy example of a dynamic graph. The labels on the links indicate the instant in which the link occurs.

The authors use the notion of temporal path. A temporal path from $v_0$ to $v_{k+1}$ consists of:

- a starting time $t_s$;
- a sequence of links $(v_0, v_1, t_0), (v_1, v_2, t_1), \cdots, (v_k, v_{k+1}, t_k)$;

such that:

1. $\forall i, i = 0 \cdots k-1, t_i < t_{i+1}$;
2. $t_0 > ts$.

This path starts at $t_s$, and its *duration* is equal to $t_k - t_s$. A path from $u$ to $v$ starting at time $t_s$ is a shortest path if it has the shortest duration among all paths from $u$ to $v$ starting at time $t_s$. From this, the temporal distance between $u$ and $v$ at $t_s$ is defined as the duration of the shortest path from $u$ to $v$ at $t_s$, and is denoted by $d_t(u, v)$. In the same fashion as the classic path definition (see section 1.1.1), the distance is equal to infinity when there is no path between the two nodes starting at a specific instant.

To get a better understanding of this definition, we observe in Figure 3.1 a toy graph with five nodes and five links: the labels on the link represent the instant in which the link occurs. If we consider the nodes $B$ and $E$, we observe two paths starting at time 0. The first ($B \rightarrow A \rightarrow$

$C \to E$) departs at $t = 1$ and arrives at $t = 3$, its duration is equal to 3. The second path ($B \to A \to E$) departs at $t = 1$ as well; however, it arrives at $t = 4$, thus, its duration is equal to 4. This toy example shows that while the first path contains structural links than the second path, its duration is shorter than that of the second. Thus, the first path is considered the shortest path between $B$ and $E$ starting at $t = 0$.

Before we introduce the temporal closeness definition, we recall the definition of closeness centrality for node $u$ in non-evolving graph defined in [Bavelas, 1950] as:

$$C(u) = \sum_{v \neq u} \frac{1}{d(u,v)} \tag{3.1}$$

where $d(u,v)$ is the classical graph distance (see Section 1.1.1 for more details).

Replacing $d(u,v)$ by $d_t(u,v)$ gives the temporal closeness definition:

$$C_t(u) = \sum_{v \neq u} \frac{1}{d_t(u,v)}, \tag{3.2}$$

This definition of $C_t(u)$ requires to compute the value for each time instant $t$. To reduce the computational needs, we suggest computing the temporal closeness every $I$ seconds. The value of $I$ is based on median of inter-link duration, which is the time separating two consecutive links.

### 3.1.2 Snapshot

The second method was proposed by [Uddin et al., 2013]. They introduced a framework that represents a temporal network $G_t$ as a sequence of static networks known as snapshots. Each snapshot represents an aggregation of all the interactions for a given period of the temporal network. Given a static centrality metric, each snapshot is analyzed separately as a normal classic graph. Thus, the computed centrality value for a node $u$ during a snapshot represents the node's importance during the whole period that the snapshot represents. Figure 3.2 represents the toy example

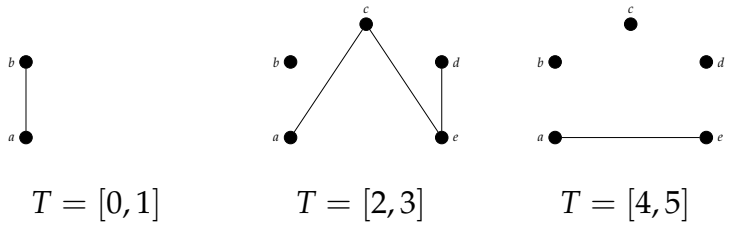



Figure 3.2: A snapshot representation of the dynamic graph in figure 3.1. Each Snapshot represents a period of two instants.

of Figure 3.1 in the form of three snapshots, each representing a period of duration 2. If one wants to compute the closeness centrality for the node $a$, the centrality would be computed on each snapshot individually.

For comparison purposes, we consider that the value of a node's centrality is constant during the whole snapshot. Here, we use this framework with the classic closeness definition, which we denote from now on as *SnapshotCl*.

### 3.1.3 Temporal Eigenvector

This method was introduced by [Taylor et al., 2017]. Unlike the two previous methods that base the importance of a node on the notion of the shortest path between nodes, this third method is a spectral one, and thus is based on paths of all lengths. The network is split into periods representing each an equal duration of the network. For each period, the authors construct an adjacency matrix $A$. The adjacency matrix at time $t$ is given by $A^t$. They combine these matrices into a large matrix called *supra-centrality* matrix. Its size is $NT \times NT$, where $N$ is the number of nodes and $T$ is the number of time periods. The diagonal contains the adjacency matrices constructed for each period. The remaining of the *supra-centrality* matrix contains an identity matrix weighted by $\omega$ parameter. See figure 3.3 for an illustration.

This weight $\omega$ couples each node with itself over the different periods. This allows the representation of the waiting times, where a node keeps information till it can pass it forward. When $\omega \to 0^+$, the layers are uncoupled and

$$A = \begin{bmatrix} A^0 & \omega I & \omega I & \cdots \\ \omega I & A^1 & \ddots & \ddots \\ \ddots & \omega I & \ddots & \ddots \\ \ddots & \ddots & \ddots & A^t \end{bmatrix}$$

Figure 3.3: Example of a *supra-centrality* matrix, with $A^t$ represents the adjacency matrix of instant $t$.

a node does not store the information over several layers. When $\omega \to \infty$, the layers (periods) are strongly linked, the weights on these inter-layer links becoming higher than those inside each period. Thus, the inter-layer links become more important than the intra-layer links. In other words, paths that consist of storing the information with same node for a long time become more important than those that consist of passing the message forward right away.

With this *supra-centrality* matrix, the eigenvector centrality can be computed. This computation would consider all the temporal aspects of the network such as the diffusion of information over time. It also attributes each node a centrality value for each period. We consider this method with a period equal to $I$ and denote it as *Temporal Eigenvector*.

### 3.1.4 Coverage Centrality

The fourth method and considers the temporal aspect in a different manner and was proposed by [Takaguchi et al., 2016]. They represent a temporal network as a static network where each *temporal node* consists of a pair of a *structural node* (original node of the network) and a timestamp $t$. For example, a node $u$ that interacts with other nodes at $t_1$ and $t_2$ will be represented by two temporal nodes $(u, t_1)$ and $(u, t_2)$. These two nodes are then linked, to simulate the retaining of information within the same node for a period of time. The second type of links represents the true interactions of the network. In a graph, if the path $(u, v, t)$ exists, it is represented by a two link, one from node $(u, t)$ to $(v, t+1)$ and a second one from node $(v, t)$ to $(u, t+1)$. See Figure 3.4 for illustration.

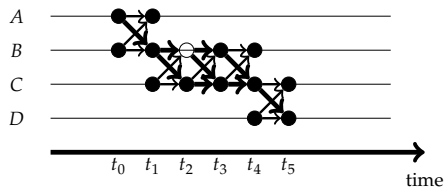



Figure 3.4: A toy graph in the form of a *direct acyclic graph.*

Building on this representation, the authors introduce two notions of centrality. First, *temporal coverage centrality* represents the importance of a temporal node $(u, t)$ by the fraction of structural nodes for which a shortest path pass through the node $(u, t)$.

In certain cases, a node can have several opportunities to pass the message, and all these opportunities produce

the same result (the same departure and arrival time). In Figure 3.4, if information is diffused from $A$ to $D$, we can observe that node $B$ has three occasions (at $t_1, t_2$ and $t_3$) to pass forward the information. If $B$'s presence at $t_1$ is removed all the the paths from $A$ to $D$ are destroyed. However, if the $B$ is removed at $t_2$ (white node in the figure), a path still exist via $B$ at instant $t_3$. While, the presence of $B$ at $t_3$ is the last chance for $B$ to forward the message. The authors argue that instants where the node can be removed without altering the diffusion need to be detected. Thus, they introduce the *temporal boundary coverage centrality*, which gives higher importance to nodes on the boundary (node $B$ at $t_1$ and $t_3$) than the nodes that can be removed (node $B$ at $t_2$).

Our preliminary results showed that both centralities reacted in the same way and the difference in results is not significant. Thus, we consider the *temporal coverage centrality* in this chapter. We denote it by *Coverage*.

## 3.2 Comparison protocol

In this section, we introduce a formal method that compares any two centrality metrics. This method compares these metrics on two levels. First, on the global level, we study how each metrics globally evaluates the importances of nodes. Second, on the node level, we study how each node's global importance is perceived by the metric as well as the evolution of this importance overtime. Before presenting this comparison method, we discuss certain tools that are required to facilitate this comparison.

### 3.2.1 Ranking

A node's centrality quantifies it's importance. However, inevitably this value alone does not indicate the node's significance compared to the others. For example, a node with a degree centrality equal to 100 is not significant if another node in the network has a degree centrality of 1000. On the other hand, this node is significant if no

other node in the network has a degree centrality higher than 100. Thus, we can see that the net value of centrality is not necessarily informative without any knowledge about the other nodes' centrality values. Furthermore, if two centrality metrics attribute different values to the same node, comparing these values without any knowledge of the other nodes' centrality values is meaningless.

To be able to take into account the centrality values of the remaining nodes, we consider the node's rank rather than its centrality value. This remains a numerical figure and it indicates how this node performs against all the others. Thus from a node's rank, we can deduce the significance of the node in comparisons to the remaining nodes of the network. To rank the nodes, we consider the *inverse competition ranking* method. This method is defined as follows:

1. The rank 0 is always attributed to the least important nodes;

2. The ranking of each node is equal to the number of less important nodes.

$$\underbrace{A = B > C = D = E > F}_{\text{Centrality Relation}} \longrightarrow \underbrace{4 \quad 4 \quad 1 \quad 1 \quad 1 \quad 0}_{\text{Ranking}}$$

Figure 3.5: Centrality relation of six nodes and the ranking.

To get a better understanding of how this method ranks the nodes. We consider the example in Figure 3.5, we observe six nodes, with the following relationships between their centrality values: $F$ is the least important, nodes $C$, $D$ and $E$ have an equal importance and finally $A$ and $B$ are the most important nodes with equal importance. The node $F$ evidently is attributed the rank 0 as it has the smallest ranking value. Afterward, the nodes $C$,$D$ and $E$ are assigned the rank 1 as their centrality value is only higher than that of $F$. Finally, nodes $A$ and $B$ are assigned the rank 4 as they are more central than the rest. It is easy to see that knowing that $A$'s rank is 4 gives more indication that it's centrality value.

With the ranking of nodes, we get a better image of how central each node is compared to the others. For all the instants, we rank all the nodes and produce a ranking vector. For all the centrality metrics we consider, the *nth* element of this vector gives the rank of the *nth* node in the dataset. We now discuss how to compare the ranking vectors of two centrality metrics.

### 3.2.2 Kendall Tau correlation

To compare two rankings, we calculate the Kendall tau correlation [Kendall, 1938]. The Kendall tau correlation measures the similarity between two rankings. The correlation is equal to 1 if both rankings are perfectly equal, and equal to $-1$ if the rankings are the inverse of one another; if both ranks are independent, the correlation is 0.

We start by defining a concordant pair. For two nodes $u$,$v$ and two ranking vectors $X$ and $Y$, if in both ranking vectors $u$ is more important than $v$ or vice versa, the pair $u$,$v$ is considered concordant. If this condition is not satisfied then they are considered discordant, in other words, ranking $X$ considers $u$ to be more important than $v$ and $Y$ considers $v$ to be more important than $u$. With these two notions, the Kendall tau correlation between two rankings $X$ and $Y$ is defined as:

$$\tau(X,Y) = \frac{\text{Number of concordant pairs} - \text{Number of discordant pairs}}{n(n-1)/2} \tag{3.3}$$

As each metric will be computed every *Ith* instant, the nodes' will be ranked at these instants producing a ranking vector for every *Ith* instant. We measure the Kendall tau correlation between these ranking vectors, producing the correlation between the strategies over time. The correlation provides a complete image of the evolution of the difference between strategies, showing how each method reacts to the different activities of the datasets.

### 3.2.3 Difference in ranks

Once each metric is computed on a dataset, each node has a rank that evolves overtime for each metric. The Kendall tau correlation described above does not give any insight on the difference in ranks between metrics on a given node; it only provides a global perspective of how all nodes are perceived differently by the metrics. To deepen our understanding of the differences between two metrics on the node level, we compute for every node the difference between the nodes' two ranks. For a given node and two ranks given by two different metrics, we compute the difference between the two ranks. A large difference indicates that the two rankings have a strong divergence regarding the importance of the node, while a value closer to 0 indicates that they both agree on its relative importance in the network. Similarly to the Kendall tau method, this method will be computed for all the time instants of the network, revealing how the difference evolves as time passes.

### 3.2.4 Global Importance

A node that is frequently attributed a high rank intuitively can, be considered as a globally important node. To provide a global perspective on the importance of a given node, we study the number of times the node is attributed a high or a low rank.

To compute this number, first, we need to be define a high and low ranks. We start by defining three regions in the ranking vector called the top, middle and bottom region. A rank in the top region is considered as a high rank, while a rank in the bottom region is considered as low. Formally, for a network of $n$ nodes, a node with a rank higher than $\lfloor n * 0.75 \rfloor$ is considered to be in the top region and thus, to be highly important. While a node with a rank lower than $\lfloor n * 0.25 \rfloor$ is considered to be in the bottom region and therefore, to have a low rank. Ranks in between are considered to be in the middle region.

As we compute the centrality and ranks for the nodes

every $I$-th second, the number of times each node is in the top (resp. bottom) region can be represented by a duration that we denote $Dur_{top}$ (resp. $Dur_{bot}$). To compute these durations, we consider that a node is present in the top region or bottom region from the instant where we compute the centrality to the following computing instant. Given a node $u$, if $R(u)$ is the sequence of ranks of $u$ and the $i$-th element of $R(u)$ denoted by $r_i$ gives the node's rank at the $i$-th instant, we formally define $Dur_{top}(u)$ and $Dur_{bot}(u)$ as:

$$\begin{aligned} Dur_{top}(u) &= I \cdot |\{i \leq k-1, r_i \geq \lfloor n * 0.75 \rfloor\}|, \\ Dur_{bot}(u) &= I \cdot |\{i \leq k-1, r_i \leq \lfloor n * 0.25 \rfloor\}|. \end{aligned} \tag{3.4}$$

From these values, for each node, we gain a better understanding of how each node is globally perceived by a centrality metric. Moreover, if we want to understand the difference between two centrality metrics, we can compute these values for a node using the ranking produced by each metric and compare the computed values.

## 3.3 Results

In this section, we compare how the different methods presented in section 3.1 quantify the importance of the nodes in the dynamic networks. We start by analyzing the global difference between the methods before evaluating how this impacts the relative importance of nodes. Finally, we compare how the nodes are globally perceived by the methods as well as which nodes are identified as globally important.

From the datasets that we presented in Chapter 2, we compared these centrality methods on six datasets: Enron, Radoslaw, Rollernet, HashTags, Retweets, and Facebook. However, we present the results for only three datasets, that are representative of the different phenomenas observed: Enron, Rollernet, and HashTags. We should also note that coverage centrality was too computationally expensive to be computed on any other dataset than Enron.

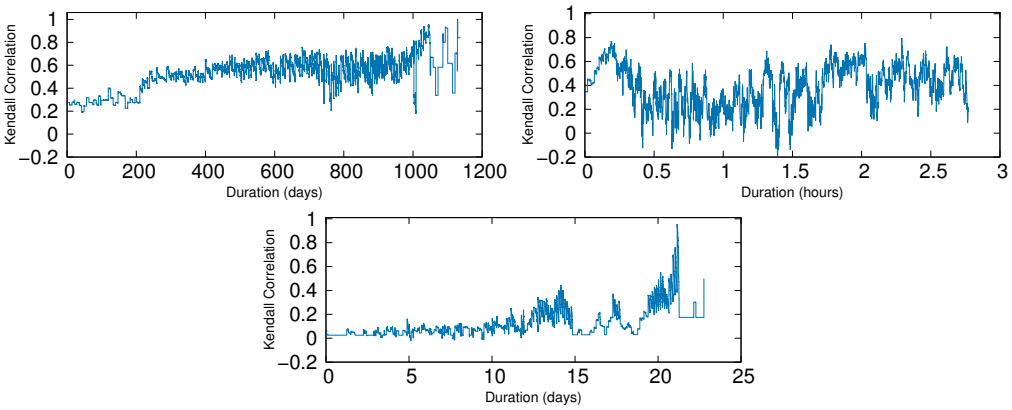



Figure 3.6: Evolution of the Kendall tau correlation between Temporal closeness and Snapshot over time for the three datasets. Top left: Enron. Top right: RollerNet. Bottom: HashTags.

### 3.3.1 Global Observation

The first two methods that we compare are Temporal Closeness and SnapshotCl. Figure 3.6 presents the evolution of the Kendall tau correlation between the rankings provided by the two methods for the three datasets. For Enron (Figure 3.6 top left), the correlation is quite low at first, near 0.2, as large number of nodes are inactive and SnapshotCl attributes the lowest ranking (0) to all of them. However, temporal paths involving links that appear later in the network exist for most of the nodes. Therefore, they have a non-zero centrality value and thus a non-zero ranking value is attributed to these nodes by temporal closeness.

As the network evolves, more nodes become active and they are taken into account by SnapshotCl, therefore the correlation between both methods increases. This phenomenon is not restricted to the beginning of the network. We observe that in certain cases there is a drop in correlation, for example around the 1000-th day. Manual investigation revealed that these drops are caused by the same phenomenon. At these instants, a significant number of nodes become inactive, and as SnapshotCl considers only the activity during a specific period, the nodes that are inactive during these periods, are attributed the rank 0 by SnapshotCl. However, as in the totality of the network they still participate in temporal paths, Temporal Closeness then attributes then non-zero ranks. We denote these instants as *temporary inactive moments*. Finally, we observe

how at the very end the correlation shoots and increases rapidly; this is due to the fact that nodes become permanently inactive and are ranked 0 by both methods, which decreases the difference between them. Thus, at the very end, a large percentage of nodes are inactive, increasing the correlation significantly.

In contrast with the Enron dataset, if we consider the RollerNet dataset (Figure 3.6 top right), we can observe two different properties. First, the correlation fluctuates highly. Second, the correlation is globally lower than that of Enron. Third, the correlation can be negative. These observations are clearly related to the high activity of RollerNet (See Chapter 2). This high activity generates snapshots much denser than in Enron. As the density increases, the number of paths increases, but also the number of paths that do not respect time increases. Thus, the snapshots contain a large number of these temporally impossible paths, causing the huge divergence with Temporal Closeness.

Finally, we focus on the HashTags dataset (Figure 3.6 bottom). As we observed in Chapter 2, the activity in this dataset is quite low at the start; the number of active nodes starts to increase only at the 14-th day. Thus, for the same reasons observed in Enron, the correlation is quite low till this time. Additionally, in Enron we observed drastic drops in the correlation due to drops in the activity and here we observe the same phenomenon. We recognize several periods with high correlation; these periods are due to increases in activity. We can conclude from these three datasets that the correlation between both methods is highly affected by the activity; extreme activity cases (too high or too low) decreases the correlation drastically.

Now we turn our attention to Temporal Eigenvector. Figure 3.7 presents the evolution of the correlation between Temporal Closeness and Temporal Eigenvector for the three datasets. First, in Enron (Figure 3.7 top left), the correlation fluctuates constantly. However, it globally follows an increasing trend over time, before dropping as the total ac-

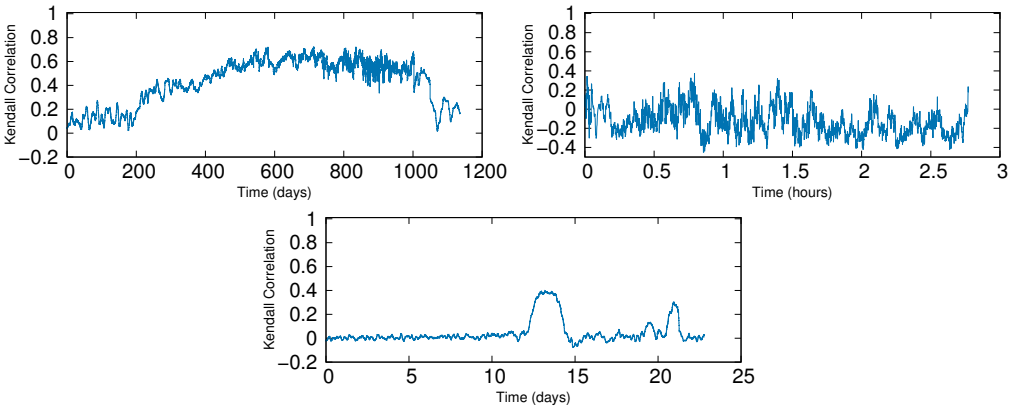



Figure 3.7: Evolution of the Kendall tau correlation between Temporal Closeness and Temporal Eigenvector over time for the three datasets. Top left: Enron. Top right: RollerNet. Bottom: Twitter.

tivity drops (around 1000-th day). From this, we can conclude that the correlation is linked to the level of activity; if the activity increases (resp. decreases), the correlation increases (resp. decreases).

More interestingly, the correlation drops at the very end, instead of increasing as observed with the snapshot method. Manual investigations here confirmed an observation made by [Fenu and Higham, 2017] that Temporal Eigenvector considers paths that do not respect the chronological order of links and therefore go backward in time. Thus, nodes becoming permanently inactive are still attributed a non-zero rank by Temporal Eigenvector.

In the case of RollerNet (Figure 3.7 top right), we observe a similar evolution to that witnessed with SnapshotCl. The correlation fluctuates excessively. However, it is much lower than that observed with SnapshotCl. Finally, in Twitter (Figure 3.7 bottom), we find a constant low correlation, except for two peaks that can be linked to an increase in activity. We can easily conclude that these two methods do not share the same notion of importance; their global vision on the does however become close in certain cases when the activity is at a certain level.

Finally, we compare Temporal Closeness and Coverage. Figure 3.8 presents the evolution of the correlation between both methods on the Enron dataset. We can remark how the correlation is very low and only increases at the very

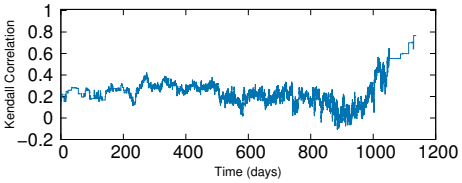



Figure 3.8: Evolution of the Kendall tau correlation between Temporal Closeness and Coverage over time on Enron dataset.

end. Even when the activity increases, the correlation remains low, unlike the other methods. This is a good indication that these methods detect two different notions of importance. Finally, as observed with SnapshotCl, the correlation slightly increases at the very end of the dataset. This increase is due to the fact that a small number of nodes remain active. This compensates for the difference in definition of importance, as being active becomes enough to be considered important by both methods.

From this first analysis, we can conclude that the correlation between Temporal Closeness and SnapshotCl is strongly related to the number of active nodes in the network. However, we observe two strong limitations of SnapshotCl: when the nodes become temporarily inactive, it is unable to detect the importance that future connections gives them; conversely, when many nodes are active, it considers many temporally impossible paths and therefore cannot quantify accurately the importance of the nodes. This likely causes an overestimation in the nodes' importance. Though the Temporal Eigenvector and Coverage do not have the same limitations, we have seen that they both detect different types of temporal importance compared to Temporal Closeness. We will investigate this further but it is worth noting that Temporal Eigenvector considers paths that may go backward as well as forward in time.

### 3.3.2 Impact on individual nodes

The previous section revealed that the four approaches generate significantly different rankings for the importance of nodes. However, this does not necessarily mean that the ranking attributed to a given a node is very different. In order to study this aspect, this section analyses the difference in the ranks provided by the four methods for each node. More precisely, for each time instant and for each node, we compute the difference between the ranking granted by Temporal Closeness and the one provided by either SnapshotCl, Temporal Eigenvector or Coverage. We then study

the distribution of the obtained values.

We start by comparing the difference between Temporal Closeness and SnapshotCl. Figure 3.9 (top) presents the inverse cumulative distribution of the difference of the ranks for each node at every instant and for the three datasets. For Enron and Twitter, Temporal Closeness attributes a higher ranking than SnapshotCl in more than 70% of the cases. This is in agreement with the conclusions drawn in the previous section that in temporary inactive moments SnapshotCl attributes a ranking 0 to a node, while the Temporal Closeness ranks it higher. However, in RollerNet which is a highly active network, the numbers of negative and positive values are more balanced. This is in contrast to what we observe when we compare Temporal Closeness and Temporal Eigenvector (Figure 3.9 middle). Except for a slight percentage in Rollernet, both methods almost never attribute the same rank. Finally, we consider the difference between Temporal Closeness and Coverage centrality (Figure 3.9 bottom). Interestingly, similarly to the comparison to the SnapshotCl, Temporal Closeness tends to attribute higher ranks than Coverage (around 70% of the nodes). Since Coverage is not affected by temporary inactive moments, this indicates that the Coverage measures the importance differently.

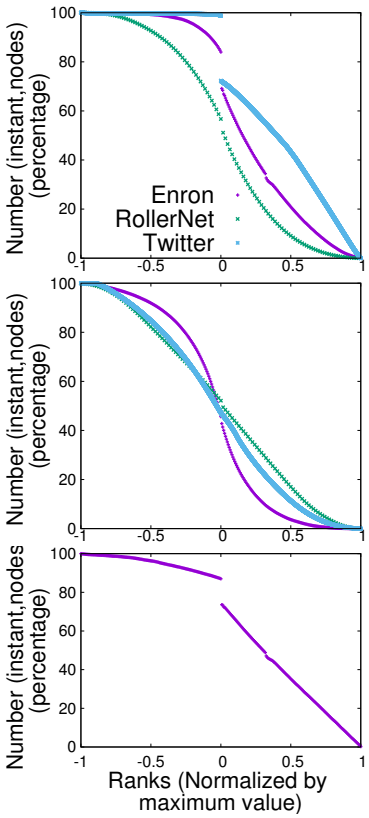



Figure 3.9: Inverse cumulative distribution of the difference in ranks between Temporal Closeness and other methods on the datasets. Top: SnapshotCl; Middle; Temporal Eigenvector and Bottom: Coverage.

We recall that SnapshotCl does not consider the temporary inactive moments. To get a better understanding of the importance of these moments, we study the ranks attributed by Temporal Closeness. Figure 3.10 presents the inverse cumulative distribution of the ranks for the three datasets. In Enron, the temporary inactive moments account for over 50% of all the (node,instant). In 20% of the (node,instant), nodes have ranks in the top region. A similar trend is observed with RollerNet; however this dataset is extremely active, so temporary inactive moments are rare. Finally, in Twitter, due to low activity the temporary inactive moments account for over 70% of the total number of (node,instant). In 30% of the (node,instant), nodes have a rank in the top region. As so many (node,instant)

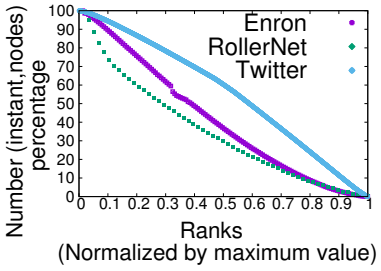



Figure 3.10: Inverse cumulative distribution of ranks attributed by Temporal Closeness during temporary inactive moments.

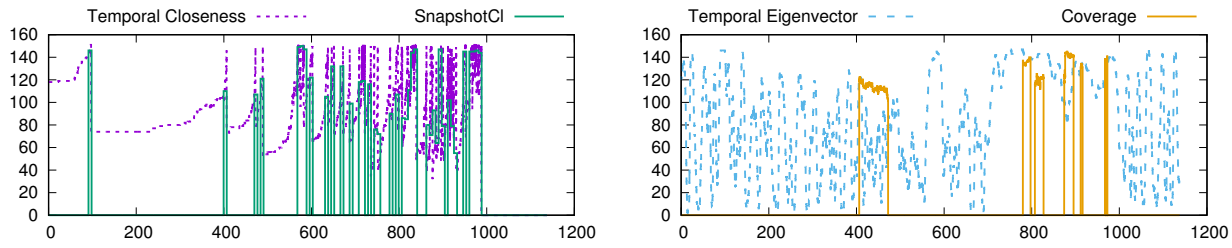



Figure 3.11: Evolution of ranks attributed by the four methods over time for a randomly selected node.

are inactive, this has an immense effect on the divergence between the two methods.

In order to illustrate the differences between the four methods, figure 3.11 presents the evolution of the ranking for a given node of Enron, for the four methods. We can observe that this node has many temporary inactive moments, where SnapshotCl ranking is equal to 0. In contrast, Temporal Closeness takes into account future communications and therefore attributes a ranking higher than 0 at these moments. This is particularly remarkable between days 100 and 400. The links occurring around day 400 give it a high Temporal Closeness and hence a high ranking not only at that time, but also influence previous times: even though the Temporal Closeness rank at certain instants, e.g., 300 is lower than at instant 400, it is still high enough to guarantee a significant ranking for this node. This is again in sharp contrast with SnapshotCl which perceives the node as unimportant for the whole period and clearly detect the role of the node only during peaks of activity.

In the case of the Temporal Eigenvector, although it can detect the importance during inactive moments, the ranks fluctuate extremely for no apparent reason. Quite interestingly, we observe in particular that, after the 1000-th day, although this node is no longer active, the ranking still fluctuates and at certain instants is very high. This confirms the observation we mentioned at the end of Section 3.3.1: the Temporal Eigenvector method considers paths that go backward in time (otherwise it would give a ranking 0 to this node). Even more strikingly, we see that

the importance of this node evolves in a nonmonotonic way even though it does not have any activity. This indicates that this method attributes centrality values in a non-intuitive way.

Concerning to the Coverage, although it is difficult to conclude on its relevance, the ranking evolution confirms that it captures a different notion of importance during the temporarily inactive moments. One can see in particular that between days 400 and 500 the Temporal Closeness and Coverage evolve in opposite directions: the Temporal Closeness ranking increases as the future links get closer, while the coverage ranking decreases and drops to 0 after the links have occurred.

The observations presented above confirm and refine the conclusions outlined in the previous section: not only do the four methods give different global rankings, but they also have strong differences for individual nodes.

### 3.3.3 Identifying globally important nodes

Although the results presented in the previous sections suggest that networks are highly dynamic and that nodes' importance varies over time, this does not mean that there are no nodes (or groups of nodes) that are globally important. Indeed, some nodes could stay important during a large period of the dataset. In this section, we investigate how long each node is considered important ($Dur_{top}$) and unimportant ($Dur_{bot}$) by each method.

Figure 3.12 (Left) presents the correlation between $Dur_{top}$ and $Dur_{bot}$ (defined in section 3.2.4) for each of the four methods, for the Enron dataset. We observe that certain nodes in Temporal Closeness are always important (resp. irrelevant); the $Dur_{top}$ (resp. $Dur_{bot}$) value is equal to the duration of the dataset. However, mainly the nodes have a wide distribution of values and thus a wide distribution of importance. This is in contrast with SnapshotCl, where we can observe the nodes are almost either in the top or bottom region ($Dur_{top} + Dur_{bot} \simeq Dur_{total}$)[1]. This indicates that ranks in the middle region ($\lfloor N * 0.25 \rfloor <$

1. $Dur_{total}$ is equal to the dataset's duration.

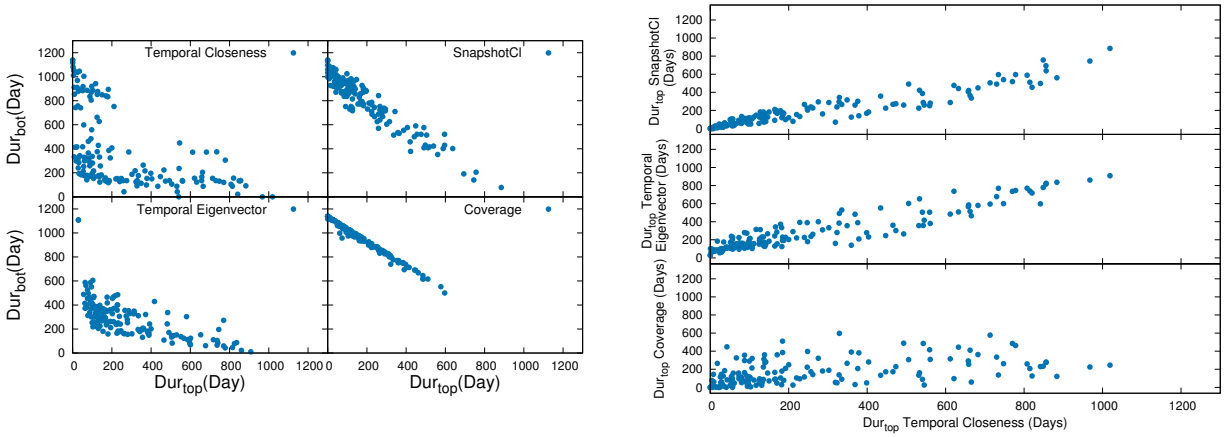



Figure 3.12: Left: For each node, the $Dur_{top}$ and $Dur_{bot}$ values attributed by each method on the Enron dataset. Right: For each node the $Dur_{top}$ value attributed by Temporal Closeness versus the $Dur_{top}$ attributed by the other methods on the Enron dataset.

$R < \lfloor N * 0.75 \rfloor$) are rarely attributed to nodes, which is well illustrated by the case presented in Figure 3.11. In Temporal Eigenvector, we observe that values are well distributed but are generally all lower than those attributed by the other methods. This can be an indication that the fluctuation seen in Figure 3.11 is a general phenomenon. Manual investigation showed that all nodes fluctuate in the same manner. In Coverage, we observe a similar distribution to that of the SnapshotCl, but the $Dur_{top}$ values are at most 600 days.

Despite these differences, one can see in Figure 3.12 (right) that the Temporal Closeness, SnapshotCl, and Temporal Eigenvector perceive similarly the global importance of nodes; the difference between the three approaches therefore lies mainly in how they evaluate unimportant nodes, as well as the nodes of average importance. However, we observe different results in Coverage, because the most important node for Coverage is of average importance for the other methods. Vice-versa the most important node in the three methods has an average importance in Coverage. This further supports our claim that coverage centrality detects a completely different notion of importance.

Interestingly, these observations are completely different for RollerNet. In this dataset, one can see in Figure 3.13 (left) that no node spends time in the top (or bottom) region more than half of the total duration, whatever the

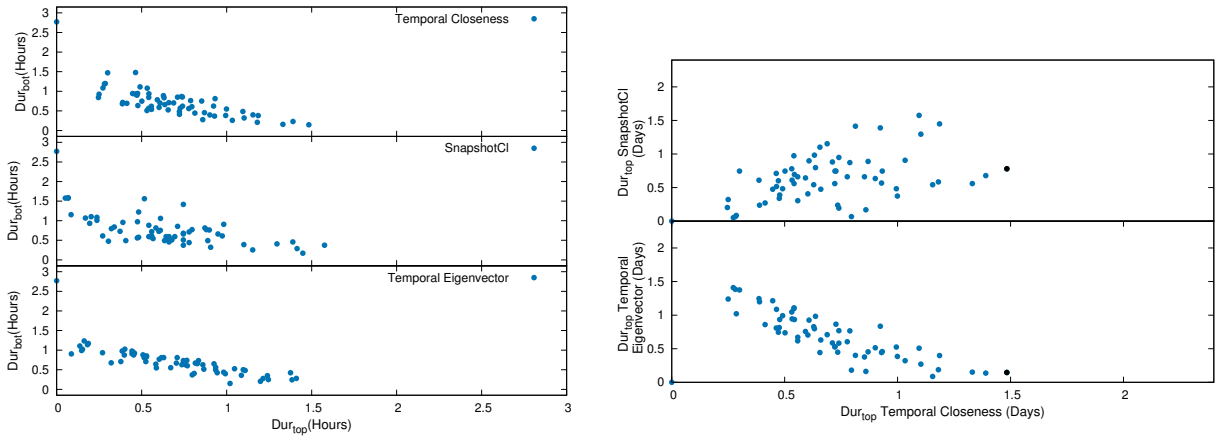



Figure 3.13: Left: For each node, the $Dur_{top}$ and $dur_{bot}$ values attributed by each method on the RollerNet dataset. Right: For each node the $Dur_{top}$ value attributed by Temporal Closeness versus the $Dur_{top}$ attributed by the other methods on the RollerNet dataset.

method used (except for one node that is almost always in the bottom region). Besides, when comparing the global importance attributed to nodes by different methods (evaluated by $Dur_{Top}$, Figure 3.13 right), one can see that the correlation between Temporal Closeness and the other methods is not very strong and is even anti-correlated for Temporal Eigenvector. All these observations indicate that in this dataset, the notion of global importance may not be meaningful. We claim that this is due to the very dense (both temporally and structurally) nature of this dataset.

The Twitter dataset is also interesting as it combines the behaviors previously seen in Enron for the four methods. Figure 3.14 (left) shows that the comparison between $Dur_{top}$ and $Dur_{bot}$ is similar to what we observed on Enron, and even more extreme: for SnapshotCl, points are all situated on the line $y = T - x$ (where $T$ stands for the total duration), meaning that at all times, any node is either in the top or the bottom region, but never in-between. Behaviors are more nuanced for the Temporal Closeness, and for Temporal Eigenvector, all the nodes have very similar values. This is similar to what we observed for Enron, but far more extreme. However, in contrast with Enron, the global importance (i.e. the $Dur_{top}$ value) attributed by Temporal Closeness is not at all correlated to the one attributed by the other methods (see Figure 3.14 Right).

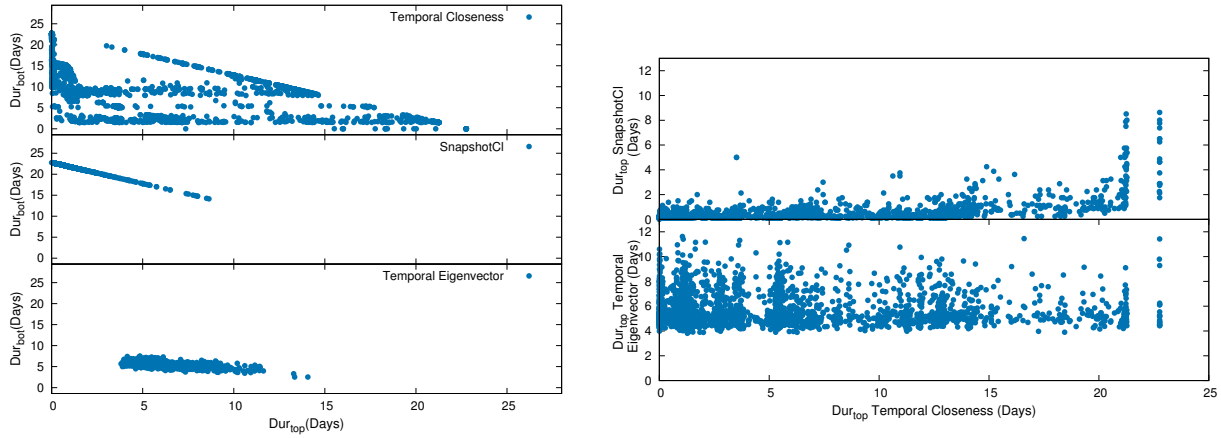



Figure 3.14: Left: For each node, the $Dur_{top}$ and $dur_{bot}$ values attributed by each method on the Twitter dataset. Right: For each node the $Dur_{top}$ value attributed by Temporal Closeness versus the $Dur_{top}$attributed by the other methods on the Twitter dataset.

Finally, we study the results obtained for Coverage centrality for the Enron dataset. We observe that Coverage is close to SnapshotCl in the sense that all nodes are almost always either in the top or in the bottom region (Figure 3.12, left figure, bottom right). However, as pointed out before, Coverage does not capture the same notion of importance as Temporal Closeness, which can be seen by comparing the correlation between the $Dur_{Top}$ values (Figure 3.12 right figure, bottom plot). The elements provided in this study do not allow to conclude on the reasons of the divergence observed with coverage and we leave this question for further studies. To be able to have a better understanding, obtaining datasets with a known ground truth would be a first step.

## 3.4 Discussion

### 3.4.1 Average centrality

In this work, we considered four centrality methods that quantify the time-evolution of nodes' importance. For the comparison of the values computed by each of the four centralities, we performed several steps. This included ranking the nodes with respect to their centrality values,

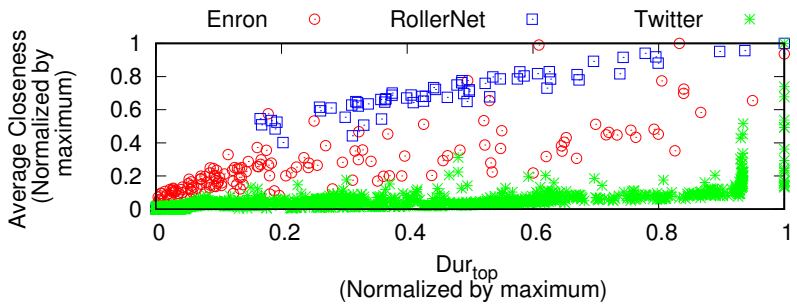



Figure 3.15: $Dur_{top}$ value versus the average Temporal Closeness centrality.

as well as computing a global duration that represents a node's global importance. [Kim and Anderson, 2012] proposed the study of the average temporal centrality rather than its evolution. They consider that this single value is representative of the node's complete evolution during a dataset.

To assess this claim, we propose to analyze the correlation between $Dur_{top}$ and the average Temporal Closeness. Figure 3.15 presents this correlation for the three datasets. In the case of Enron and RollerNet, these values seems correlated (particularly for RollerNet). However, the correlation is very low for Twitter: some nodes have a high average Temporal Closeness, yet a low $Dur_{top}$, and conversely. We argue that the average Temporal Closeness is not representative as it does not give each instant an equal amount of importance. As we saw, a node can have an extremely high Temporal Closeness at a single instant (and therefore a high average Temporal Closeness) even though it may have a very low activity (and hence, very low Temporal Closeness) in the rest of the dataset. However, the $Dur_{top}$ value considers that all instants have an equal importance.

### 3.4.2 Absence of importance

In section 3.3.3, we observed in RollerNet the absence of a notion of meaningful importance; all the nodes had quite close values of $Dur_{top}$. In this section, we investigate this further. We investigate twelve datasets of Chapter 2 to

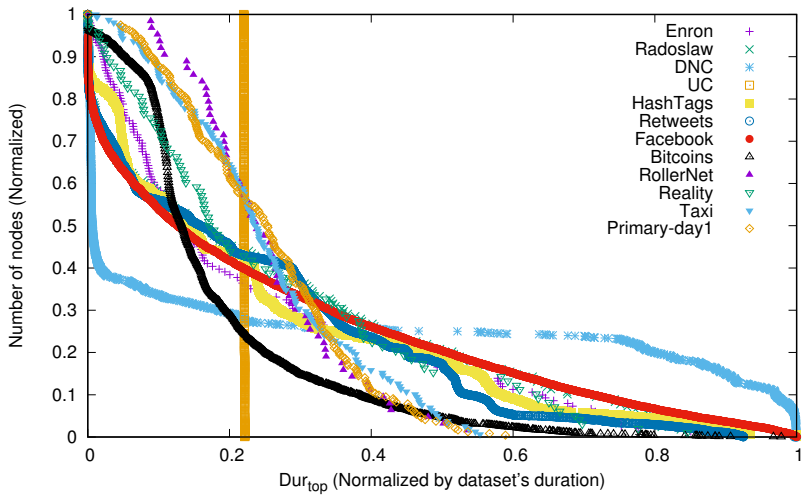



Figure 3.16: Inverse cumulative distribution of the $Dur_{top}$values attributed to the nodes normalized by the duration of the datasets for twelve datasets.

verify the presence or absence of importance. For each dataset, we compute the $Dur_{top}$ values of all the nodes and plot the inverse cumulative distribution. Figure 3.16 presents the distribution for the twelve datasets with the $Dur_{top}$ values normalized by the duration of each dataset. The first thing that we notice is the vertical line of UC. This indicates that all the nodes have a $Dur_{top}$ of around a quarter of the duration of the dataset. In other terms, all the nodes spend an equal duration in the top region. This is a clear sign that the absence of importance is not limited to RollerNet, and occurs in other datasets. Second, we observe that most of the datasets, have nodes with $Dur_{top}$ values that are equal to the datasets' duration. In other words, they contain nodes that are constantly important. Naturally, we can conclude that in these datasets, the notion of importance exists. Finally, we observe that the datasets Taxi and Primary have a similar distribution to that of RollerNet. The highest $Dur_{top}$ value is at most equal to 60% of the duration of the dataset. This means that in these datasets and RollerNet, not a single node becomes clearly more important than the others.

From this study, we can conclude that the notion of global importance is not always meaningful. This is similar to a case, where we would like to find the closest node to

all the others and find that all the nodes are at an equal distance from one another: pointing out the prime node would not be an obvious task. In certain cases, such as RollerNet or Primary the nature of the dataset explains this phenomenon. The individuals in these datasets have similar activity profiles, they all become active and inactive at the same moments and communicate between each other in a similar manner.

### 3.4.3 Ranking methods

In this chapter, we considered the *inverse competition ranking* method. However, a natural and common way to rank nodes is to simply order them by increasing value. Here, we discuss our choice of ranking method. The normal ranking method has two issues, let us consider the example in Figure 3.17. In the figure, we observe six nodes and the relationship between their centrality values. Certain nodes share the same centrality values.

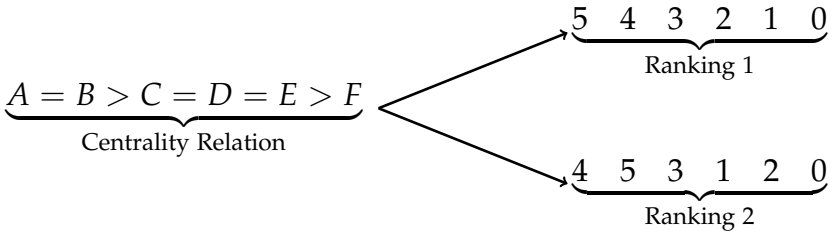



Figure 3.17: Centrality relation of six nodes and the two possible ranking possibilities.

The first issue is observed when nodes share the same centrality value: this method can produce several possibilities, hence the two possible ranking possibilities in Figure 3.17. Nodes with equal importance can be ranked in any configuration, as we observe $C$, $D$ and $E$ can be ranked in the order $3, 2, 1$ or $3, 1, 2$. Both ranks are formally correct, thus there is a lack of consistency in this ranking method. Similarly, node $A$ can be ranked higher than $B$ or conversely. This brings us to the second issue with this ranking method: it does not allow the detection of equally important nodes. In the example in figure 3.17, we notice how the ranking results do not display the equality in im-

portance between nodes $A$ and $B$ nor between nodes $C$, $D$ and $E$.

This might seem to be an unnecessary modification, as one would assume the chances of nodes sharing the exact same centrality value is rare. Thus, the standard ranking should be quite sufficient. However, in real-world networks this is far from the case. Figure 3.18 presents the evolution of the rank for six nodes randomly selected from the Enron dataset. The nodes were ranked in relationship to their Temporal Closeness values. In the top figure, the nodes are ranked using the *inverse competition ranking* and in the bottom figure the standard ranking was used. In the top figure, we can observe how the ranks decrease slowly towards the rank 0. This is due to the fact that the nodes become permanently inactive. We can observe that the standard ranking does not detect this phenomena at all. The more nodes become inactive, the more nodes have the same centrality value (0), the more nodes are attributed random ranks for the reasons mentioned previously. More interestingly, due to this artifact the rank of the fourth node ($Id = 3$), which is always inactive, increases at a certain moment. As other nodes become inactive, their centrality becomes equal to that of the fourth node. Thus, this node becomes affected by the random ranking phenomena and its rank increases. This effect is completely absent with the *inverse competition ranking*.

With this ranking of nodes, we get a better image of how each node is central compared to the others.

### 3.4.4 Conclusion

In this chapter, we proposed a methodology to compare centrality methods. We applied this methodology to four centrality methods. Each one of them had its unique technique to take the temporal aspect into account. We compared these method on both the single node and the global level and our observations can be summarized as follows:

1. different centralities produce different results; nodes that were identified as important by the Temporal Closeness,

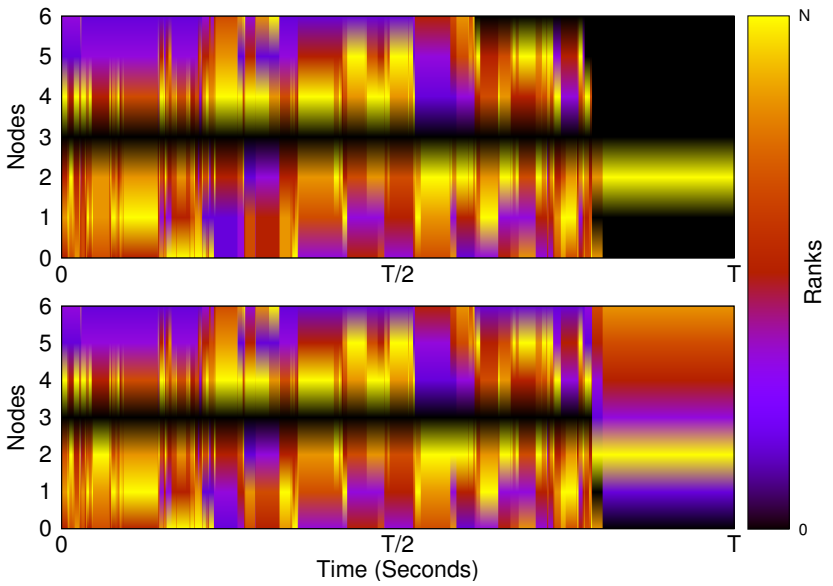



Figure 3.18: A heatmap in which each line represents the evolution of the rank for a node. Top figure: nodes ranked by *inverse competition ranking;* Bottom figure: nodes ranked by standard ranking.

were perceived as irrelevant by coverage centrality;

2. different datasets have different properties regarding the node's importance; for one of our datasets, the importance of all nodes fluctuates extremely rapidly between high and low values; it is meaningless in this case to state that one node is globally more important than another, except for a very limited time span; in other datasets however, we find that some nodes are consistently important for the whole network time span;

3. a node can be inactive (i.e. not have any links) yet be highly important since it can be a waiting point in an important temporal path between two nodes;

From these observations, we can draw a couple of conclusions regarding how these metrics perform. First, SnapshotCl is unable to relate a temporary inactive node to its future connections and it attributes this node a centrality equal to zero. Therefore, it does a poor job in quantifying the importance of a node as a relay for future communication. Another interesting conclusion drawn from this study is that Temporal Eigenvector is less accurate that Temporal Closeness. Both methods detected the same important nodes, however, Temporal Eigenvector considers

paths that go backward in time, which reduces its accuracy. Additionally, we can conclude that Coverage perceives a different importance than that of the others centralities.

Finally, during these comparisons, we introduced a notion of global importance that we compared to the average centrality. Our comparisons showed that nodes can remain important while having a low average centrality. Finally, we have observed the absence of global importance in the RollerNet dataset. Further investigations showed that is was not limited to RollerNet, but was found in four datasets, one of which where all nodes had a quite similar $Dur_{top}$value (one quarter of the dataset's duration). Thus, in certain cases the notion of global importance can be meaningless.

# Chapter 4

# Approximation and Identification

In this chapter, we investigate one of the main issues concerning temporal centralities: their high computational demand. For a node, a temporal centrality should be able to give the node's centrality at every instant in the dataset. Thus, the centrality has to be updated or recomputed from scratch for each of those instants, hence the high computational cost of these methods.

One of the main purposes behind computing the temporal centrality is detecting the globally important nodes. For this, the knowledge of how each node ranks compared to the others is sufficient. In other terms, for such a task the exact centrality values are not necessary. Here we propose different methods to reduce the computational demand for finding the highest ranked nodes for Temporal Closeness.

## 4.1 Less computation

In [Eppstein and Wang, 2001, Brandes and Pich, 2007], to reduce the computational time and approximate the centrality values, the authors sample out a certain number of nodes. Using these nodes, the authors approximate the centrality values by computing only the paths between or towards this set of nodes. Temporal Closeness offers more

possibilities to reduce the computational cost. In this section, rather than reducing the number of nodes that are considered in the computation, we reduce the number of instants. Temporal Closeness is computed every $I$ seconds. In other words, for each node, we have a centrality value every $Ith$ second representing the node's importance at this exact instant. We previously stated that an $I$ value equal to the median of inter-contact difference gives a good compromise between accuracy and computational cost. Later on, we denote this value by $I_{op}$. The higher the value of $I$ is, the lower the number of instants for which a computation is made, which translates to a lower computation time. However, a downside of this solution is the decrease in precision. As the value of $I$ increases the number of computing instant decreases, which affects the accuracy of $Dur_{top}$ (which is the time during which a node has a rank in the top 25%). After a certain decrease in accuracy, the global ranking of nodes becomes affected. To get a better understanding of this, we consider the artificial node in Figure 4.1. We observe the evolution of the ranking captured with two $I$ values. We can observe that at the $6th$ second, the node's rank crosses the top 25% limit. This increase is detected when $I$ is equal to 2, and not when $I$ is equal to 4. Therefore, when $I$ is equal to 4, the computed $Dur_{top}$ value will be lower than the true value; in other cases it could be higher.

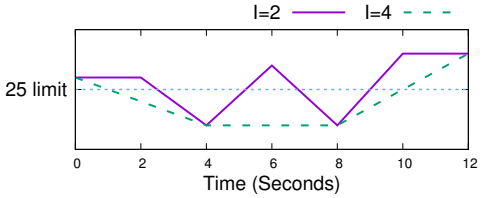



Figure 4.1: Toy example of the evolution of ranks for a node, captured with two different values of $I$.

For each dataset, we investigate the impact of increasing the value of $I$ on the global ranking of nodes. We compute the Temporal Closeness with different multiples of $I_{op}$. For each multiple, we compute the global ranking in relationship to the obtained values of $Dur_{top}$, and compare this ranking to the global ranking produced using $I_{op}$. We compare both rankings using the Kendall tau correlation. Additionally, we study the decrease in the computational time.

For each dataset, for multiples of $I_{op}$ (2, 10, 100, 1000), Table 4.1 presents the original computation time, the Kendall tau correlation and the computation time[1]. First, we focus

1. Normalized by the computation time when $I = I_{op}$.

| **Datasets** | $I_{op}$ | $I_{op} \times 2$ | | $I_{op} \times 10$ | | $I_{op} \times 100$ | | $I_{op} \times 1000$ | |
|---|---|---|---|---|---|---|---|---|---|
| | time (Seconds) | $\tau$ | time/time $I_{op}$ | $\tau$ | time/time $I_{op}$ | $\tau$ | time/time $I_{op}$ | $\tau$ | time/time $I_{op}$ |
| Enron | 161 | 0.99 | 0.54 | 0.99 | 0.19 | 0.99 | 0.11 | 0.92 | 0.099 |
| Radoslaw | 909 | 1.0 | 0.53 | 0.99 | 0.14 | 0.85 | 0.05 | 0.85 | 0.05 |
| DNC | 199580 | 0.99 | 0.49 | **0.99** | **0.098** | 0.99 | 0.002 | 0.977 | 0.002 |
| UC | 45526 | 0.99 | 0.468 | 0.99 | 0.11 | 0.99 | 0.03 | 0.99 | 0.02 |
| HashTags | 30300 | 0.99 | 0.57 | 0.99 | 0.13 | 0.99 | 0.04 | 0.97 | 0.03 |
| Articles | 5649 | 0.99 | 0.85 | 0.99 | 0.70 | 0.95 | 0.64 | 0.72 | 0.62 |
| Facebook | 438950 | 0.99 | 0.51 | 0.99 | 0.12 | 0.99 | 0.02 | 0.97 | 0.003 |
| Bitcoin | 878 | 0.99 | 0.62 | 0.98 | 0.30 | 0.80 | 0.23 | 0.60 | 0.20 |
| RollerNet | 46 | 0.97 | 0.95 | 0.90 | 0.89 | 0.73 | 0.89 | 0.48 | 0.89 |
| Reality | 186 | 0.94 | 0.85 | 0.94 | 0.81 | 0.92 | 0.79 | 0.74 | 0.76 |
| Taxi | 17 | 0.99 | 1.0 | 0.98 | 0.94 | 0.80 | 0.94 | **−0.18** | **0.94** |
| Primary-day1 | 26 | 0.98 | 0.88 | 0.92 | 0.84 | 0.63 | 0.80 | 0.01 | 0.80 |

Table 4.1: For the 12 datasets, the computation time of the temporal closeness with $I_{op}$, and Kendall tau correlation between global ranking $I$ and $I_{op}$. The computation time is normalized by the true computation time for different multiple of $I_{op}$.

on the evolution of the correlation as $I$ increases. We observe in most cases that the correlation remains quite high until the multiple 100. Afterwards, at 1000, we note that the correlation starts to drop drastically in certain cases. In Taxi, the correlation drops from 0.80 to $-0.18$. This is due to the fact that the value of $I$ is close to the duration of the datasets. This proximity decreases the number of computing instants drastically, hence the computed $Dur_{top}$ becomes meaningless and thus, the correlation is low. In other cases, like Enron, this decrease is absent, since the multiple of 1000 remains quite small compared to the duration of the dataset. Additionally, manual investigation showed that the correlation starts to decrease once the value of $I$ becomes larger than or equal to certain nodes' $Dur_{top}$value. Notice that for some datasets. such as Bitcoins, Taxi, or Primary-day1 the correlation begins to lower significantly at $I = I_{op} \times 1000$.

Now we consider the computation time gained as the value of $I$ increases. In most cases, the evolution is similar. The computation time decreases linearly with the value of $I$. When the value of $I$ is doubled, the time is halved and

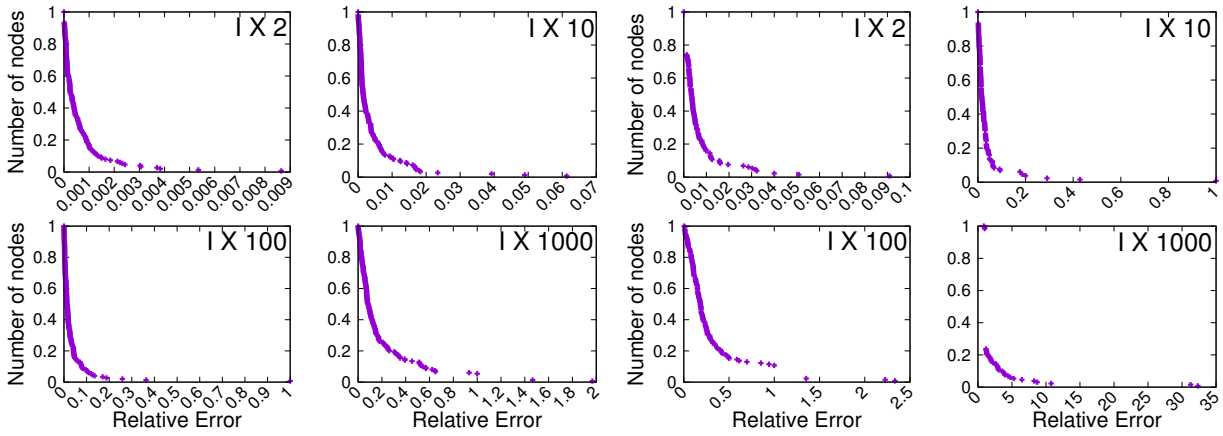



Figure 4.2: Inverse distribution of relative error for different multiplies of $I_{op}$. Left four figures: Enron; right four figures: Taxi.

when $I$ is multiplied by 10, the time is 10% of the original time. Finally, we can observe that afterwards, the time gain becomes quite small. The computation times for $I$ equal to 100 and 1000 are almost the same. This is due to the fact that the computation time that remains is the time required to analyze the raw dataset[2] and thus, the effect of $I$ on the computational time is negligible. In other cases, mostly in the movement datasets, we can observe that changing the value of $I$ has a limited effect on the computational time. These datasets are quite dense and the effect of $I$ is already negligible.

2. The dataset without the extra timestamps added by every $I$ instant.

After studying the global ordering, for each node, we observe how the $Dur_{top}$ value changes as the value of $I$ increase. For each value of $I$, for each node, we calculate the relative error between its $Dur_{top}$ value and the value computed using $I_{op}$[3]. Figure 4.2 presents the inverse cumulative distribution of the relative error for two datasets: Enron (left four figures), which has a stable Kendall tau correlation over all multipliers of $I$, and Taxi (right four figures), for which the Kendall tau correlation decreases drastically[4]. In Enron, we can observe that as $I$ increases, the distribution keeps the same shape, though the values change. As we observed previously the Kendall tau remains high, which means that the ordering of nodes remains stable. This can explain that the similarity here is a result of all the nodes being affected by $I$ in the same man-

3. Only one node had a $Dur_{top}$ equal to zero, which we removed from our calculation of relative error.

4. The plots for the other datasets are presented in the appendix.

ner. We observe two differences as the multiple of $I$ increases from 2 to 1000. First, the maximum error increases from 0.009 to 2; second the average error is multiplied by 100. We manually study the nodes with a large error. We observe that those nodes had a small $Dur_{top}$ value and once the value of $I$ is larger that nodes' $Dur_{top}$, the newly computed $Dur_{top}$ increases a lot, thus the large error.

In the Taxi dataset, we observe larger errors compared to those observed in Enron. This is in coherence with the evolution of the Kendall tau correlation observed in Table 4.1. We observe that as $I$ increases more nodes have larger relative errors. Finally in the extreme case where is $I$ is multiplied by 1000, several nodes have a huge relative error, while others have a considerably smaller error compared to the rest. And, finally around 70% of nodes have an error equal to 1. Manual investigation revealed that nodes had a $Dur_{top}$ value equal to 0 or the value of $I_{op} \times 1000$. This is not shocking as the value of $I$ is actually larger than half of duration of the dataset, therefore, there is only one computing instant in this case. We also observe that the maximum error reaches 32 when $I$ is multiplied by 1000. Afterwards, we study the average error for these 2 datasets in Table 4.2. When $I$ is multiplied by 2 the average error in Enron (resp. Taxi) is 0.0006 (resp. 0.006). This average finally increases in Enron (resp. Taxi) to 0.19 (resp. 2.0) when $I$ is multiplied by 1000. We can see clearly a factor of 10 between both datasets.

| $I$ \ Datasets | Enron | Taxi |
|---|---|---|
| $I \times 2$ | 0.0006 | 0.006 |
| $I \times 10$ | 0.004 | 0.03 |
| $I \times 100$ | 0.03 | 0.3 |
| $I \times 1000$ | 0.19 | 2.0 |

Table 4.2: Average error for the different multiples of $I$ in Enron and Taxi.

Finally, we study each node of these two datasets more in details. For each node, we observe the relative error as the value of $I$ increases. Figure 4.3 presents for both datasets, for different multiples of $I$, the relative error versus the rank attributed for each node using $I_{op}$. In Enron, we observe that as the value of $I$ increases, the nodes of low rank are the ones most affected. This further confirms our previous assertion, that the nodes are affected by the increase when the value of $I$ is near the nodes' $Dur_{top}$ values. In the Taxi dataset, we observe that all nodes are rapidly affected by the increase in $I$; however, as seen with

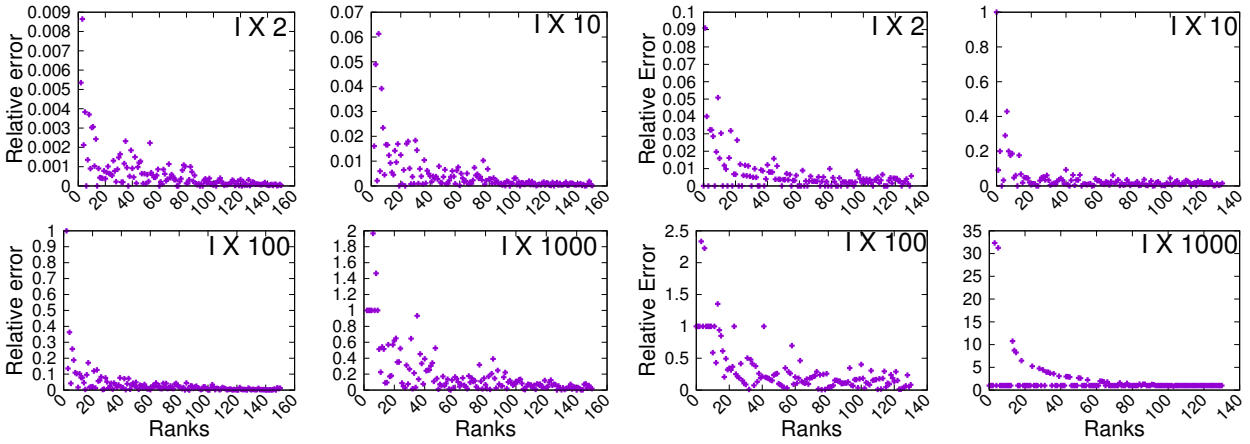



Figure 4.3: Relative error for $Dur_{top}$ as a function of the rank. Left four figures: Enron; right four figures: Taxi.

Enron, the least important nodes are affected more by this increase. Finally, when $I$ is multiplied by 1000, we observe that most nodes have a relative error equal to one. This is due to the fact that the value of $I$ becomes extremely large. All these nodes are given a $Dur_{top}$ value equal to zero, thus, the relative error is equal to one.

### 4.1.1 Conclusion

We conclude that increasing slightly the value of $I$ can be beneficial. Doubling the value of $I$ decreases the computational time by half, while practically not modifying the results. However, as the value of $I$ becomes closer to the total duration of the dataset it becomes disadvantageous. The computation time ceases to decrease, yet the quality decreases dramatically. On the node level, we observed that for any node, the instant the value of $I$ is larger than the node's $Dur_{top}$ (computed using $I_{op}$), the newly computed $Dur_{top}$ value is meaningless. Additionally, we demonstrate that in most cases as $I$ increases, only the nodes of low importance are affected. Thus, for detecting the top 25%, a value of $I$ larger than the optimal can be used.

## 4.2 Identifying important nodes

As observed in the previous section, increasing the value of $I$ can be quite beneficial. However, the computational time

can remain substantial. Additionally, this method has a limited effect, since after a certain point no additional beneficial effect occurs. Here, we propose a second method to reduce the computational time further. We suggest methods similar to those introduced in [Eppstein and Wang, 2001, Okamoto et al., 2008]. Instead of computing the Temporal Closeness, we exploit structural properties of the nodes to rank the nodes. In the obtained ranking, the prime nodes are expected to have a globally important Temporal Closeness. Thus, using this ranking, we can identify the top nodes. This method does not require the computation of Temporal Closeness, and thus reduces extremely the computation time.

### 4.2.1 Strategies

First, we propose to rely on global properties for each node that are easy to compute. We argue that they represent the base of Temporal Closeness. A temporal graph $G = (V, E)$ where $V$ is a set of nodes and $E$ is a set of timed links, is transformed into an aggregated graph $G_A = (V, E_A)$ where $E_A = \{(u, v) \in V \times V | \exists t, (u, v, t) \in E\}$. In other words, we consider only the nodes and links between each node without taking into account when each link has occurred.

From the aggregated graph we compute for each node $u$ several properties:

- **Closeness Centrality** (CC): The sum of the inverse of distance between $u$ and all other nodes in the graph [Bavelas, 1950].
- **Degree Centrality** (DC): The number of neighbors of $u$ in the aggregated graph.

In addition to these structural properties, we consider three properties that take into consideration the temporal aspect of the node's activity:

- **Number of links** (NL): The total number of interactions involving $u$.

- **Duration** (DU): The duration of activity for the node $u$, that is the time elapsed between the last and first interactions that involve $u$.
- **Average inter-contact time** (LD): The average of the difference in time between each link in which $u$ is involved.

We propose two strategies that exploit these five strategies to give each node a rank.

**Parameter based strategy**: The first class of strategies combines any of these two properties. For a node $u$ and a property $P_x$, the function $\texttt{rank}(P_x(u))$ returns the ranking of $u$ with respect to the property $P_x$. From this we attribute the node $u$ a final rank that considers the two properties $P_1$ and $P_2$ in the following manner:

$$R(u) = \alpha \times \texttt{rank}(P_1(u)) + (1 - \alpha) \times \texttt{rank}(P_2(u)). \quad (4.1)$$

The value of $\alpha$ is in $[0, 1]$. Note that $\alpha = 1$ (resp. $\alpha = 0$) implies that only the ranking of $P_1$ (resp. $P_2$) is taken in consideration. From this strategy, we can produce all possible combinations of properties as well as consider a single property. Table 4.3 presents the combination of properties that we consider, the denomination of each combination as well as which strategy is taken into account when $\alpha$ is equal to either one or zero. We should mention that we also considered the case of combining the raw values and then ranking nodes rather than combining the ranks; however, combining the ranks seemed to produce better results.

**Parameterless strategy**: As the Parameter based strategies contains an $\alpha$ parameter for which we do not know the best value, we propose a second strategy that does not contain a parameter. This allows the strategy to be used directly; however, this has a disadvantage as the strategy cannot be refined for each dataset. This strategy considers only two properties: the number of links and the duration. In addition, it considers the raw values rather than the ranks as seen with the parameter based strategy. Formally for a node $u$, this strategy is defined as follows:

$$R(u) = \texttt{rank}(NL(u) \times DU(u)). \qquad (4.2)$$

| Denomination | Strategies | $\alpha = 1$ | $\alpha = 0$ |
|---|---|---|---|
| $CC/DC$ | Closeness<br>Degree | Closeness | Degree |
| $CC/NL$ | Closeness<br>Number of links | Closeness | Number of links |
| $CC/DU$ | Closeness<br>Duration | Closeness | Duration |
| $CC/LD$ | Closeness<br>Average inter-contact time | Closeness | Average inter-contact time |
| $DC/NL$ | Degree<br>Number of links | Degree | Number of links |
| $DC/DU$ | Degree<br>Duration | Degree | Duration |
| $DC/LD$ | Degree<br>Average inter-contact time | Degree | Average inter-contact time |
| $NL/DU$ | Number of links<br>Duration | Number of links | Duration |
| $NL/LD$ | Number of links<br>Average inter-contact time | Number of links | Average inter-contact time |
| $DU/LD$ | Duration<br>Average inter-contact time | Duration | Average inter-contact time |

Table 4.3: Denomination of all the combination of strategies that are considered, and the considered strategy when $\alpha = 1$ and $\alpha = 0$.

### 4.2.2 Best combination

In order to find the best combination of properties for the Parameter based strategy as well as the best value for $\alpha$ a comparison method is required. We start by defining the *hit vector*. For any strategy, this vector represents the evolution of the strategy's success. In this vector, the *kth* element represents the number of nodes correctly found by the strategy when trying to detect the top $k$ nodes[5]. The hit vector of a perfect strategy is equal to $[1, 2, 3, \ldots, n]$, where $n$ is the total number of nodes in the dataset. Figure 4.4 shows an example of the hit vector for three strategies: the perfect strategy, the worst strategy and the random strategy.

Intuitively, the closer a strategy is to the perfect strategy,

5. For example, if the value is equal to 5 for $k = 10$, this means that half of the nodes in the strategy's top 10 belong to the real top 10.

the better the ranking of the strategy is. To find to which extent a strategy is efficient, we measure the euclidean distance between its hit vector and the perfect strategy's hit vector[6]. Finally, to be able to compare the performance of the strategy over different datasets, we need to normalize this value. To accomplish this, we consider the distance separating the perfect and worst strategy as the normalizing factor. From here, we can attribute to each strategy a score signifying its quality. This score is formally defined for a strategy $S$ as follows:

$$\texttt{score}(S) = 1 - \frac{distance(perfect\_strategy, S)}{distance(perfect\_strategy, worst\_strategy)}. \tag{4.3}$$

To study the performance of the strategies in another way, we introduce also the `hitratio`. We define $\texttt{real}(D,k)$ as the set of top $k$ nodes of the dataset $D$. Moreover, we represent the set of top $k$ nodes identified by a strategy $S$ by $\texttt{sampled}(S,D,k)$. From these two definitions, we calculate the fraction of correctly detected nodes by $S$ in $D$, in the following manner:

$$\texttt{hitratio}(S,D,k) = \frac{|\texttt{sampled}(S,D,k) \bigcap \texttt{real}(D,k)|}{k}. \tag{4.4}$$

This ratio is between 0 and 1, where 0 implies the strategy was not able to identify a single node accurately. On the contrary, 1 signifies the strategy is perfect for the corresponding value of $k$. This is somewhat redundant with the hit vector, but we argue that it is easier to study the hit ratio as the value it returns for any $k$ represents directly the efficiency of the strategy.

6. The distance between two vectors $p$,$q$ is $distance(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$.

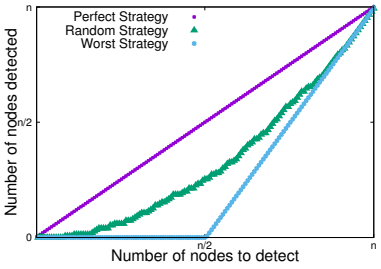



Figure 4.4: Evolution of hit vector for Random Strategy, Best Strategy and Worst Strategy.

### 4.2.3 Results

Here we consider all the combinations of strategies on the twelve datasets. Firstly, we will determine the two combinations that produce the best results as well as the $\alpha$ that

gives these results. Secondly, we will examine these two strategies as well as the parameterless strategy more in details.

Figure 4.5 shows for all the combinations, the `score` value as a function of $\alpha$. We first consider the communication datasets. In the Enron dataset, we observe that $NL/DU$ and $DU/LD$ give the highest score. Both produce a score of 0.84, while the third highest score is 0.74. Furthermore, $NL/DU$ is also the top strategy for Radoslaw, while the second highest score is produced by $CC/NL$. Nevertheless, the scores for all the strategies remain globally quite close. For DNC, the same observations can be made, however with scores that are lower than that of Radoslaw. Finally, in the case of UC, the strategies are all quite similar.

In the case of the two co-occurrence and the two social datasets, the strategies have all the same behavior. For example, the $DU/LD$ score increases as $\alpha$ increases in the four datasets. In the case of HashTags and Facebook, $CC/DU$ is the most efficient. However, the difference between the strategies' peak scores remains negligible. In the case of Articles and Bitcoins, $DU/LD$ produces the best results. However, the best scores also remain quite close.

Finally, we consider the motion datasets. The strategies behave differently on each dataset. Moreover, we can observe behaviors that are unique to these datasets. First, for certain strategies, the difference between the maximum and minimum score can be remarkably large. Second, other strategies produce a consistent score for any $\alpha$. Finally, the top strategy is different in each of the four datasets. From these results, we observe that while not always the best, $DU/LD$ and $NL/DU$ produce frequently the best outcome. From now on, we eliminate the other strategies and consider only $DU/LD$, $NL/DU$, and $PS$. We study the values of $\alpha$. Table 4.4 gives the values of $\alpha$ that give the best score for $DU/LD$ and $NL/DU$ for each dataset. We can observe that the nature of the dataset influences the choice of $\alpha$. For instance, in RollerNet and Primary-day1 all the nodes are active for the same amount of time, hence

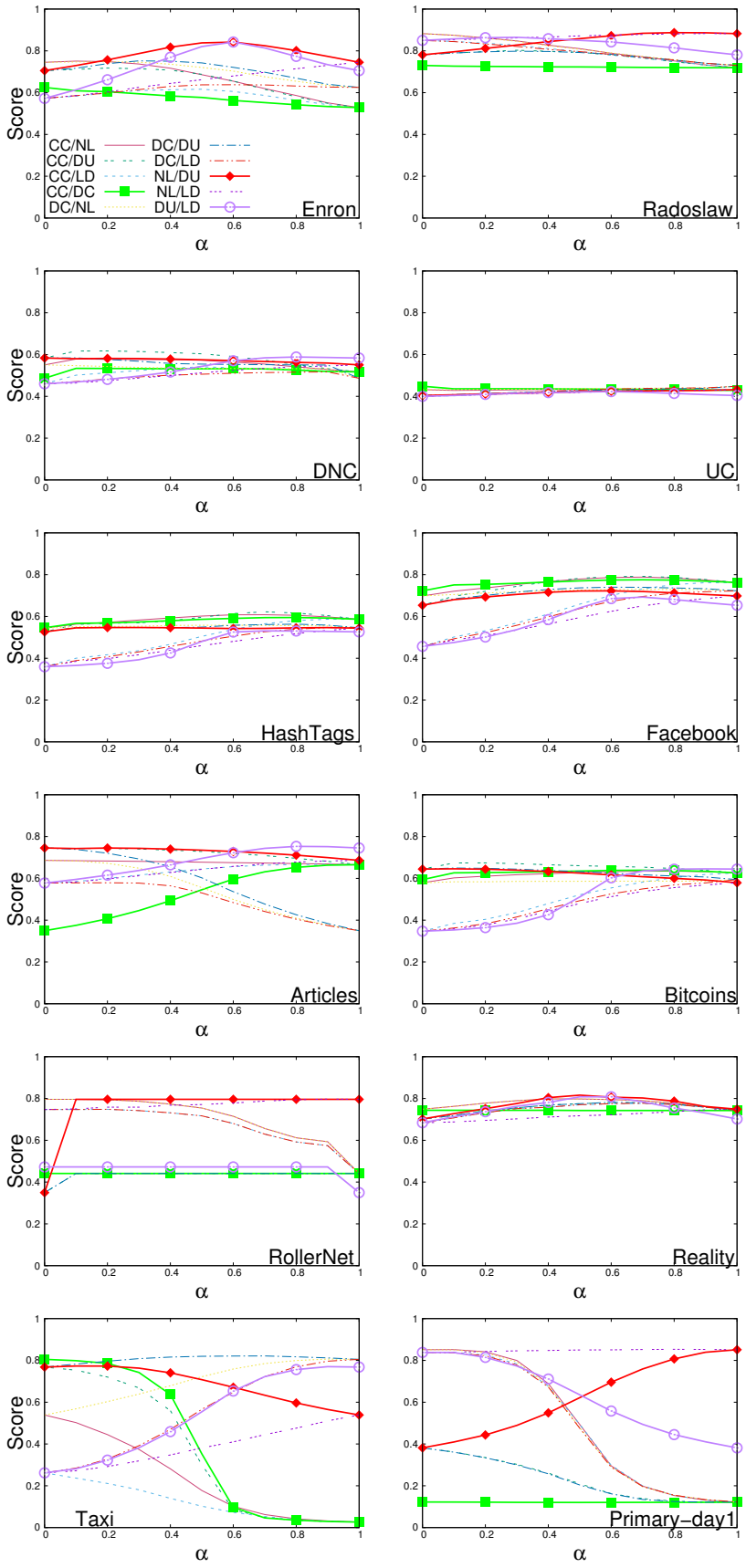



Figure 4.5: Score as a function of $\alpha$ for each strategy on the twelve datasets.

| Datasets | $DU/LD$ | $NL/DU$ | Datasets | $DU/LD$ | $NL/DU$ |
|---|---|---|---|---|---|
| **Enron** | 0.6 | 0.6 | **Radoslaw** | 0.3 | 0.8 |
| **DNC** | 1.0 | 0 | **UC** | 0.6 | 1.0 |
| **HashTags** | 0.7 | 1.0 | **Facebook** | 0.7 | 1.0 |
| **Articles** | 0.6 | 0.8 | **Bitcoin** | 0.9 | 0.1 |
| **RollerNet** | 0.0 | $0.1 - 0.9$ | **Reality** | 0.5 | 0.5 |
| **Taxi** | 0.9 | 0.1 | **Primary-day1** | 0.0 | 1.0 |

Table 4.4: Best $\alpha$ for $DU/LD$ and $NL/DU$ for all the datasets.

$\alpha = 0$ in $DU/LD$. We therefore argue that the nature of the dataset should be taken into account when choosing the value of $\alpha$.

| Datasets | $DU/LD$ | $NL/DU$ | $PS$ | Datasets | $DU/LD$ | $NL/DU$ | $PS$ |
|---|---|---|---|---|---|---|---|
| **Enron** | 0.84 | 0.84 | 0.81 | **Radoslaw** | 0.86 | 0.88 | 0.88 |
| **DNC** | 0.58 | 0.58 | 0.58 | **UC** | 0.42 | 0.43 | 0.42 |
| **HashTags** | 0.53 | 0.54 | 0.52 | **Facebook** | 0.69 | 0.72 | 0.70 |
| **Articles** | 0.75 | 0.74 | 0.74 | **Bitcoin** | 0.64 | 0.64 | 0.63 |
| **RollerNet** | 0.42 | 0.79 | 0.81 | **Reality** | 0.80 | 0.81 | 0.79 |
| **Taxi** | 0.77 | 0.77 | 0.67 | **Primary-day1** | 0.83 | 0.85 | 0.84 |

Table 4.5: `score` values produced by $DU/LD$, $NL/DU$ and $PS$.

Before studying these two strategies as well as $PS$ more in details, we compare the scores produced by $DU/LD$ and $NL/DU$ with the best values of $\alpha$ and those of $PS$ in table 4.5. We can observe that in most cases, $PS$ scores are close to the best score produced by the other methods.

Now we consider how accurate each strategy is. For each strategy, we examine the $\texttt{hitratio}(S, D, k)$ for all values of $k \in [0, n]$, where $n$ is the number of nodes of the dataset in question. Figure 4.6 presents the values of the `hitratio` as a function of $k$ for the three strategies on three datasets that are representative of what we observed for the twelve datasets, using the optimal value of $\alpha$. In Enron and most the datasets, we observe that the three strategies behave similarly. The only exceptions are RollerNet and Taxi. In the case of RollerNet, $DU/LD$ has the weakest performance; this is due to the fact that all the nodes in

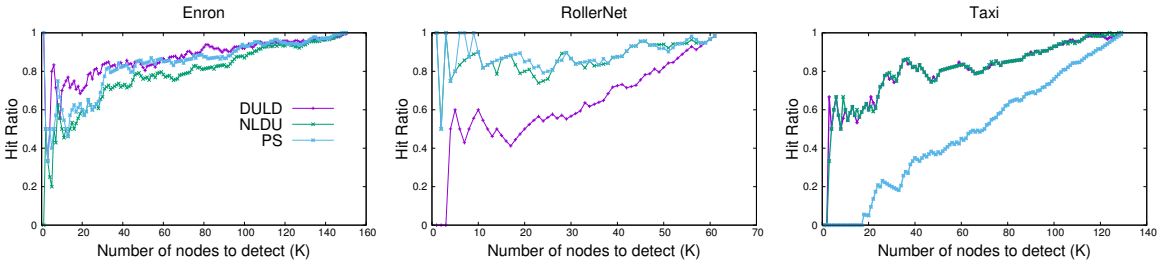



Figure 4.6: Evolution of the `hitratio` as a function of $k$ for 3 datasets: Enron; RollerNet; Taxi.

this dataset set communicate constantly with each other, thus the property $LD$ can be meaningless.

In Taxi, $PS$ has a weaker performance than the other strategies. Manual investigations showed that a lot of nodes have a similar number of links and durations, but they appear for first time at different instants. $PS$ does not take this in account, but Temporal Closeness does. Nodes that become active later are more important than nodes active for the same duration but earlier, as they participate in temporal paths for a longer time.

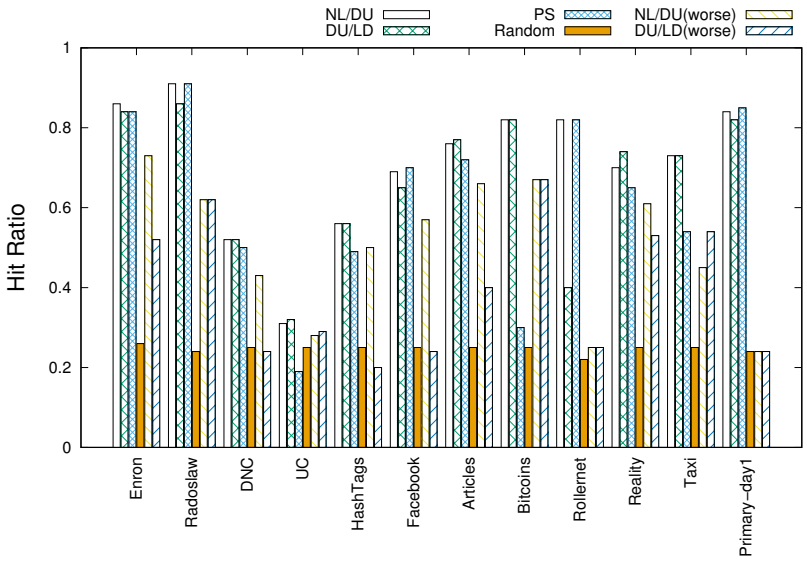



Figure 4.7: Results of `hitratio`$(25,\_,\_)$ for both $DU/LD$ $NL/LD$ with best and worst $\alpha$, $PS$ and random strategy.

We now examine how each strategy performs when detecting the top 25% nodes. We compare $NL/DU$, $DU/LD$ (best and worst $\alpha$) and $PS$ against a random sampling strategy[7]. Figure 4.7 shows the value of the `hitratio` for

7. The average hit score of 100 trials of randomly selecting 25% nodes.

$k = 25\%$ of the nodes on the 12 datasets for the five strategies in question. Firstly, as expected the random strategy produces a `hitratio` approximately equal to 0.25 for the 12 datasets. Secondly, both $NL/DU$ and $DU/LD$ produce similar results in the optimal $\alpha$. Thirdly, remarkably in several cases, $PS$ performs as well as $NL/DU$ and $DU/LD$. Lastly, the `hitratio` can vary notably between the optimal and the worst $\alpha$ in certain datasets. In the case of the worse $\alpha$, except UC, $NL/DU$ always performs better than $DU/LD$. In other words, $DU/LD$ is affected more by the changes in the value of $\alpha$. Again these observations remain coherent with the previous results.

### 4.2.4 Ordering

In certain cases detecting the top 25% nodes might be insufficient; additionally, one might require the correct ordering between those nodes. Thus, for each strategy, we investigate how its ranking correlates with the actual ranking. To obtain the actual ranking, we compute the Temporal Closeness and $Dur_{top}$ values of the 25% nodes ranked highest by the considered strategy, and then rank the nodes in relationship to their $Dur_{top}$ values. Table 4.6 presents the Kendall tau correlation between the actual ranking and the ranking produced by each strategy using the optimal $\alpha$ of each dataset. We can observe that the correlation is low; the highest value is 0.69. Additionally, we can observe that the performance is far from constant and varies extremely across datasets and strategies. For example, the strategy $PS$ has correlations ranging from $-0.33$ to 0.69 across the datasets, which are also the highest and lowest values overall. From this, we conclude that while these strategies can detect the important nodes, they are not able to detect them in the same order as that of $Dur_{top}$. This low correlation is quite expected, since a node mis-identified by a strategy is normally less important than all the other true important nodes. Thus, if a strategy miss samples the second node, it naturally causes discordant pairs for a high number of nodes.

| **Datasets** | $DULD$ | $NLDU$ | $PS$ |
|---|---|---|---|
| **Enron** | 0.47 | $-0.25$ | 0.45 |
| **Radoslaw** | 0.11 | $-0.05$ | 0.48 |
| **DNC** | $-0.04$ | 0.53 | 0.58 |
| **UC** | 0.08 | $-0.02$ | 0.06 |
| **HashTags** | 0.005 | 0.02 | 0.02 |
| **Articles** | 0.01 | 0.05 | **0.69** |
| **Facebook** | $-0.08$ | 0.25 | 0.25 |
| **Bitcoin** | 0.03 | 0.05 | 0.03 |
| **RollerNet** | 0.06 | 0.46 | **−0.33** |
| **Reality** | 0.14 | 0.1 | 0.68 |
| **Taxi** | $-0.01$ | $-0.08$ | 0.19 |
| **Primary-day1** | 0.01 | 0.04 | 0.35 |

Table 4.6: Kendall tau between each strategy's ranking and the accurate ranking for the 25% sampled nodes.

### 4.2.5 Conclusion

We studied three strategies that use structural properties to detect the central nodes. Two of these strategies require a parameter and in most cases, the optimal value of this parameter depends on the nature of the dataset. Furthermore, the third strategy does not require a parameter, yet produces results quite similar to the first two strategies when detecting the top 25% nodes. $NL/DU$ found on average 70% (resp. 50%) of the top nodes with optimal (resp. worst) $\alpha$. $DU/LD$ found on average 60% (resp. 30%) of the top nodes with optimal (resp. worst) $\alpha$. $PS$ found on average 50% of the nodes. Finally, we observe that these strategies are capable to find efficiently the important nodes. However, those methods cannot provide the ranking of these nodes. We also note that these methods are computed instantly, compared to Temporal Closeness that can take several hours for certain datasets.

## 4.3 Approximation

In addition to identifying the important nodes, one might want to know the duration for which each node is important. In this section, we propose a method that gives an approximation of this duration, without the need to compute it. In other words, we shall try to approximate the $Dur_{top}$ value for each node without computing the Temporal Closeness. We exploit statistical characteristics related to the distribution of these values. Here we consider only the datasets that have a clear notion of importance: Enron, Radoslaw, DNC, HashTags, Facebook and Articles[8].

8. See Section 3.4.2.

[Saxena et al., 2017] showed that the ranking versus closeness centrality always follows a specific pattern in static graphs. We investigate this here for the temporal case. Figure 4.8 presents for all the nodes the ranks versus the $Dur_{top}$ values normalized by the dataset's total duration, for the six datasets we consider. We observe two interesting characteristics. Firstly, in five datasets, the largest $Dur_{top}$ is roughly equal to the total duration of the dataset:

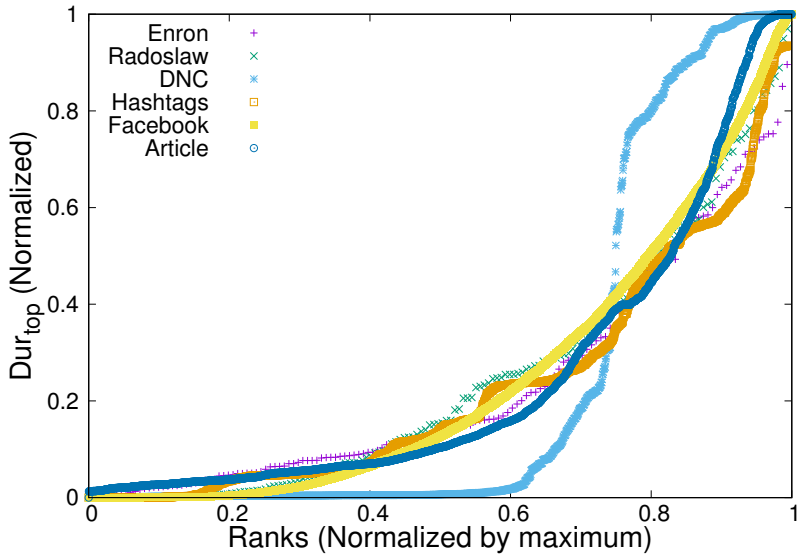



Figure 4.8: For all the nodes the $Dur_{top}$ values versus the ranks for Enron, Radoslaw, DNC, HashTags, Facebook and Articles.

it is over 97% of the total duration; it is equal to 85% for the sixth dataset. Secondly, for all the datasets (except DNC), the distribution of points can be represented by exponential function. The distribution for DNC can be represented by a sigmoid function. We can exploit these observations to approximate the values of $Dur_{top}$. We denote from now on the accurate $Dur_{top}$ value as $Dur_{top}Accurate$.

### 4.3.1 Exponential function

We start by studying the datasets that present an exponential function[9]. For each dataset, the number of nodes, the minimum $Dur_{top}Accurate$ value (equal to 0) and the maximum $Dur_{top}Accurate$ value are known. To generate approximations of the $Dur_{top}$ value for each node, we consider an exponential function that generates $n$ values between 0 and $M$, where $n$ is number of nodes and $M$ is maximum $Dur_{top}Accurate$ value. We propose the following function:

$$f(x) = M \times \left[ \left( \frac{1}{1 - e^{-\beta}} \right) \left( e^{\beta \left( \frac{x}{n} - 1 \right)} - 1 \right) + 1 \right], \qquad (4.5)$$

where $\beta$ is a constant that changes the curvature of the

9. Enron, Radoslaw, HashTags, Facebook and Articles.

plot; we discuss this value later on. When $x = 0$, this function returns 0 thus, the node with lowest ranking will have a $Dur_{top}$ equal to 0. On the other side, when $x = n$, $f(x) = M$, thus the node with the highest rank will have a $Dur_{top}$ equal to $M$.

To evaluate the quality of this method, we will first evaluate the performance in the optimal situation: $M$ is equal to the highest $Dur_{top}Accurate$; we have identified correctly the top 25% nodes and their ranking; $\beta$ is equal to the value that gives the best results[10]. With these values, we generate $n$ values, from which we keep the top 25%, and attribute these values to the top 25% nodes following the correct ranking. To assess the performance of this method, for each of the top 25% nodes, we study the relative error between the correct $Dur_{top}Accurate$ and the estimated value.

10. We manually computed the optimal value for $\beta$.

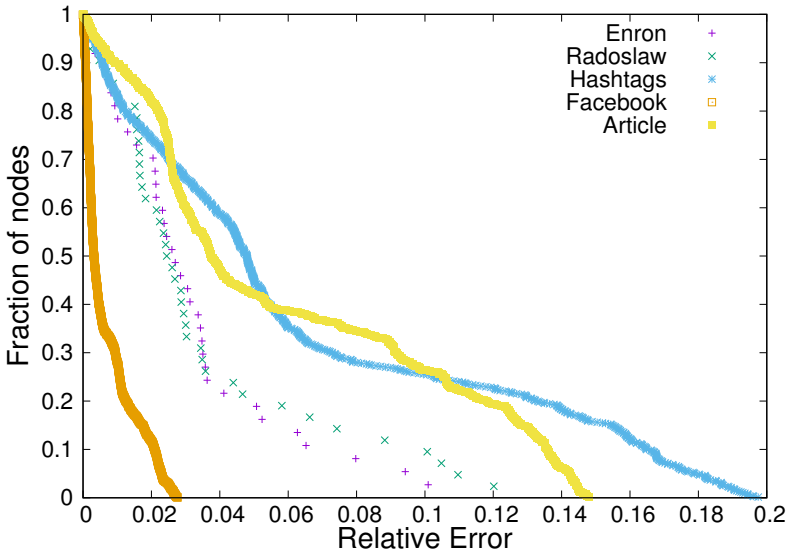



Figure 4.9: Inverse cumulative distribution of the relative error for the $Dur_{top}$ estimation in the best case.

Figure 4.9 presents the inverse cumulative distribution of the relative error for the five datasets. We can see that the worst error is just under 0.2, in HashTags. Both email exchange datasets (Enron, Radoslaw) have a quite similar distribution, with a maximum error quite close (0.1 for Enron and 0.12 for Radoslaw). In the case of Facebook, the

maximum error is quite low (around 0.02); finally for Articles, we can see a similar distribution to that of HashTags, with a lower maximum error. Finally, the average relative error in these five datasets is between 0.006 and 0.06. One can assume that a tailored function for each dataset would give better results; however this is impossible as a tailored function would require pre-knowledge, such as the exact curve, that we do not necessarily have.

In reality the correct value of $M$ is unknown. Nevertheless, we observed in figure 4.8 that the maximum value ($Dur_{top}Accurate$ value) is close to the duration of the datasets. From that, we propose to set $M$ to the duration of the dataset. With this value, the average relative error increases to a minimum of 0.02 with the maximum remaining at 0.06. Thus, we conclude that a non-optimal value of $M$ has a low effect on the estimation and that the duration of the dataset can be used. Table 4.7 shows the average relative error for both $M$ values.

| Dataset \ $M =$ | $Dur_{top}$ $Accurate$ | Duration |
|---|---|---|
| **Enron** | 0.05 | 0.06 |
| **Radoslaw** | 0.03 | 0.02 |
| **HashTags** | 0.06 | 0.06 |
| **Facebook** | 0.006 | 0.02 |
| **Articles** | 0.05 | 0.05 |

Table 4.7: Average relative error using $M$ equal to maximum $Dur_{top}Accurate$ and Duration for the five datasets.

Before proceeding to the practical case, where we lack any pre-knowledge about the dataset, we study the optimal values of $\beta$. The optimal values for the 5 datasets are all between 3.4 and 4.0, even through these datasets do not have the same nature. We study the effect of modifying the value of $\beta$ on the 5 datasets. For each dataset, for all $\beta$ values between 3.4 and 4.0, we compute the average relative error value of the top 25% node. Figure 4.10 presents for each dataset the average error as a function of $\beta$.

We can observe that in all datasets except Facebook, the evolution of the relative error is similar. The lowest relative error for the four datasets is between 3.8 and 4.0; additionally, the relative error does not vary much between these two values. The Facebook dataset has a different evolution: as the value of $\beta$ increases, so does the relative error. The relative error increases only when the value of $\beta$ is lower than 3.4. In most cases the relative error remains low compared to the other four datasets. While this is not definitive and would require more testing, we think values between 3.8 and 4.0 can be used for any dataset. Further testing on

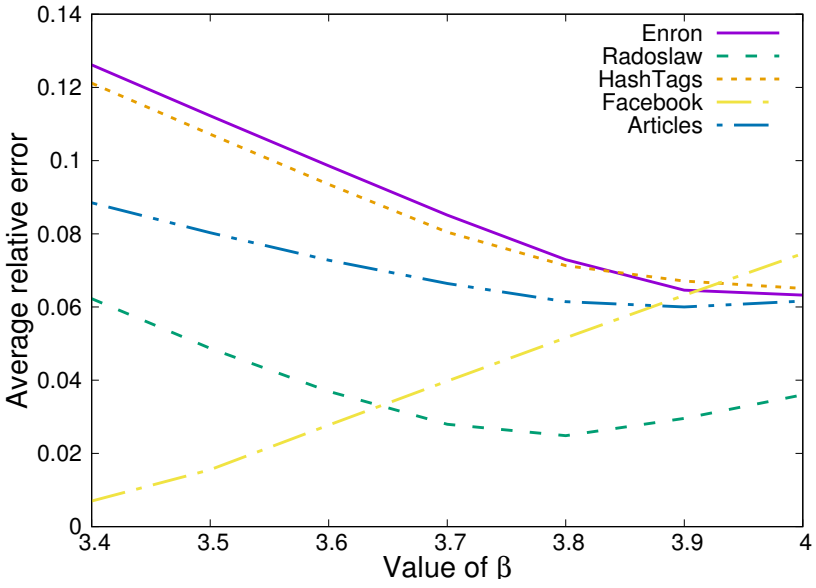



Figure 4.10: Average relative error as a function of $\beta$ for the 5 datasets: Enron; Radoslaw; HashTags; Facebook; Articles.

more datasets would give better indicators for this interval.

### 4.3.2 Sigmoid function

Now, we concentrate on the case of DNC, where the distribution can be represented by a sigmoid function. A sigmoid can be defined using the formula $S(x) = \frac{1}{1+e^{-x}}$. However, this function only reaches 0 (resp. 1) when $x = -\infty$ (resp.$x = \infty$). While in our case, when $x = 0$ (resp. 1) we want the function to return 0 (resp. $M$). To obtain a sigmoid distribution between 0 and $M$, we define the two following functions:

$$g(x) = -ln(\frac{1-x}{x}), \tag{4.6}$$

$$f(x) = \frac{M}{1+e^{-\beta(g(x)-\alpha)}}, \tag{4.7}$$

where $M$ is the maximum value, $\alpha$ is a constant that controls the center of the curvature and $\beta$ controls the degree of curvature.

As we previously did with the exponential distribution, we study the relative error in the optimal case, where $M$ is equal to the maximum $Dur_{top}Accurate$ value, and we

know the optimal $\alpha$ and $\beta$ values. Additionally, we study the case where $M$ is equal to the duration of the dataset.

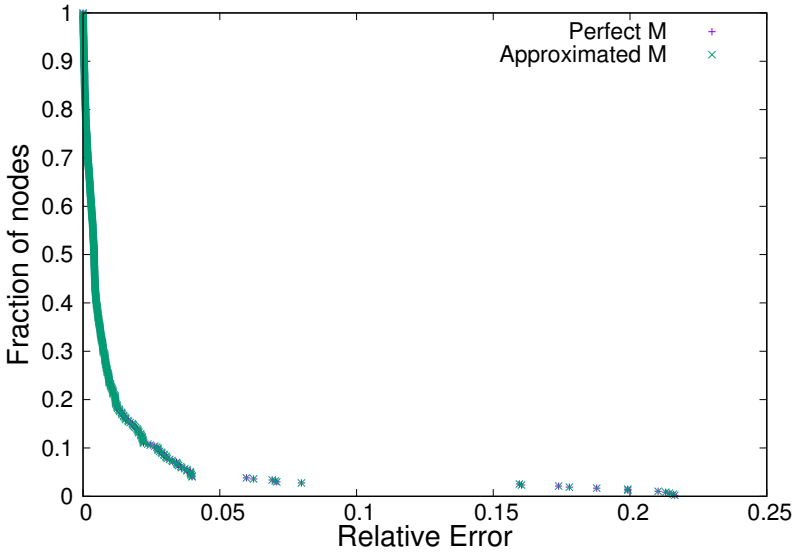



Figure 4.11: Inverse cumulative distribution of the relative error for the DNC dataset.

Figure 4.11 presents the inverse cumulative distribution of the relative error for the $Dur_{top}$ estimation for the top 25% nodes, for $M$ equal to the maximum $Dur_{top}Accurate$ value and to the dataset's duration. First of all, the two distributions are almost the same; this is expected as the difference between both values of $M$ values is negligible. The largest error is close to 0.2, the same as we observed in the case of the exponential. Finally, we note that the average relative error is 0.012.

Now, we study the effect of changing the values of $\alpha$ and $\beta$. Figure 4.12 presents the average relative error between the estimated and $Dur_{top}Accurate$ for the top 25% nodes in function of $\alpha$ and $\beta$. For most combinations (over 90%) of $\alpha$ and $\beta$ the average error is less than 0.1. Thus we conclude that the effect of $\alpha$ and $\beta$ is limited unless one uses extreme values such as $\alpha = 1$ and $\beta = 1$ which produce large average errors. Unfortunately, we didn't encounter the sigmoid distribution on any other dataset, thus we are unable to verify if the changes in the values of $\alpha$ and $\beta$

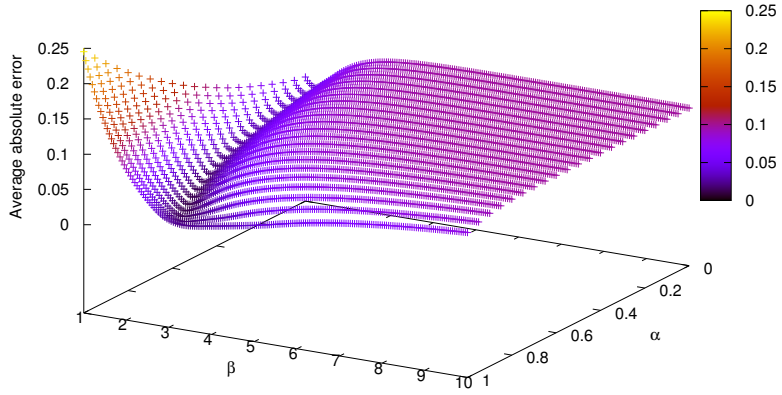



Figure 4.12: Average relative error between $Dur_{top}$ estimation and true $Dur_{top}$ for the top 25% as a function of $\alpha$ and $\beta$ for DNC.

would have the same effect on other datasets.

### 4.3.3 Conclusion

From this section, we observed that in most cases, the datasets have a quite similar distribution of $Dur_{top}Accurate$ values, corresponding to an exponential distribution. For corresponding datasets, we proposed a function that generates estimated values of $Dur_{top}$. Additionally, for the sixth dataset where the nodes' $Dur_{top}Accurate$ values distribution corresponds to a sigmoid distribution, we proposed a function as well. In the case where the exact ranking of the nodes is known, both functions produces an excellent estimation.

## 4.4 Overall protocol

In this section, we consider the case that is the closest to a real context, where we lack any pre-knowledge about the dataset, and our goal is to detect the top 25% nodes and approximate their $Dur_{top}Accurate$ value. To achieve this goal, we combine the methods introduced previously in Sections 4.1, 4.2 and 4.3.

| **Datasets** | $I_{op} \times 2$ | | $I_{op} \times 10$ | | $I_{op} \times 100$ | | $I_{op} \times 1000$ | |
|---|---|---|---|---|---|---|---|---|
| | $\tau$ | time | $\tau$ | time | $\tau$ | time | $\tau$ | time |
| Enron | 1.0 | 0.22 | 0.98 | 0.19 | 0.97 | 0.11 | 0.91 | 0.11 |
| Radoslaw | 1.0 | 0.28 | 1.0 | 0.15 | 0.98 | 0.07 | 0.94 | 0.07 |
| DNC | 0.99 | 0.18 | 0.99 | 0.08 | 0.99 | 0.01 | **0.95** | **0.008** |
| HashTags | 0.99 | 0.12 | 0.99 | 0.06 | 0.99 | 0.02 | 0.97 | 0.02 |
| Facebook | 0.99 | 0.07 | 0.99 | 0.04 | 0.99 | 0.04 | 0.96 | 0.04 |
| Articles | 0.99 | 0.17 | 0.98 | 0.16 | 0.86 | 0.15 | **0.67** | **0.15** |

Table 4.8: Correlation and computation time normalized by that of $I_{op}$ for different $I$ values (Sampled 25%).

### 4.4.1 Methodology

First, we use the identification method of Section 4.2 to pick out the top 25% of nodes. For simplicity we use the parameterless strategy ($PS$), thus we are not required to find the optimal $\alpha$. From this strategy, we obtain a (unknown) percentage of the true important nodes, 50% on average. In Section 4.2.4 we observed that ranking these nodes according to their $PS$ value does not necessarily give the correct order among them. To get the correct order in an efficient way, we apply the method introduced in Section 4.1. First, we start by computing the exact Temporal Closeness for each node. However, instead of using $I_{op}$, we use a larger value of $I$. Second, we rank the nodes according to the obtained values at each time step, in the same manner as in Chapter 3. Finally, we compute the $Dur_{top}$ value value for each node, however, this value is incorrect for two reasons. First, the optimal $I$ was not used. Second only 25% of nodes are taken into account, thus the computed $Dur_{top}$ for a node represents the amount of time it is in the top 25% of those identified nodes. We call this value $Dur_{top}25$. We will try to improve the estimation of the $Dur_{top}$ value by using the method proposed in Section 4.3 using the ranking produced by the $Dur_{top}25$.

### 4.4.2 In practice

We apply this protocol on the six datasets for which we observe a clear notion of global importance. First, we recall that in those six datasets, the identification method was

able to detect correctly around 50% of the top 25%. For the identified nodes, we compute the Temporal Closeness centrality as well as the $Dur_{top}25$, and use this value to obtain an ordering on these nodes.

Before proceeding, we study how accurate this ordering is, as well as the effect of increasing the value of $I$. We compute the Kendall tau correlation between the ranking produced by the $Dur_{top}25$ for each $I$ and the one obtained by the normal procedure[11]. Table 4.8 presents for the six datasets, the Kendall tau correlation and the computation time for different multiples of $I_{op}$. We observe that when $I_{op}$ is multiplied by 2, the correlation is almost perfect, yet the computation time is much shorter than the case where all the nodes are taken into account. As $I_{op}$ is multiplied by 100, the correlation starts to decrease and the computation time ceases to decrease. Thus, we can conclude from this that $I_{op} \times 10$ should be used instead of $I_{op}$ to further decrease the computation time.

11. Computing the Temporal Closeness for all nodes with $I_{op}$, ranking all nodes and computing $Dur_{top}Accurate$.

With $I_{op} \times 10$, we compute the Temporal Closeness, ranks and $Dur_{top}25$ values for the identified nodes. Table 4.9 presents the average and median relative error between $Dur_{top}25$ and $Dur_{top}Accurate$ for our six datasets. This validates our argument that these values are incorrect and the method proposed in Section 4.3 should be used. To better estimate the $Dur_{top}$ values for a dataset, we need to know which function (exponential or sigmoid) can be used to represent the distribution of the $Dur_{top}$ values. Recall that the six datasets did not exhibit the same distribution shape. Hence, for the identified nodes, we study the ranks against the $Dur_{top}25$ values. Figure 4.13 presents the values of $Dur_{top}25$ normalized by the longest $Dur_{top}25$, versus the ranks. We observe the same behaviors as in Figure 4.8. Five datasets[12] have a distribution that corresponds to an exponential function, while DNC presents a sigmoid function. Note that in the figure, it is hard to notice that Articles can be represented by an exponential function. However, the distribution corresponds to an exponential between the range 0.9 and 1. From this, we conclude that the identified

12. Enron, Radoslaw, HashTags, Facebook and Articles.

nodes represent a good indication of the shape of the distribution of the $Dur_{top}$ values. Therefore, we can use this distribution to determine the function that should be used to estimate the $Dur_{top}$ values.

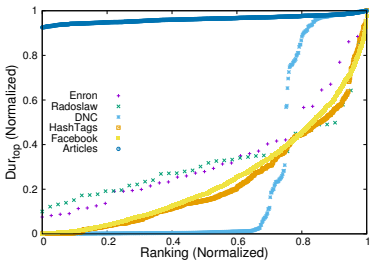



Figure 4.13: $Dur_{top}25$ values versus the ranks for the sampled nodes.

At this point, we have sampled the top 25% nodes using the *PS* strategy and computed the Temporal Closeness with $I = I_{op} \times 10$. From this we have a close approximation of the ordering between those nodes (see Table 4.8). Additionally, we know the correct shape of the distribution of the $Dur_{top}$ values and the dataset's duration. Hence, we have all the required information to estimate the $Dur_{top}$ value following the method described in Section 4.3. We call the obtained value $Dur_{top}Est$.

Now, we observe how close the $Dur_{top}Est$ values are to the true values ($Dur_{top}Accurate$). Figure 4.14 shows the distribution of the relative error for the six datasets we consider. In the case of Enron and Radoslaw, the relative error reaches a value of around 1.0, yet the average remains low: 0.25 and 0.20 in Enron and Radoslaw respectively (see Table 4.9). In these datasets, the nodes that were wrongfully identified, *i.e.* the ones that are not supposed to be in the top 25%, are the ones with a high relative error. The relative error is larger for the other datasets as the sampling had a lower *hitratio*. Therefore, a larger number of misidentified nodes were attributed values belonging to the top 25% when normally they should have lower values. Nevertheless, if we compare these values with the error of $Dur_{top}25$, see Table 4.9, the average error tends to be higher than with $Dur_{top}25$, but the median is much lower.

| **Datasets** | $Dur_{top}25$ | | $Dur_{top}Est$ | |
|---|---|---|---|---|
| | Average | Median | Average | Median |
| **Enron** | 0.5 | 0.6 | 0.25 | 0.1 |
| **Radoslaw** | 0.5 | 0.6 | 0.20 | 0.1 |
| **HashTags** | 0.5 | 0.5 | 8.9 | 0.5 |
| **Facebook** | 0.6 | 0.6 | 1.4 | 0.1 |
| **Articles** | 0.8 | 0.9 | 0.8 | 0.3 |
| **DNC** | 0.6 | 0.8 | 0.4 | 0.46 |

Table 4.9: Average and median of the relative error for the six datasets for both values: $Dur_{top}25$ and $Dur_{top}Est$.

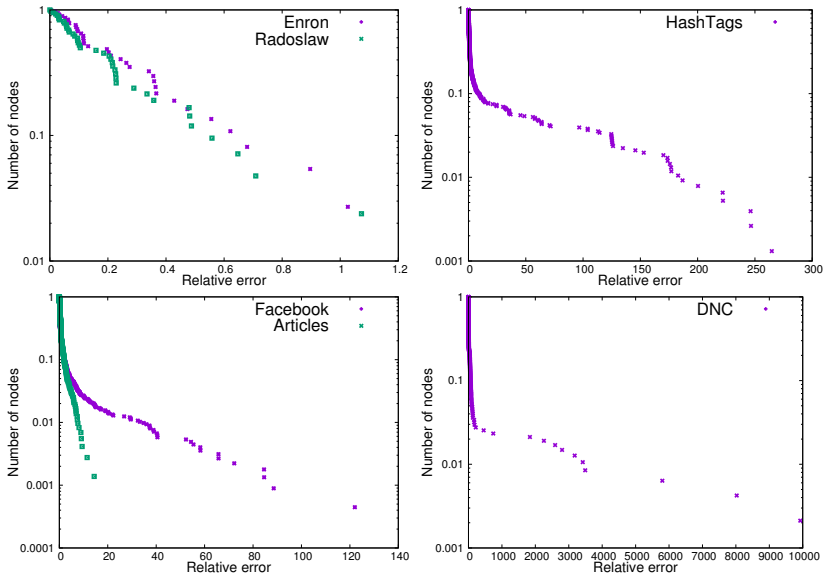



Figure 4.14: Inverse cumulative distribution of the relative error for $Dur_{top}Est$ for the six datasets.

This further shows that the estimation gives good results for the nodes that have been correctly identified by $PS$. Hence, the ability to remove the wrongfully identified nodes would decrease this error.

## 4.5 Discussion

### 4.5.1 Absence of importance

In Section 4.3, to estimate the $Dur_{top}$ value, we considered only datasets that had a clear notion of importance. To identity these datasets, we studied the distribution of the $Dur_{top}$ values in section 3.4.2. Here, we discuss how to detect these datasets without needing to compute the Temporal Closeness and $Dur_{top}$ values for all the nodes.

The first obvious solution is to study the distribution of properties that we used in Section 4.2. Figure 4.15 represents the inverse cumulative distribution of raw $PS$ values, *i.e.* the number of links multiplied by the duration, for the 12 datasets that we considered in Section 3.4.2.

For RollerNet and Primary-day1 we can observe how most nodes are closer to the average compared to the other datasets. This is a sign of the absence of a notion of global importance: the nodes with the highest value are less dif-

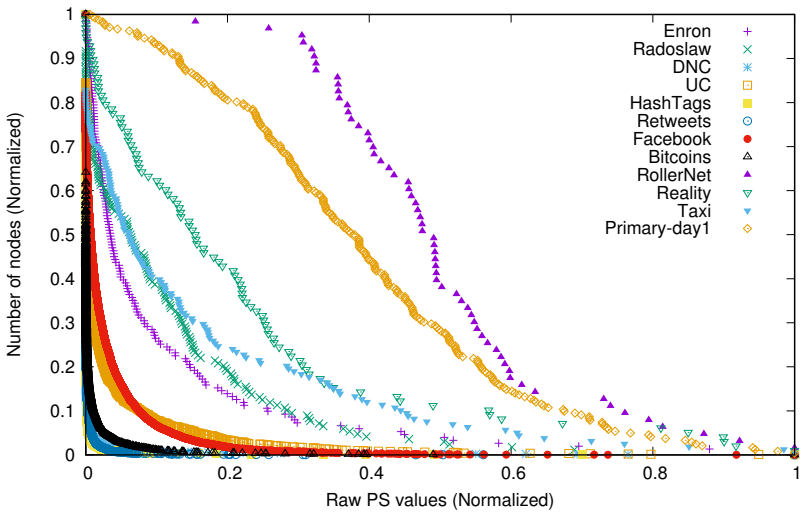



Figure 4.15: Inverse cumulative distribution of raw *PS*.

ferent from the other nodes than for the other datasets. From this method, distinguishing the other datasets that we know do not have a meaningful notion of global importance is not feasible. We discuss other approaches in the perspectives in Chapter 6.

### 4.5.2 Conclusion

In this chapter, we proposed several methods to replace or increase the efficiency of the complete process of computing the Temporal Closeness and $Dur_{top}$value. The first method consisted of increasing the value of $I$, thus having fewer computing instants. This decreases the computation time without significantly affecting the accuracy of the results. However, after a certain point the computation time becomes constant.

Therefore, we propose a second method that involves several strategies that exploit structural prosperities to sample out the important nodes. These strategies found up to 90% of the important nodes. These methods gave instantly the results, compared to several hours of computation for certain large datasets with the exact method. However, we observed that this method does not necessarily give the correct order for the sampled nodes.

Thirdly, in certain cases the value of $Dur_{top}$ can be required. For this, we observed that the values of $Dur_{top}$ tend to have a similar distribution across datasets. From the ranking of nodes, the number of nodes and duration of dataset, to generate estimations for these values.

Finally, we proposed a protocol that uses these three methods, to detect the top nodes and approximate their $Dur_{top}$ values. This protocol depends highly on the quality of the sampling. Thus, one perspective is developing a method to eliminate the mis-identified nodes.

We should also note that this protocol reduces the computation time extreme, as the sampling method is computed almost instantly in most cases even for large datasets. Additionally, computing the Temporal Closeness for the sampled nodes takes around one eighth of the original computational time. Finally, approximating the $Dur_{top}$ values is also almost instantaneous. Thus, globally this method is fast compared to the normal method.

# Chapter 5

# Ego-betweenness

A node's ability to relay information in a network is one of the most common criteria to evaluate its importance. This importance is often measured using the betweenness centrality [Freeman, 1977]. As we discussed in Chapter 1, several adaptations extending this centrality notion to the dynamical context have been proposed. These adaptations tend to be computationally demanding, which renders their use in real-world situations unfeasible.

Here, we concentrate on a special case of networks, Delay Tolerant Networks (DTN), in which end-to-end connectivity is not guaranteed. Unlike standard networks, these networks can be partitioned for long periods of time, with links between nodes absent for long durations (for more details, please refer to [Jain et al., 2004]) and, most often, a path requires waiting time on the intermediary nodes, before the required link appears. Real-world examples of this type of networks are disaster situations (diffusing warning notifications to victims) or intra-vehicle communication networks (diffusing traffic news). In these networks, several constraints exist such as: communication is not always guaranteed; global knowledge of the network may not exist; information transfer is not instantaneous and can be expensive. Thus, when diffusing messages in

these networks, choosing which node to pass the message is an important question. One way to select these nodes is using betweenness centrality. This bases the importance of a node, on the amount of shortest paths that go through it.

Protocols that use betweenness centrality or more general information about nodes to select them are known as *social-aware* protocols [Magaia et al., 2015, Zhu et al., 2013]. [Kim et al., 2014] used an ego-point of view to introduce a DTN routing protocol which considers the chance of future contacts occurring between nodes. In a similar manner, [Daly and Haahr, 2007] proposed a DTN routing protocol known as *SimBet*, which mixes a graph-based ego-betweenness centrality, which considers an ego-centric point of view. A similarity measure was also used. It measures how similar two nodes are in order to evaluate if a given node is an appropriate relay for message passing. Another good example is BubbleRap [Hui et al., 2011], in which the authors adapted the idea to temporal data by proposing a measure which is derived from the simulation of propagation through flooding. In these works and others, simulations often achieve good performances in terms of delivery ratio and cost when compared to benchmarks, which validates these approaches. However, the large majority of these protocols use a static definition of the betweenness centrality. Moreover, some of these definitions require the knowledge of the global structure of the network, which in practice is not always available.

For this reason, we introduce a novel adaptation of the betweenness centrality that takes into account real-world constraints. First, we consider dynamic graphs to take into account the temporal aspect. Second, as the global knowledge of network may not be available, we consider an ego-centric point of view, which also lowers the computational demand of this centrality. Finally, we think that in these situations, the novelty (how recent) of the information is a more interesting criterion than the speed of transfer or number of links required to transfer information between

two nodes. We therefore propose the notion of the most recent path instead of the classical shortest path. We evaluate the performance of this centrality in detecting the important nodes against other adaptations of the betweenness centrality.

## 5.1 Ego-centric vision

As previously mentioned, our goal here is to introduce a centrality measure that takes into perspective real-world constraints. We consider an ego-centric vision, *i.e.* we consider that a node does not have access to all the network but only to its neighborhood. In this section, we study the classic ego-graph definition and introduce the equivalent notion for the dynamic case.

### 5.1.1 Ego-graph and ego-dynamic graph

An ego-graph is composed of a node $e$ (usually referred to as *ego*), its links to its neighbors and the links between its neighbors [Everett and Borgatti, 2005]. In other words, it is the subgraph induced by $e$ and its direct neighbors. An ego-dynamic graph is the natural equivalent of this notion in the dynamic context. Therefore, an ego-dynamic graph is centered around a node $e$ and contains only the interactions between $e$ and any of its neighbors, as well as the interactions between any two neighbors of $e$.

Let us consider a dynamic a graph $G = (V, E)$, where $V$ is a set of nodes and $E$ is a set of edges in the form of $(u, v, t)$ where $u, v \in V \times V$ and $t$ is a timestamp. Each link stands for an interaction between nodes $u$ and $v$, taking place at instant $t$. If the interactions are directed from $u$ to $v$, we will refer to the network as a directed, otherwise, it is undirected. Additionally, instead of $t$, we will often use the notation $t_{uv}$, or $t_{u \to v}$ if the graph is directed, to indicate to which interaction $t$ is related. For a node $e \in V$ (ego), the ego-dynamic graph $G_e = (\mathcal{N}_e \cup \{e\}, E_e)$, where $\mathcal{N}_e$ is the set of neighbors of $e$, $E_e = \{(u, v, t) \in E | u, v \in \mathcal{N}_e \times \mathcal{N}_e\}$.

If we consider the dynamic graph of Figure 5.1, we can

observe in bold red the ego-dynamic graph obtained with $A$ as the ego node. It contains the nodes $A$ as well as $B$, $C$ and $E$ as they have direct links with $A$. Node $D$ does not belong to the induced graph, as it does not have a direct link with $A$, and naturally the link $(D, E, 3)$ does not belong to it.

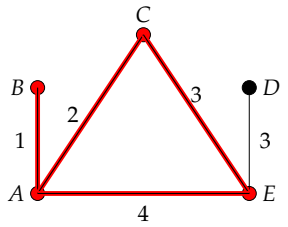



Figure 5.1: A toy example of a dynamic graph. The labels on the links indicate the instant in which the link occurs. In bold red the subgraph graph induced by the node $A$ as an ego node.

### 5.1.2 Paths

*Time-respecting paths*

Recall that a path in a dynamic graph is a sequence of links, in which each link occurs before the following link. We use here the same definition of path. A path from $u_1$ to $u_n$ is a a sequence of links $(u_1, u_2, t_{u_1u_2}), \ldots, (u_{n-1}, u_n, t_{u_{n-1}u_n})$ such that $\forall i$, $t_{u_iu_{i+1}} > t_{u_{i-1}u_i}$. Using this definition, information is not transferred instantly and there is a delay between the emission of a message and its reception. For example, the path $\{(C, E, 3), (E, D, 3)\}$ is invalid as both links occurred at the same instant, and information is cannot be transferred instantly.

*Most recent path*

As previously mentioned, we consider that relaying the most recent information is more important than transmitting old information rapidly. Hence, instead of computing the shortest path or fastest path here, we consider the notion of most recent path. A most recent path is *the path that gives the most recent information about the source*.

Formally, a *most recent path* between $u_1$ and $u_n$ at time $t$ is a path $(u_1, u_2, t_{u_1u_2}), \ldots, (u_{n-1}, u_n, t_{u_{n-1}u_n})$ such that $t_{u_{n-1}u_n} \leq t$ and $t_{u_1u_2}$ is maximum time among the times of all paths from $u_1$ to $u_n$ that occur before $t$. In other words, the information arrives to the destination at time $t$ at most, and the path starts at the latest possible time (most recent information).

Before proceeding with the computation of the most recent paths in dynamic graphs, we study the computation

of the shortest path in an ego-graph. This computation is quite straightforward, as the distance between two nodes is at most 2. Let us consider two nodes $u$ and $v$ that are neighbors of $e$. Nodes $u$ and $v$ are either directly connected or they are not. If they are directly connected the distance is 1, otherwise the distance is 2 because of the path $u - e - v$ and possibly because of another path $u - w - v$ passing via another neighbor $w$ of $e$. We now consider the situations that can be encountered in an ego-dynamic graph.

Figure 5.2 presents three situation; in each situation, each horizontal line represents a node, and the dotted red and blue arrows represent a path between two nodes (transfer of information). In all theses situations, we consider that the time of analysis is $\tau$. In the first case (top figure), a direct communication from $u$ to $v$ at time $t_{u\to v} = 2$ allows $v$ to know $u$'s status dating from time 2 (blue path in figure). If we consider the path that goes via $e$ (red path), it arrives at time $t_{e\to v} = 5$, which is after $t_{u\to v}$, however it provides information about $u$ that dates from $t = 1$. As we are interested here about the most recent information, at instant $\tau$ the most recent path is $\{(u, v, 2)\}$. In the second case (middle figure), the blue path that goes via $e$ allows $v$ to gain information about $u$ dating from $t_{u\to e} = 2$, while the red path that goes through $w$ reaches $v$ before the blue path but gives the status of $u$ at $t_{u\to w} = 1$. As a result, the most recent path from $u$ to $v$ at instant $\tau$ is $\{(u, e, 2), (e, v, 5)\}$. Finally, let us consider a more complex situation (bottom figure), with several possibilities for the status of $u$ to reach $v$: the direct path between both nodes at instant $t_{u\to v} = 2$, as well as a second path via $e$ that allows $v$ to have the status of $u$ from $t_{u\to e} = 1$. Both these paths require a one or two links, however they give information that is older than the one given by the path $\{(u, w_1, 3), (w_1, w_2, 6), (w_2, v, 8)\}$, which gives the status of $u$ from $t_{u\to w_1} = 3$. Thus, we see here that the most recent path can contain more links or take a longer time to reach the destination.

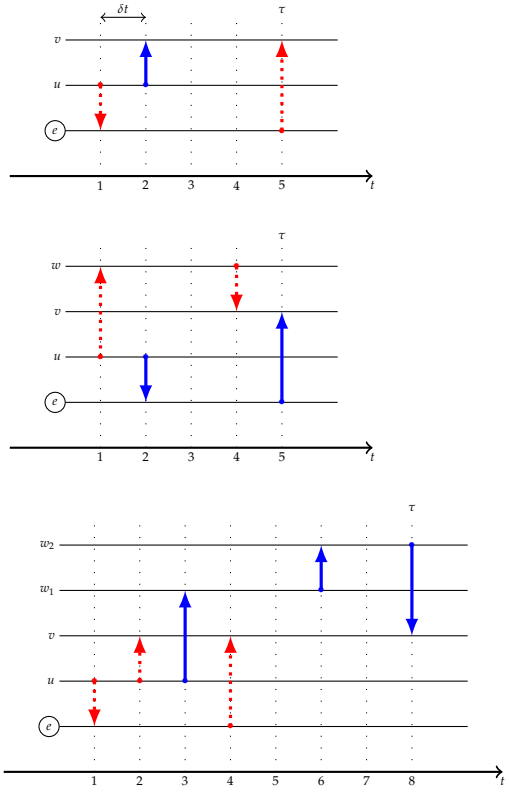



Figure 5.2: Various examples of most recent paths from $u$ to $v$ (in blue) at time $\tau$. In dotted red, alternative paths from $u$ to $v$.

To conclude, in this section, we presented the natural

adaptation of ego-graphs to the dynamic case. Additionally, we defined the *most recent path*, which is the path that delivers the most recent information about the source.

## 5.2 Ego-betweenness

### 5.2.1 Definition

Recall that the betweenness centrality of a node $u$ in a static graph is defined as the sum of the fractions of shortest paths between the pairs of nodes on which $u$ is located. From this, the classic ego betweenness definition was introduced in [Freeman, 1982]. For a node $e$ (*ego*), it is the sum of the fraction of shortest paths between the pairs of neighbors of $e$ on which $e$ is located. Formally this is defined for a node $e$ as:

$$C(e) = \sum_{i,j \in \mathcal{N}_e \times \mathcal{N}_e} \frac{g_{ij}(e)}{g_{ij}},$$

where $g_{ij}$ is number of shortest paths between $i$ and $j$ and $g_{ij}(e)$ is the number of shortest paths that go through $e$. Let us now introduce the ego-betweenness centrality in dynamic graphs. The ego-betweenness centrality at instant $\tau$ is defined in a similar way. It is the sum the fraction of most recent paths between the pairs of neighbors of $e$ on which $e$ is located. Moreover, to remain close to the spirit of the classic definition, we consider only the paths that are at most of length 2. Here, we apply the same definition of *length* as in the static case, *i.e.* the number of links. For example the blue path in Figure 5.2 (bottom) is not taken into account when computing the ego-betweenness definition.

Formally, we define the ego-betweenness for a node $e$ at instant $\tau$ in a dynamic graph in the following way:

$$C(e, \tau) = \sum_{i,j \in \mathcal{N}_e \times \mathcal{N}_e} \frac{p_{ij}(e, \tau)}{p_{ij}(\tau)}, \tag{5.1}$$

where $p_{ij}(\tau)$ is the number of most recent paths of length at most 2 from $i$ to $j$ at time $\tau$ and $p_{ij}(e, \tau)$ is the number

of these paths that go through $e$. This definition can be applied to both directed and undirected dynamic graphs. It should be noted that our point of view is retrospective, since at a time $\tau$, we measure the paths that already exist in the graph. Consequently, the ego-betweenness centrality depends on the past and not the future. Thus, at time $\tau$, the centrality can be seen as the extent to which the node has recently been important as a relay of information. This can be considered as an approximation of the chance of the node remaining important in the future.

### 5.2.2 Computation

We propose here an algorithm to compute the ego-betweenness of a node $e$. We consider the case of a directed dynamic graph, the undirected case being simple to deduce from it. As we are interested in the ego-dynamic graph around $e$, we only consider the links involving the nodes of $\mathcal{N}_e \cup e$, that is $e$ and its neighbors. We go through these links in chronological order.

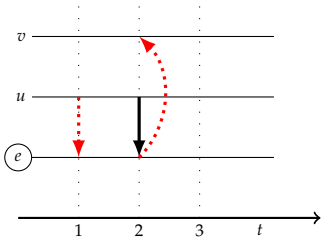



Figure 5.3: Toy example of a graph with three structural nodes and three links that occur at two instants.

At time $t$, we store the links that occurred at instant $t$. In addition, for each pair of nodes $u, v$ we store two informations: first, the timestamp of the last direct link between them; this allows us to construct the possible paths, and second, the timestamp of the most recent paths between both nodes as well the intermediate node on these paths. Before discussing how we compute the ego-betweenness, we discuss the need to store the links at $t$ as well as the last direct link. In Figure 5.3, we observe three links between the three nodes, and in dotted red the most recent path. At $t = 2$, we must keep the information about the link $(u, e, 1)$ in order to be able to detect the dotted red path. For that reason, when computing the ego-betweenness at an instant $t$, the links at instant $t$ are treated differently than other links.

To compute the ego-betweenness for a node $e$ at instant $t$, we compute all the most recent paths between all pair of nodes $u, v$. Three cases have to be considered:

1. there exists a direct link between $u$ and $v$ at time $t$, therefore $v$ has the latest information of $u$;
2. the links at time $t$ do not produce any interesting paths compared to the already stored paths;
3. that deliver more recent information than the previously computed most recent paths are created by using both the direct links stored and the links at instant $t$.

Note that in the last case, the links at instant $t$ can only be arrival links[1]; these links cannot be the start of a path, as there is no links that occur after them. Once all the most recent paths are computed, computing the centrality is straightforward.

1. The second link in a path of length 2.

**Algorithm 1:** Calculate Dynamic Ego-Betweenness Centrality

**Input**: Node $e$ , $\mathcal{N}_e$, links $E_e$ ordered by timestamp

1 **begin**
2 $\quad \mathcal{I} \leftarrow \emptyset$ ▷ Information about most recent paths
3 $\quad direct \leftarrow \emptyset$ ▷ Information about direct links
4 $\quad current\_links \leftarrow \emptyset$ ▷ Information about links of current timestamp
5 $\quad current\_t \leftarrow 0$
6 **for** $(u_t, v_t, t) \in E_e$ **do**
7 $\quad$ **if** $t = current\_t$ **then**
8 $\quad\quad current\_links[u_t \rightarrow v_t] = t$
9 $\quad$ **else**
10 $\quad\quad$ **for** $(u, v) \in \mathcal{N}_e \times \mathcal{N}_e$ **do**
11 $\quad\quad\quad c_e[u \rightarrow v] = 0$
12 $\quad\quad\quad$ **if** $u \rightarrow v \notin current\_links$ **then**
$\quad\quad\quad\quad$ ▷Obtain the nodes exist on a path between $u$ and $v$
13 $\quad\quad\quad\quad \mathcal{F} = \{x \in \mathcal{N}_e \cup \{e\} \setminus \{u, v\} \mid$
14 $\quad\quad\quad\quad\quad direct[x \rightarrow v] > direct[u \rightarrow x] \vee current\_links[x \rightarrow v] > direct[u \rightarrow x])\}$
$\quad\quad\quad\quad$ ▷Obtain from $\mathcal{F}$ the nodes with latest information about $u$
15 $\quad\quad\quad\quad \mathcal{Q} = \{x, x = \arg\max_{x \in \mathcal{F}} (direct[u \rightarrow x])\}$
$\quad\quad\quad\quad$ ▷Obtain from $\mathcal{F}$ the timestamp of the latest information about $u$
16 $\quad\quad\quad\quad latest = max_{x \in \mathcal{F}}(direct[u \rightarrow x])$
17 $\quad\quad\quad\quad$ **if** $I[u \rightarrow v] = (t_{u \rightarrow v}, \_) \wedge latest > t_{u \rightarrow v}$ **then**
18 $\quad\quad\quad\quad\quad I[u \rightarrow v] = (latest, \mathcal{Q})$
19 $\quad\quad\quad\quad$ **else if** $I[u \rightarrow v] = (t_{u \rightarrow v}, \mathcal{Q}') \wedge latest = t_{u \rightarrow v}$ **then**
20 $\quad\quad\quad\quad\quad I[u \rightarrow v] = (latest, \mathcal{Q})$
21 $\quad\quad\quad\quad$ **else if** $u \rightarrow v \notin I$ **then**
22 $\quad\quad\quad\quad\quad I[u \rightarrow v] = (latest, \mathcal{Q} \cup \mathcal{Q}')$
23 $\quad\quad\quad\quad (\_, \mathcal{Q}'') = I[u \rightarrow v]$
24 $\quad\quad\quad\quad$ **if** $e \in \mathcal{Q}''$ **then**
25 $\quad\quad\quad\quad\quad c_e[u \rightarrow v] = 1/|\mathcal{Q}''|$
26 $\quad\quad\quad$ **else**
27 $\quad\quad\quad\quad I[u \rightarrow v] = (current\_t, \{\})$
28 $\quad\quad centrality = \sum_{(u \rightarrow v) \in \mathcal{N}_e \times \mathcal{N}_e} c_e[u \rightarrow v]$ ▷Compute *centrality* at time *current_time*
29 $\quad\quad$ Print *centrality*
30 $\quad\quad current\_t = t$
$\quad\quad$ ▷Update *direct* using *current_links*
31 $\quad\quad$ **for** $u \rightarrow v \in current\_links$ **do**
32 $\quad\quad\quad direct[u \rightarrow v] = current\_links[u \rightarrow v]$
33 $\quad\quad current\_links \leftarrow \emptyset$
34 $\quad\quad current\_links[u_t \rightarrow v_t] = t$

▷Repeat lines 10 to 29

The algorithm is presented in Algorithm 1. For each pair of nodes, we compute all the possible paths of length 2 between them. From these paths, we keep only those that deliver the most recent information. Afterwards, we compare these paths against the paths stored. If both deliver information that date from the same instant we keep them all, else we keep the paths that give the latest information only. Finally, we compute the ego-betweenness centrality.

*Complexity*

Finally, we study the complexity of this algorithm. We can express it as a function of $N_e = |\mathcal{N}_e|$, and $M_e = |E_e|$, the number of links in the ego-centered dynamic graph. The complexity for the centrality of one node is $O(M_e N_e^3)$. We should mention that this algorithm is naive as we recompute all the paths from scratch (line 13). The computation can be much more efficient. For example, we can compute only the modified paths rather than computing all the paths from scratch at every instant. Hence, update the ego-betweenness rather than computing it from zero.

*Practical example*

To illustrate this definition on a practical example, we compute the evolution of the centrality of $e$ at time $t$, denoted by $\mathcal{C}(e,t)$, in the dynamic graph of Figure 5.4. At times 0 and 1, there is no most recent path going through $e$, so that $e$'s centrality is equal to 0. Starting from time 2, $e$ is located on the only most recent path from $u$ to $v$, so $\mathcal{C}(e,2) = 1$. But at time 3, there is a new most recent path from $u$ to $v$, which is $\{(u,v,3)\}$, and as $e$ is not located on it, $\mathcal{C}(e,3) = 0$. At time 4, we identify that $e$ is located on $\{(u,e,1),(e,w,4)\}$ which is the only most recent path from $u$ to $w$, that implies that $\mathcal{C}(e,4) = 1$. At time 5, $e$ is also located on a path from $w$ to $u$: $\{(w,e,4),(e,u,5)\}$, however there is another path which is as recent: $\{(w,u,4)\}$; $\mathcal{C}(e)$ is therefore increased by $\frac{1}{2}$ and is equal to 1.5. At time 6, there are two new most recent paths which allow

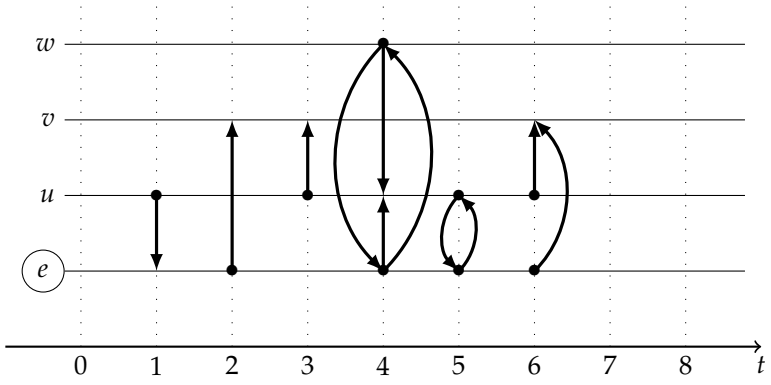



Figure 5.4: Toy example of a dynamic ego graph with four nodes, with $e$ as the ego node.

to relay information from $w$ to $v$: $\{(w, e, 4), (e, v, 6)\}$; and $\{(w, u, 4), (u, v, 6)\}$, the first one contributes for $\frac{1}{2}$ to $e$'s centrality, so that $\mathcal{C}(e, 6) = 2$.

## 5.3 Dynamical betweenness under scrutiny

In this section, we discuss the analysis processes. We shall compare ego-betweenness against the closest centrality metrics; metrics that capture a similar notion of importance. The first centrality metric we consider is coverage centrality, that we first studied in Section 3.1.4. This centrality quantifies the importance of a node $u$ by the number of shortest path that passes via $u$. This method quantifies the importance quite similarly to ego-betweenness. The second method is the snapshot method (presented in Section 3.1.2). The network is represented as a sequence of static graphs, each graph representing a fixed period of the dataset and being analyzed separately. For a node $u$ in a snapshot, the classic betweenness definition is computed; the computed value represents the node's importance during the whole period of the snapshot. We denote this method as *snapshot*.

### 5.3.1 Common base

To compare the three methods, we define a common base for the comparison. We use a *flooding method* as the reference. A flooding process consists of diffusing the infor-

mation as much as possible and as soon as possible. Once a node receives the message it starts sending it to nodes as it encounters them for the first time. We simulate one flooding protocol starting from each node and count the average number of times a node relays a message. This provides a common reference to all the other measurements. We should mention that the flooding method is used in practical contexts as a substitutes for a centrality measure [Hui et al., 2011]. Thus, it can be used as a technique to evaluate the capacity of a node to act as a relay. We shall refer to this method as *flooding*.

### 5.3.2 Datasets

For the comparison, we considered four datasets. Three are motion datasets: HyperText, Infocom, Primary. These datasets can be considered as a DTN networks. The centrality metrics we study can have other uses so we considered as well the Enron dataset, an email exchange dataset.

### 5.3.3 Comparison tools

After computing each of the centrality metrics, for each node a centrality value for every instant, at a regular interval, is obtained. As previously mentioned in Section 3.2.1, comparing the values of centrality is not necessarily informative. Thus, we rank the nodes using the *inverse competition ranking* method. This produces a ranking vector for each method for each computation instant.

To understand how each centrality metric differs, first we compute the Kendall tau correlation between the ranking produced by each metric against the ranking produced by the flooding method. We then the evolution of the Kendall tau over time for each method.

Secondly we consider the Spearman footrule correlation, defined formally, for two rankings $r_1$, $r_2$ as:

$$\mathcal{F}(r_1, r_2) = 1 - \frac{\sum_i |r_1(i) - r_2(i)|}{M},$$

where $r_x(i)$ designates the position of node $i$ in ranking $x$, $M$ is the maximum footrule distance, *i.e.* the largest possible distance which is equal to $2\lceil N/2 \rceil \lfloor N/2 \rfloor$ and $N$ is the length of the ranking (which is also the number of nodes in the network). This is complementary to the Kendall tau. However, it considers each node's ranks difference solely, without taking into account how its relationship of importance to other nodes is modified as the Kendall tau does.

## 5.4 Results

Before studying the results of the comparison, we discuss briefly the computation times of the three methods. For the four datasets, we obtain the following computation times.

| | Coverage | Ego-betweenness | Snapshot |
|---|---|---|---|
| Hypertext | 3.7 days | 6 hours | 0.5 seconds |
| Infocom | 12.5 days | 15 hours | 2 seconds |
| School | 83 days | 40 hours | 2 seconds |
| Enron | 4 days | 5 hours | 1 seconds |

Table 5.1: Computational time for coverage, ego-betweenness and snapshot in four datasets.

In the four datasets, snapshot performs faster. Nevertheless, ego-betweenness performs between 20 and 50 times better than coverage centrality. This is quite expected as at each instant coverage considers all the nodes, while ego-betweenness considers only the node's neighborhood. The computation of the coverage centrality at each instant has a complexity of $O(|N|log(|M|))$, where $|N|$ is the number of nodes and $|M|$ is the number of links in the datasets. Hence, computing the centrality for one node is in $O(|N|M|log(|M|))$. This is quite expensive compared to the complexity of ego-betweenness. Naturally, the snapshot method is much faster as it uses the static definition, which has a lower complexity.

Figure 5.6 presents the evolution of the Kendall tau between the ranking obtained for each centrality and the flooding method, for the four datasets. The first obvious observation is that at certain instants the correlation is constant. These instants represent periods of inactivity. During these periods, flooding method, ego-betweenness, and

coverage constantly attributes the last computed value. However, when it comes to snapshot, only the period of the snapshot is taken into account. All the nodes are attributed a centrality equal to 0, hence the correlation is close to zero, which can be considered as a random ranking in relation to the flooding's ranking. Secondly, ego-betweenness is the most correlated to flooding in most cases, compared to the two other centrality metrics. The snapshot method does not consider the past nor future; additionally, it can consider paths that do not respect the chronological order. Hence, it is normal to expect ego betweenness centrality to be more correlated than the snapshot method. On the other side, the low correlation of coverage centrality is unexpected, as it seems to rely on a similar intuition as ego-betweenness. Several reasons can be evoked to explain this observation: the fact that coverage centrality is not normalized by the number of paths going from one node to another or the difference between a fastest and a most recent path. However, we think that the most plausible cause is the fact that coverage centrality considers the whole stream, while ego-betweenness is restricted to the sub-stream around e. In a flooding experiment, a node which receives a message sends it to all its future neighbors, so this measurement also relies mostly on the local structure around e. Thus, the ego-betweenness as we defined it seems to be closer to what a flooding process would do.

Finally, for Infocom dataset, we can observe that the correlation for snapshot and coverage is highly related to the level of activity (see figure 5.5). Comparing with the evolution of Kendall tau, it is easy to notice that when the activity increases snapshot becomes more correlated, and that coverage becomes less correlated, and vice-versa. This can also be observed in the other datasets, however to a lesser extent[2].

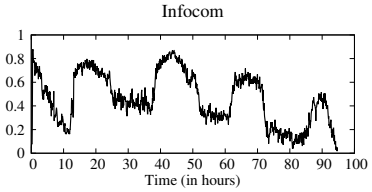



Figure 5.5: Evolution of fraction of the active nodes over time.

2. See Figures 2.1, 2.11 in pages 35, 45 respectively.

Figure 5.7 presents the Spearman footrule correlation. It confirms the global observations made on the Kendall tau correlation. In the four datasets, the ego-betweenness re-

mains the closest to the flooding method. Globally, the difference between ego-betweenness and the other methods is larger than that observed with the Kendall tau correlation.

This effect is likely due to the fact that when a node's rank is different, this changes its relation of importance with the other nodes and has a high impact on the Kendall tau, whereas this change only shifts the other node's ranking and affects less the Spearman footrule correlation.

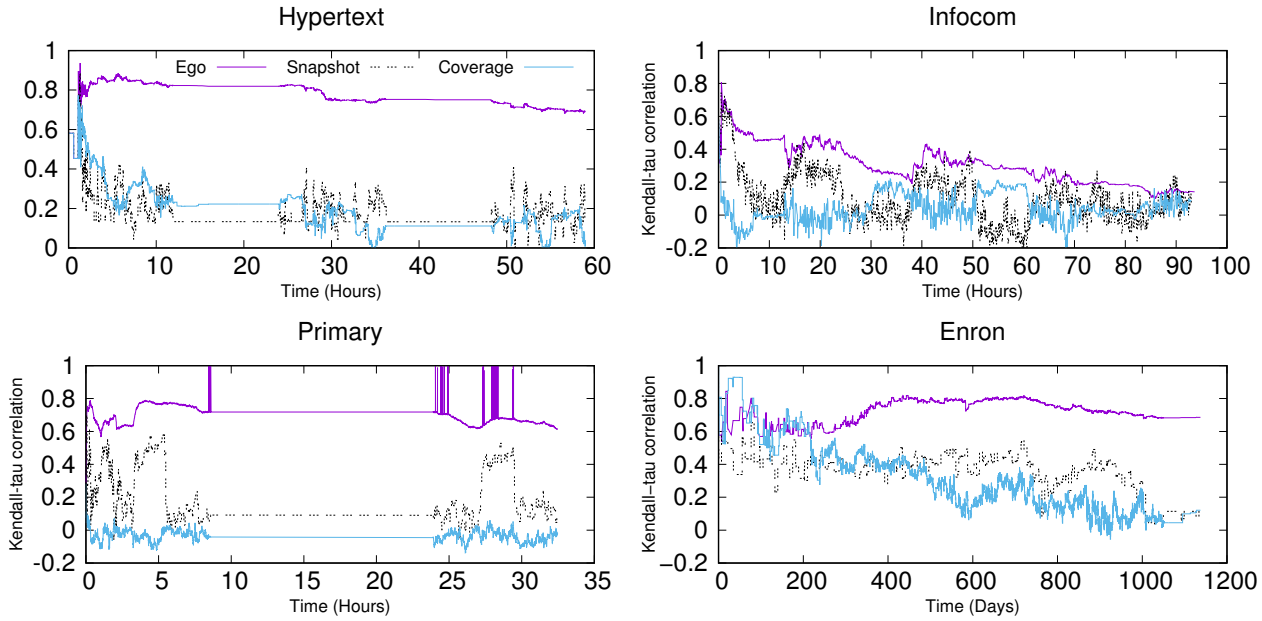



Figure 5.6: Evolution of the Kendall-tau correlation between each method and the flooding method over time for the four datasets: HyperText(Top left), Infocom(Top right), Primary(Bottom left), Enron(Bottom right).

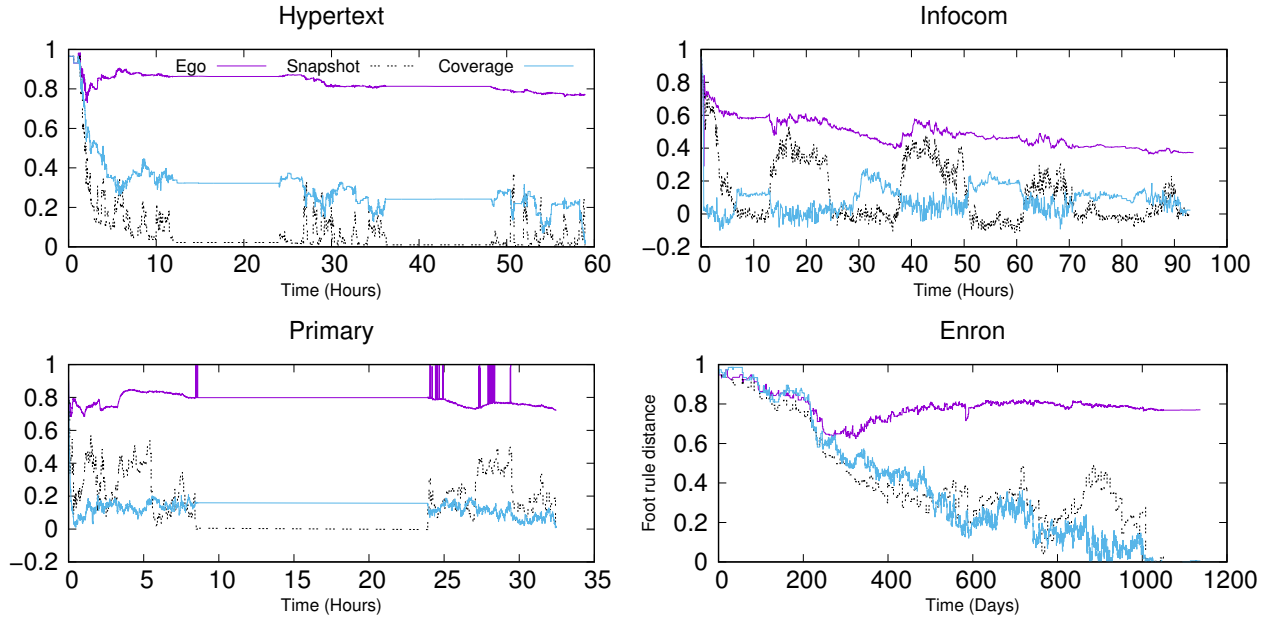



Figure 5.7: Evolution of the Spearman footrule correlation between each method and the flooding method over time for the four datasets: HyperText(Top left), Infocom(Top right), Primary(Bottom left), Enron(Bottom right).

## 5.5 Conclusion

In this chapter, we defined the ego-betweenness centrality in dynamic graphs as an extension of the ego-betweenness centrality in static graphs. We also proposed an algorithm to compute this value, which proved to be tractable on several real-world datasets. Its node-centered design allows to compute it with the mere knowledge of the neighborhood of a node. Such a property is desirable in many contexts, notably networks where there is no guarantee of an end-to-end connectivity. We compared the ego-betweenness to other centrality measurements in the literature, which also aim at assessing the utility of a node as a relay of information in a dynamic network. We observed that most of the time, it is relatively highly correlated to a flooding-based centrality measure. Therefore, we have good hopes that the ego-betweenness could be useful for opportunistic routing in DTN. In order to develop this application, the next step is to implement a comprehensive protocol that uses this centrality measure. Existing protocols based on centrality often use it jointly with a similarity measurement, thus we contemplate the idea of defining an ego-centered similarity measure that would be possible to compute in this context.

Chapter 6

# Conclusion and perspectives

In this thesis, we studied the use of graph theory in the analysis of networks such as social networks. We concentrated on centrality metrics which are used to detect the important nodes of a network. In particular, we focused on the metrics that take into account the temporal aspect of the data. In this chapter, we summarize our conclusions for each chapter and discuss the perspectives.

**Temporal centralities under scrutiny**

We studied several temporal centralities from the literature, representing the principal approaches that take into account the temporal aspect of the current literature. Each method proposes a different manner to take into account the temporal information in the network. Some methods base the importance of a node on the notion of the shortest path, while others base it on paths of all lengths. Our goal was to understand how they differ from one another. To do so, we proposed a comparison framework that compares any two centrality metrics. This framework compares them on two levels: the node and the global level. We compared several metrics using this framework. From this, we observed several phenomena. First, a node can be perceived as important by one metric and as irrelevant by another. Second, a node can be inactive at a given moment

(i.e. it doesn't not have any links) and yet be highly important. Third, nodes can be globally important while having a low global average centrality.

In the comparison framework, we introduced the $Dur_{top}$ value that represents the global importance of each node. Examining the distribution of this value showed that for some datasets, all the nodes have similar values; hence, no node is significantly more important than the others. Thus, in certain cases, the notion of global importance can be absent or meaningless, though of course a node may be more important than others at specific instants. Additionally, in this chapter, we compared different ranking methods. This comparison showed that in certain cases, the classical method misrepresents the nodes, especially when they become completely inactive.

This work opens several interesting perspectives. First, we would like to apply this approach to more datasets. Studying several datasets stemming from very different contexts may strengthen the conclusions drawn from this study. This would allow us to confirm our findings and might help identify specific patterns in the evolution of the nodes' importance that are context-dependent. In addition, our current classification of networks (email based; co-occurrence; social; movement) is based on the source of the network, rather than structural or temporal properties. Identifying such patterns would help produce a better classification for the networks.

Following up on this point, it would be very interesting for many real applications to be able to detect specific patterns in the evolution of centrality measures. This would certainly allow predicting which nodes are likely to be important in the future, which is key importance for several applications, ranging from protocols of communication to recommendation systems. For example, an individual is likely to communicate with his colleagues during the working days, however, during the weekend he will communicate more with friends and family. In both cases, he might be of importance, however not to the same in-

dividuals. Detecting such cycles and evolutions of importance can be useful for many applications such as diffusing information or vaccination campaigns.

We have seen that some of our observations allow the detection of nodes that have an atypical behavior, and/or moments where something unusual happens in the network's dynamics. We have observed this both when studying the time-evolution of the closeness of individual nodes, and when comparing different importance measures. This suggests that temporal centrality measures are relevant metrics when trying to detect anomalies in the network, which is a crucial question in many contexts [Wilmet et al., 2018, Heymann et al., 2012]. In particular, it seems that nodes that are important with respect to one metric but not to another have a particularly interesting behavior. A systematic comparison of different metrics would therefore certainly lead to very interesting insights about the considered dataset.

Throughout the literature, several centrality metrics consider different types of paths such as the fastest path, shortest path and foremost path. Comparing each of these centrality metrics can give a better understanding on how the choice of path affects the centrality. For example, we observed that a node can be relevant for one centrality but not another; this is likely to be observed with the different notions of paths as well. For example, betweenness centrality can depend on shortest paths, fastest paths or most recent paths; comparing this centrality among the different path types can give a better grasp on how each path notion affects the centrality measure. It would allow a better use of the centrality metrics for real-world applications.

### Approximation and Identification

After comparing several centrality metrics, we observed that these methods can be computationally expensive. Therefore, we studied deeply one of those methods: Temporal Closeness. Our goal in the chapter was to efficiently detect the nodes with a high global Temporal Closeness .

Temporal Closeness is computed every $I_{op}$ seconds; here we argued that this value can be increased. Which leads to computing the centrality less frequently and therefore requires less computational time. We observed that this reduces the computational time by at least half, without affecting the global ranking of nodes.

Even though the computational time is significantly reduced, it can remain consequential. Thus, we proposed a second method to identify the nodes of interest. By exploiting structural properties, we were able to propose three strategies that identify the globally important nodes. On most of the datasets that we considered, these strategies were able to find a significant percentage of the 25% most important nodes in a negligible time.

Afterwards, we considered the case where the duration spent as important by each node, *i.e.* the $Dur_{top}$ value, is required. By exploiting statistical properties such as the distribution of these durations or the duration for the most important node, we proposed a method to estimate these values. In the datasets with a strong notion of global importance, the proposed method estimated the durations with a good precision.

Finally, by combining these different approaches, we developed a protocol. This protocol identifies the top nodes and estimates the duration spent by each node as important. The drawback of this protocol is the mis-identified nodes[1]. This brings us to the first perspective of this work.

1. Nodes that are not supposed to be in the top 25%.

Identifying the false positives would be beneficial, as this would lower the error in the estimation of the $Dur_{top}$ values. One way to do so would be to study the evolution of the temporal closeness for the 25% identified nodes. We can examine each node and look for nodes that are much less important than the rest. Another approach would be to introduce another identification strategy. However, we argue that unless the strategy can identify all the nodes perfectly, mis-identifying nodes will remain an issue. Furthermore, developing a method to select automatically the $\alpha$ parameter for the two parameter based strategies would

increase the performance of these strategies as well as render them easier to use.

When estimating the $Dur_{top}$ values, we are confronted with one (resp. two) parameters that have to be defined in the case of an exponential (resp. sigmoid) curve. In the case of networks with an exponential curve, values between 3.4 and 4.0 yield the best results. Formally proving that these values give the best results for all the datasets or developing a method that returns the best value for the parameter is an interesting perspective. This would be harder for the case of a sigmoid curve. Firstly we only encountered one network that exhibits this curve, thus it is difficult to generalize the results. Therefore, finding other networks with such a curve and testing them should be the first step. This can result in finding networks that exhibit other forms of distribution, which would open a new challenge. But additionally, it can introduce a new classification system for the networks in question. Secondly, as the sigmoid curve requires two parameters, finding two optimal values is naturally harder than the case of an exponential that has one parameter. Nonetheless, the results on the tested dataset showed that only extreme values of the parameters produced a weak performance. Again this would be validated with the study of more networks. In the same direction, the use of machine learning methods can be a possibility to find the value of parameters. Another perspective would be developing the same protocol for other temporal centralities. For example, betweenness centrality is expensive to compute for the temporal case, so developing an identification strategy would be useful. We would be able to find the important nodes instantly, avoiding the expensive computation.

In this work, we considered only datasets with a strong notion of global importance. This was detected using the exact computation of Temporal Closeness, hence the need to develop a method to detect the datasets without a notion of importance, without computing the exact computation of Temporal Closeness. We think one way to do so is by an-

alyzing the evolution of Temporal Closeness over time for the identified nodes. Nodes in datasets with a meaningless global importance, might have a particular evolution, for instance fluctuate extremely. On the other hand, nodes in datasets with a strong notion of global importance would have a more stable evolution, constantly high for example.

**Ego-betweenness**

In this chapter we considered another centrality metric: the betweenness centrality. We argued that most adaptations do not take into account possible applications, as each application has its own restrictions. Here, we concentrated on the case of delay tolerant networks (DTN). To take into account the aspects of delay tolerant networks, we considered the notion of *most recent path*. We use this instead of the shortest path, as we claim the novelty of information is more interesting than the speed of its transfer. Additionally, we considered an ego-centric point of view, as again in DTN networks nodes do not have a global knowledge of the network. From this, ego-betweenness centrality was introduced.

We compared ego-betweenness with similar centrality metrics on several motion networks. This comparison showed that the results produced by ego-betweenness centrality were the closest to the flooding method, which is used by several DTN protocols to select the important nodes.

From this work, two main perspectives emerge. First the developing of a DTN protocol that uses this centrality. In practice nodes do not know their neighbors in advance, but this is required by the introduced centrality algorithm requires the neighborhood in advance. Hence, an approximated centrality could be computed. In such a DTN protocol, each node would keep an up-to-date value of its centrality. Whenever two nodes meet, the decision to exchange the information or not would be based on the centrality score of each node as well as a similarity measure, such as nodes belonging to a same community or a

distance between the node and the destination of the message.

For example, consider a vehicular network (taxis for example), in a city in which nodes exchange traffic news using bluetooth or wifi. When two taxis come in contact, they update their centrality value and then exchange centrality values to decide if exchanging information is beneficial. As mentioned above a similarity measure is required. If we consider that the message is to be delivered to a specific vehicle, its last known position can be used as the base of similarity. The closer a taxi is to the the destination, the more similar it is to the destination. Hence, it is more likely to successfully deliver the message.

This brings us to other perspectives. In our work, in order to compute the ego-betweenness, the ego node has to know in advance its neighbors and keep track of their interactions. However, in reality a node does not know in advance the nodes it will interact with in the future nor can track all communications between them. Hence, an approximation of ego-betweenness can be introduced, where the ego-betweenness centrality is computed with a subset of the links of the ego-dynamic graph.

In a vehicular network, one can imagine that the vehicles appear and disappear over time (the start and end of a taxi driver's shift). The absence and presence of a node is not represented in the temporal graphs. However, it is presented in the stream graph formalism introduced by [Latapy et al., 2017]. A perspective would be adapting our centrality method for this formalism to take into account the presence and absence of nodes.

Another major perspective is the use of this centrality as an approximation of the general betweenness centrality[2]. Rather than considering all the nodes in the calculation, we would consider a subset. In this work, we considered all nodes that are directly connect to the *ego* node, *i.e.* nodes at distance 1. We can increase this distance to consider nodes that are at distance larger than 1 from the ego node. Preliminarily results showed that considering all the nodes

2. Considering all the nodes in graph, instead of only the direct neighbors.

that are at most at distance 3 or 4 produced a good approximation of the betweenness. However, work remains in this direction before validating such an approach, such as considering more datasets.

### Global Conclusion

To conclude, in this thesis we concentrated on temporal centralities. We studied several aspects. We compared several centralities against one another; proposed several methods to identify nodes with high temporal closeness; proposed a novel ego-centric centrality based on the betweenness centrality. From this work, we observed that temporal closeness is not directly correlated to the snapshot method, where the static closeness centrality is used. Yet, when we considered simpler static measure such as number of link or duration, the results were more correlated. This is quite interesting as we can observe how basic measures can actually be more efficient and closer to temporal centralities rather than static measures, even if they are based on the same notions.

Additionally, as we analyzed several networks that have different natures, as they come from different sources, we observed that these networks share statistical properties such as the distribution of $Dur_{top}$ values. This indicates that temporal networks share certain aspects, even if they come from different sources. Exploiting these properties can render the computation of other temporal metrics more efficient.

Furthermore, in these networks, certain datasets did not have a meaningful notion of global importance; all the nodes were equally important. Finding more datasets that exhibit this should be interesting. In RollerNet, where all the participants become active and inactive at the same instant, and they all remain close to one another, these two characteristics are clear signs for the absence of importance. Thus, computing simple metrics such as duration or number of links should be considered as early indications for the absence of importance. Additionally, a sys-

tematic comparison of centrality metrics on these datasets can be insightful. It would be interesting to see if these datasets have any notion of centralities that bring to light a global notion of importance, if yes which centralities. Thus, studying several centralities that do not share the same notion of importance on these datasets would be interesting.

### Global Perspectives

Many other perspectives emerge from our work. We mostly concentrated on the importance of nodes. The definition of closeness that we mainly studied relies on the computation of temporal distance from one node to all the others. This is particularly relevant in our context, where we are concerned by the importance of a node in the dissemination process: a node will be important if it can reach many other nodes quickly. In other contexts however, the importance of a node $v$ may be more closely related to the fact that the distances *from all other nodes to* $v$ are short. This may be the case for instance in Web graphs, in which the importance of a page comes from the links towards it, not from its outgoing links. Comparing these two notions of closeness would lead to interesting insights.

In the same manner, we introduced the ego-betweenness centrality for a specific application. Other applications could require replacing the most recent path by other notions of path that are more relevant to the application. Additionally, as mentioned above, it would be interesting to understand how the different paths affect the centrality metrics, however in the ego-centric context.

In [Magnien and Tarissan, 2015], the authors studied Temporal Closeness on Enron. They observed a node that is inactive for most of the dataset, except for one instant where it has two links with two other nodes. From these two links, the node becomes the third most important node at this instant. This is an indication that specific links can be quite important. Hence, we think it would be quite

interesting to define a reliable method for identifying important links. On the one hand, this would lead to another approach for event detection in the network, complementary to the one sketched above. On the other hand, notions of link centrality have been successfully used in the case of static networks for community detection [Girvan and Newman, 2002]. Using a notion of importance in a dynamic network for dynamic community detection is therefore a promising idea. This would consist of applying the method proposed by Girvan *et al.*, however we would consider the temporal information. To do so, we can start by computing Temporal Closeness (or Ego-betweenness) and compute the globally important nodes. Afterward, we remove a certain number of nodes at specific instants. Once these nodes are removed certain paths will no longer exists, separating groups of nodes. These groups would correspond to our communities. This method can also be useful in local community detection in dynamic graphs. The difference from the classical community detection is that local community detection focuses on a set of nodes of interest, and it tries to find the best community for this set of nodes. Furthermore, as these methods require knowing only top important nodes, our identification methods can be of great use.

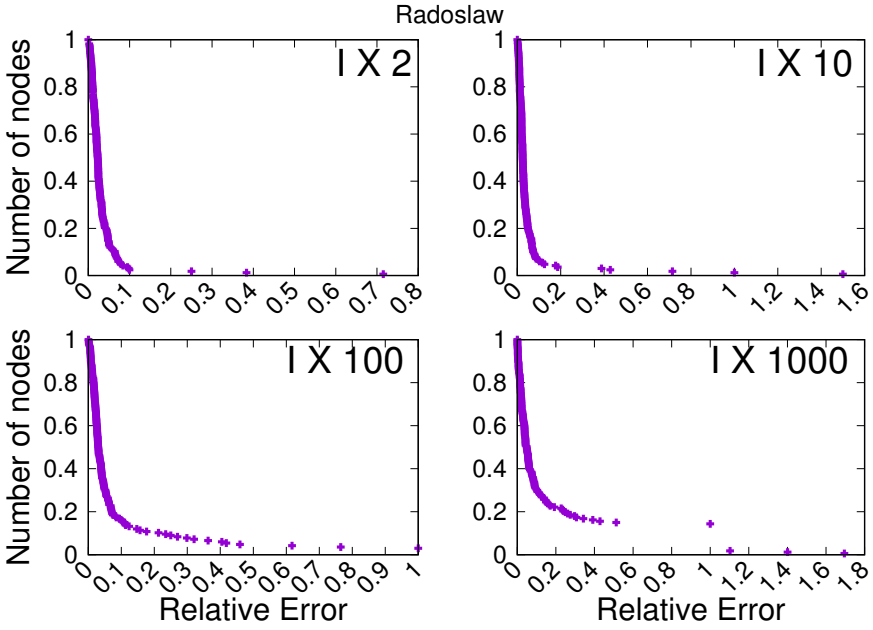
Finally, to our knowledge, there is yet no consensus on relevant generative models for dynamic networks. Since we have observed that different networks have different properties regarding the temporal closeness centrality, this is probably an important ingredient to take into account when proposing a new model. The distribution of global importance can be one characteristic for these models, as well as finer analysis of the evolution of the importance for each node.

# Appendices

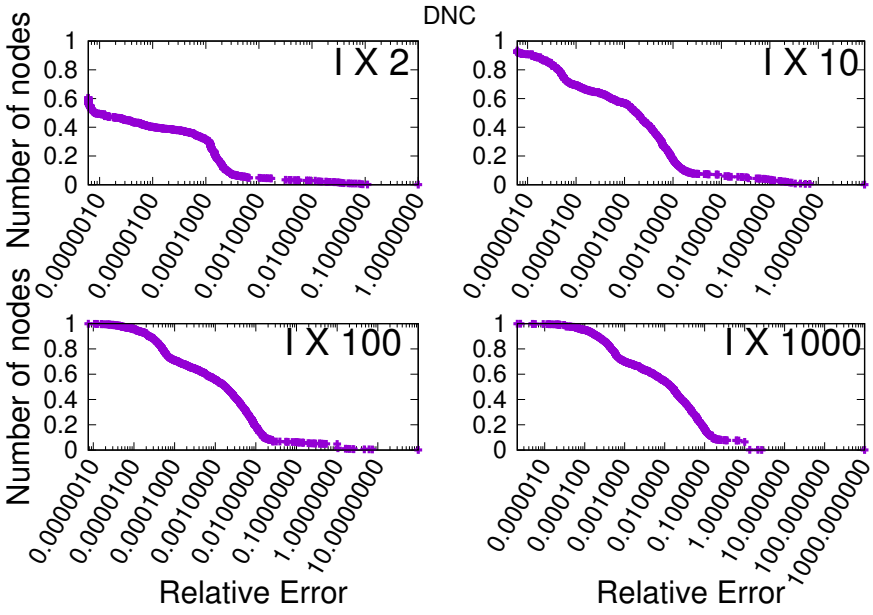
Appendix A

# Approximation and Identification

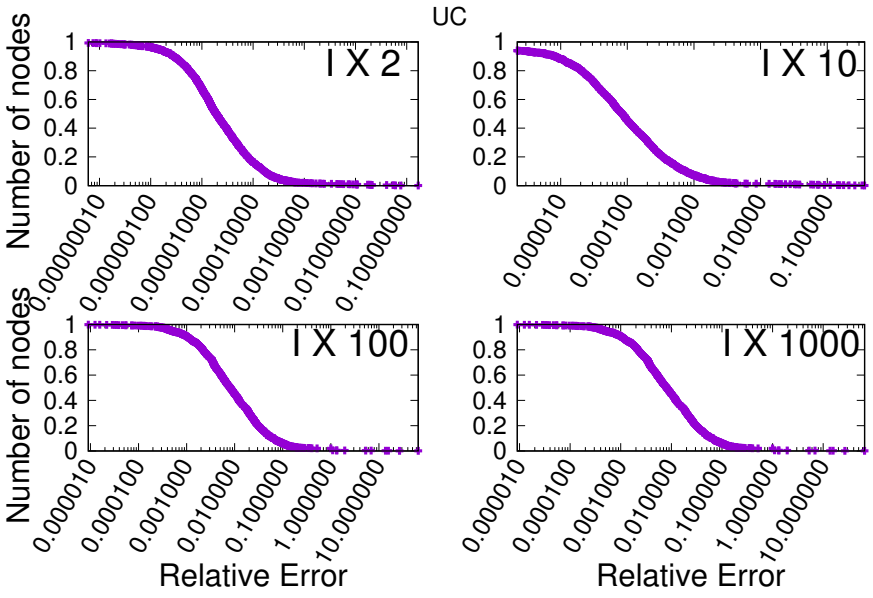
## A.1 Relative Error per multiple of $I_{op}$



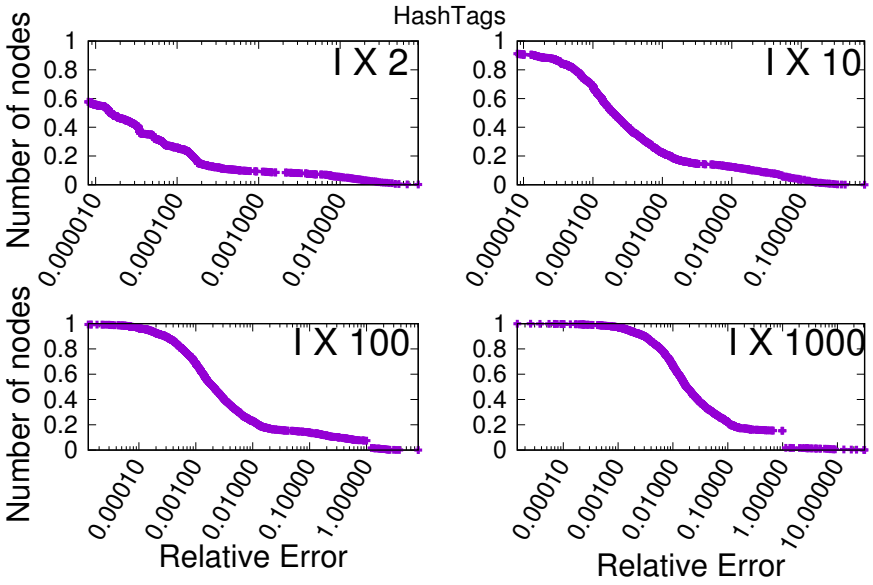

Inverse distribution of relative error for different multiplies of $I_{op}$ for Radoslaw.



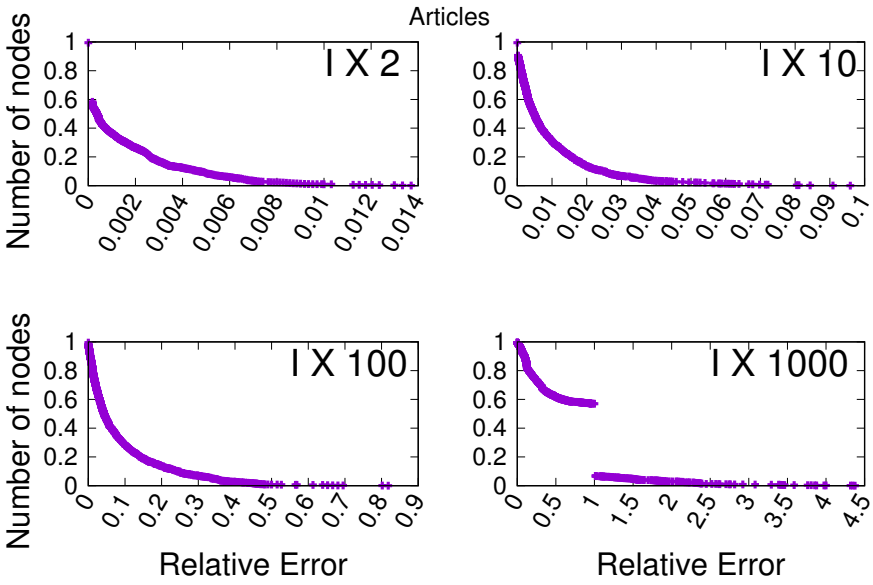

Inverse distribution of relative error for different multiplies of $I_{op}$ for DNC.



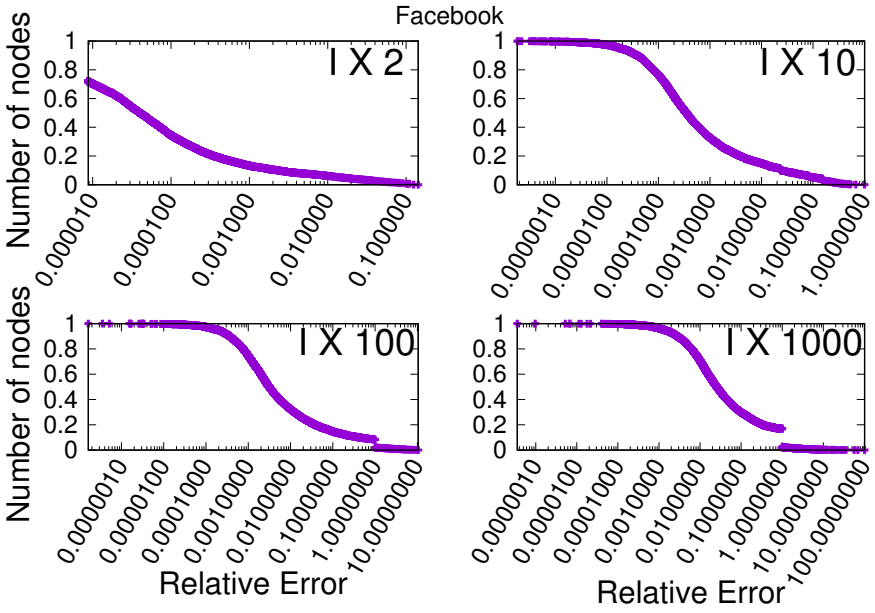

Inverse distribution of relative error for different multiplies of $I_{op}$ for UC.



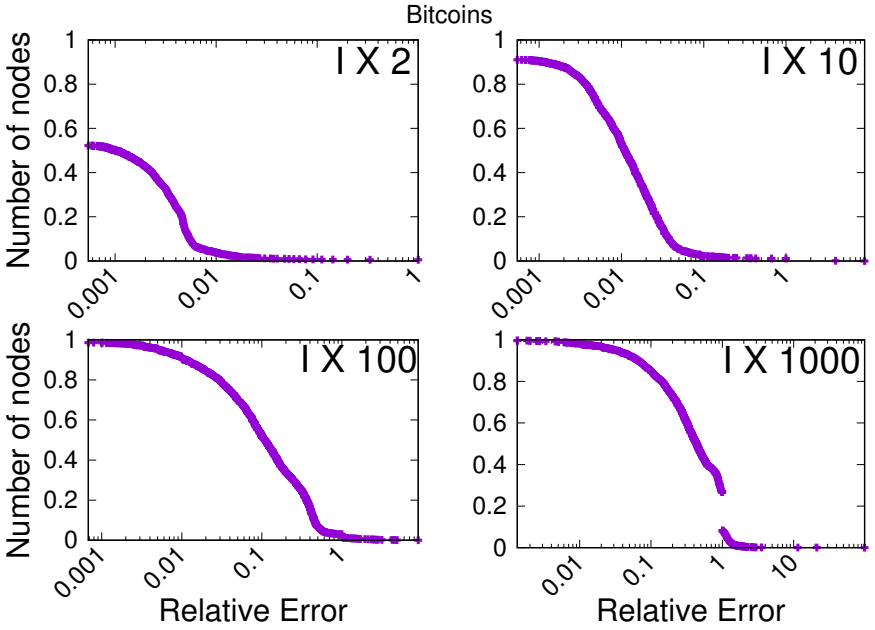

Inverse distribution of relative error for different multiplies of $I_{op}$ for HashTags.



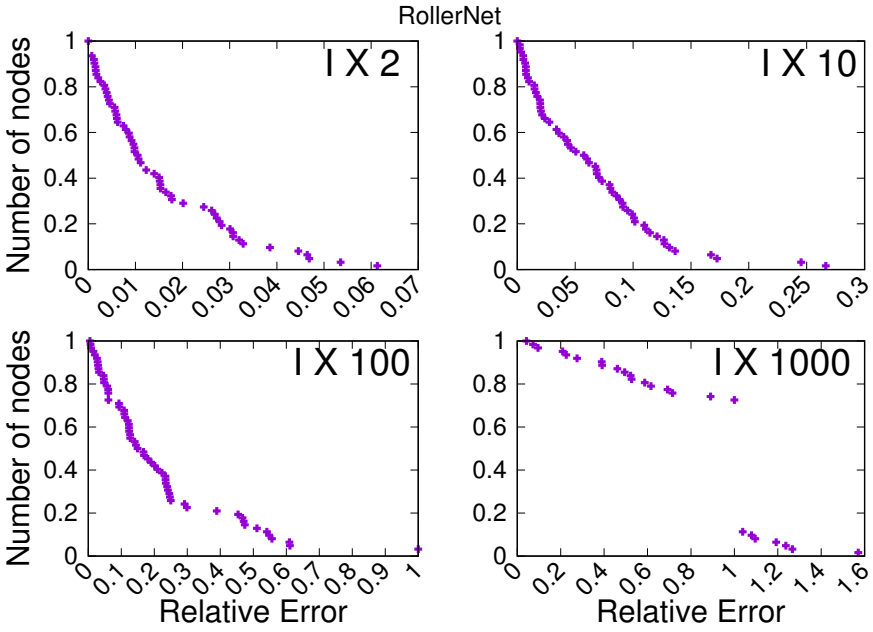

Inverse distribution of relative error for different multiplies of $I_{op}$ for Articles.



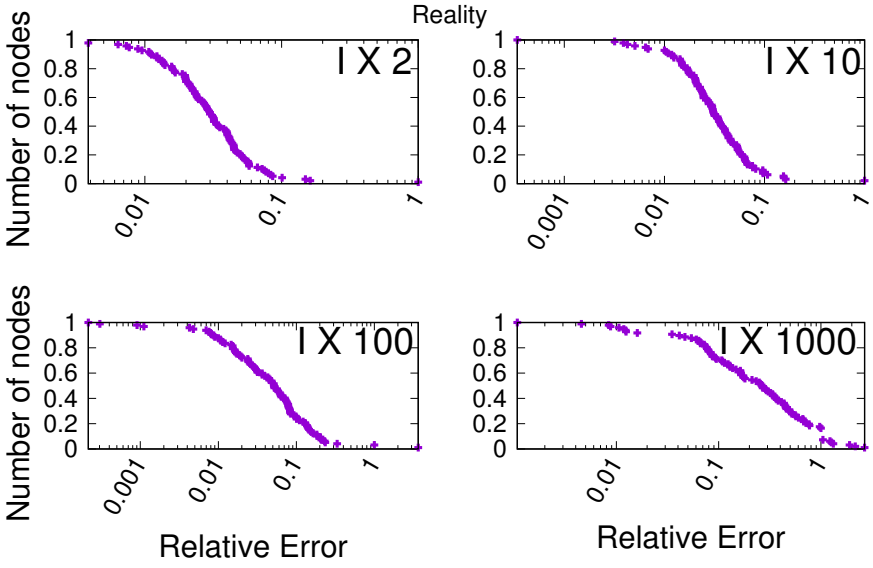

Inverse distribution of relative error for different multiplies of $I_{op}$ for Facebook.



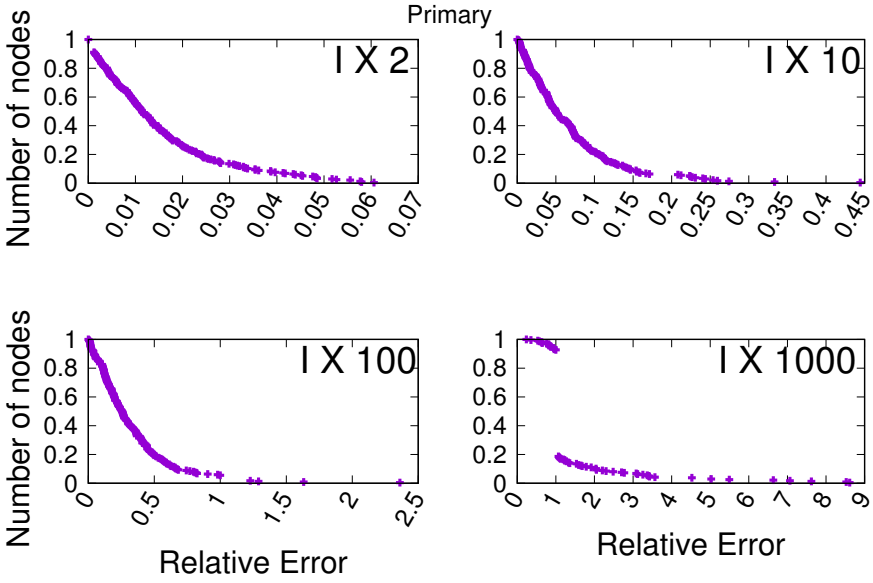

Inverse distribution of relative error for different multiplies of $I_{op}$ for Bitcoins.




Inverse distribution of relative error for different multiplies of $I_{op}$ for RollerNet.




Inverse distribution of relative error for different multiplies of $I_{op}$ for Reality.




Inverse distribution of relative error for different multiplies of $I_{op}$ for Primary.

Appendix B

# P2PTV Multi-channel Peers Analysis

In this Chapter, we present work that was done on P2PTV networks. This work was part of the master internship and was completed during the first months of the thesis, which was published [Ghanem et al., 2016]. In this paper, we analyzed several P2PTV traces that were captured using 10 PCs. It allowed us to have a better understanding on how these P2PTV applications work, even through we have no access to their code.

## B.1 Introduction

After being the support of the data and voice convergence, the Internet has become one of the main video providers (live TV or video on demand). Those multimedia services were previously confined to the video broadcasting infrastructures (terrestrial, satellite or hybrid fiber/coax). The transmission of broadcasting quality TV streams in High Definition (or soon in Ultra High Definition 4K/8K) requires the use of huge amount of communications networks resources. The development of dedicated technologies to distribute these contents is either local and limited to a residential operator (IPTV), or global but complex and expensive (CDN).

The alternative to these limited or expensive technologies could be partly or completely based on P2P. In this

context, peers communicate via virtual mesh networks (overlays) that connect them. The dynamic topology of these overlays depends on many parameters: location of resources, network status, internal mechanisms of peers, as well as the distributed content and the behavior of the peers directly involved as consumers of content.

In the case of TV streams, specific constraints require significant adaptation of P2P (specifically related to real-time aspects). A new application class realizes this kind of service: P2PTV. For these applications, the content consists of audio/video streams to distribute in real-time to a large number of receivers. The large number of streams and their intrinsic real-time characteristics generate timing constraints which are difficult to guarantee in the considered dynamic environment. Strict compliance with these constraints impacts directly on the peer's quality of experience and thus on his behavior, which in turn impacts the overlay.

P2PTV applications broadcast hundreds of channels, each carrying a live audio/video content to thousands of peers. Each channel corresponds to an overlay integrating peers wishing to receive its contents, and these peers can switch channels at any time (usually depending on the contents) adding an extra dynamic factor.

It is this dynamical aspect we intend to address in the present paper. Although several works have been proposed to measure and analyze the activity on P2PTV infrastructure, tracking the presence of peers active on different channels remains challenging. Here we show that we can rely on non-invasive measurement techniques such as Wireshark to track peers switching from one channel to another. To do so, we rely on 2 datasets obtained by measurements campaigns that coordinate several points of measure on a well known P2PTV infrastructure; we show that although the views obtained by such a measurement approach are partial, they are sufficient to detect multi-channel peers and highlight particularities in their behavior, thus leading the way to a more in-depth investigations

on the subject.

The remainder of the paper is organized as follow: we start by presenting existing works related to the analysis of P2PTV applications (Section B.2) before presenting the dataset used in the present paper (Section B.3). Then we turn to the study of the information contained in the dataset in order to analyze the behavior of multi-channel peers. We start by exploiting the dataset by aggregating all the information (Section B.4) before refining our analysis using narrowed views provided by sliding time windows (Section B.5). Then we show that comparing the two datasets gives insight on how diffusion through P2PTV has evolved (Section B.6). Finally we conclude the paper by presenting the perspectives opened by the present study (Section B.7).

## B.2 Related Works

Several studies and experiments have been done to analyze P2PTV applications [Agarwal, 2007, Alessandria et al., 2009, Spoto et al., 2009, Valenti et al., 2009, Tang et al., 2009, Rossi et al., 2011, Bermolen et al., 2011]. Rossi *et al.*proposed a framework for comparing P2P applications [Rossi et al., 2011] in which they define a set of observable features related to the protocols used by the applications. They highlighted the main similarities and differences between several P2P applications. In particular, they provided the key elements that open the way to passive analyses one can use when the applications are proprietary and no internal access is provided. Spoto *et al.*presented an investigation of PPLive using both active and passive measurements [Spoto et al., 2009]. Using a crawler, they were able to classify the traffic into three classes as well as to show that only 15% of peers could be considered as active peers, revealing the potentials and limits of PPLive active measurement strategies.

Other works have been done on a more quantitative perspective [Hei et al., 2007, Jia et al., 2007, Hoßfeld and Leibnitz, 2008, Wu et al., 2009]. Hei *et al.*proposed for instance a large

scale measurement study of P2PTV, using a PPLive dedicated crawler [Hei et al., 2007]. By collecting a huge amount of data in different scenarios, they have shown that P2PTV peers have the same behaviour as IPTV users. They also demonstrated the existence of a small set of super-peers that highly contribute to the video uploading. Similarly, using a crawler, Jia *et al.*tried to characterize PPStream [Jia et al., 2007]. They were able to find certain characteristics such as geographical clustering, arrival/departure patterns and playback quality.

Magharei *et al.*proposed a study on the structure of networks that most P2PTV applications used. They examine key issues with such structures and how bottlenecks can appear [Magharei and Rejaie, 2006].

By passively studying the traffic in P2PTV infrastructures, Silverston *et al.*were able to compare different applications pointing out their similarities and differences [Silverston et al., 2009]. Looking more deeply into the traffic, they discovered that signaling traffic tends to have a large inter-packet time while video traffic tends to have a smaller one. They also looked into peer behavior, revealing that the vast majority of peers tend to receive data more than they send, pointing out potential reciprocity issues.

There are also few studies more specific to the peer behavior and the multi-channel observations [Cha et al., 2008, Mendes et al., 2010, Wang et al., 2013, Mizutani et al., 2015]. Wang *et al.*analyzed the traffic that is characteristic to peers switching from one channel to another [Wang et al., 2013]. Using the most popular P2PTV applications such as PPLive or SOPCast, they monitored a channel for a given period and then suddenly changed to another one. They revealed that switching has a huge impact on the network efficiency as it increases the overload and adds a significant overhead. Finally, Mitzutani *et al.*were able to detect video servers as well as to find new characteristics of PPTV by monitoring multi-channel PPTV traffic [Mizutani et al., 2015].

| Property | 2013 | 2015 |
|---|---|---|
| Duration | 14 hours | 7 days |
| Number of channels | 12 | 10 |
| Number of peers | 100 809 | 289 710 |
| Maximum number of peers per channel | 21 518 | 96 258 |
| Average payload size | 504 B | 408 B |
| Total payload size | 193 GB | 601 GB |

Table B.1: Properties of the dataset

## B.3 DataSet

In this section, we present the datasets that were used during this work. It consists of two distinct measurement campaigns conducted on PPTV at a different time. The key aspect of those campaigns is that they coordinated traffic measurements from different points of measure. Concretely, the measurements were conducted on several PCs each running the application on a different channel thus, from the application's point of view, they acted as a regular peer. Every PC had an Internet connection provided by FLET'S HIKARI NEXT, 100 Mbps optical access service via Plata HIKARI Mate as an ISP in Japan. For capturing and monitoring traffic, Wireshark [wir, ], a well-known packet sniffer, was running on every measurement PC during the campaigns. Therefore we have the totality of the traffic that has been sent and received by our machines.

The first dataset was extracted from a 14 hour long traffic measured on December 2013 using 12 points of measure, while the second was extracted from a 7 day long traffic measured on July 2015 using 10 points of measure. We shall refer to those datasets later on as 2013 and 2015 respectively.

Table B.1 presents the global properties of both datasets. We can particularly notice the huge amount of data exchanged (193 GB and 601 GB for 2013 and 2015 respectively). It is also worth noticing that, although 2015 dataset is way longer (12 times longer than 2013 dataset), it exhibits a less dense traffic than expected (the total payload size is for instance only 3 times higher), which is partly

due to the lower number of channels.

As mentioned in many previous works [Hei et al., 2008], traffic generated by such applications can be shared out into two categories. Control (or signaling) traffic which could be either a heart beat signal, a peer's list exchange or buffer maps in form of a bit vector, representing the data a peer has available or missing from its video buffer. The second kind of traffic is data (or video) traffic which is transferred in the form of data chunks.

Most P2PTV applications were initially designed as a P2P mesh-based architecture [Magharei and Rejaie, 2006] including PPlive and PPStream [Jia et al., 2007]. Nowadays, most applications use hybrid P2P infrastructures with super-peers to guarantee that the viewers receive a better quality. Each transmitted channel by a P2PTV application would have its own P2P mesh-based network, which contains two types of peers. The first are the super-peers that are servers. They are active the whole time and appear in more than one P2P mesh in the same slot of time with the goal of maintaining the infrastructure of such systems. The second type is the regular peers that might appear in more than one channel due to switching behaviors [Wang et al., 2013].

## B.4 Global analysis

In this section, we exploit all the information contained in the collected data in order to detect multi-channel peers. We start by presenting global properties of the dataset (B.4.1) that complete those provided in Section B.3. Then we focus on the topics of this paper, namely the presence of multi-channel peers (Section B.4.2). Note that in the rest of the section, if not mentioned otherwise, we will present the results on the 2015 dataset as it is the more recent and the larger one.

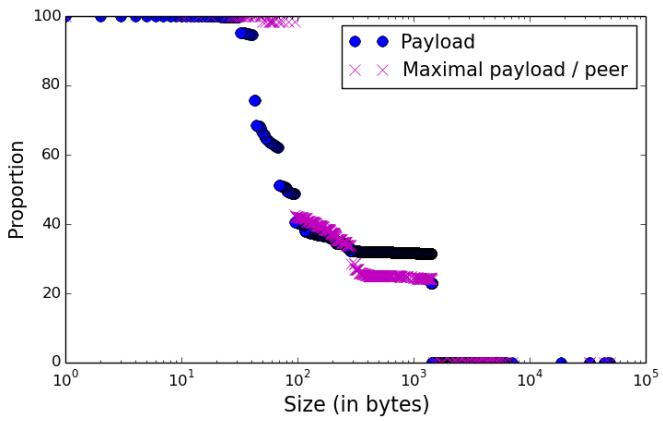



Figure B.1: Inverse CDF of payload size and maximal payload size/peer

### B.4.1 Tracking exchanges of video content

As mentioned in Section B.3, since we monitored the entire traffic using Wireshark, it is necessary to distinguish exchanges depicting the activity of a peer watching a TV program from the traffic dedicated to controlling the P2P infrastructure. To do so, it is reasonable to assume that a peer actively watching a TV program will trigger exchanges of video content, thus leading to a stream of packets with a significant size.

Figure B.1 shows the inverse cumulative distribution of the payload size (plain circles). One can clearly observe two regions. The first region involves packets smaller than 1000 bytes (68%) while a second region involves packets larger than 1000 bytes (32%). Obviously, the first one is related to control traffic while the second one can be categorized as video exchanges. In order to simplify the analysis, we will further make the assumption that any packet whose size is less than 1000 bytes is not a video content. Furthermore, a peer involved in at least one traffic containing a video will now be referred to as an *active peer*.

In order to get a better image of the peers behavior, we also display on Figure B.1 the inverse cumulative distribution of the maximal payload size a peer has exchanged (cross dots). In other words, a $(x, y)$ dot in this plot indicates that $y$% of the peers have exchanged packets with $x$ bytes at most. This plot reveals that only 25% of the peers

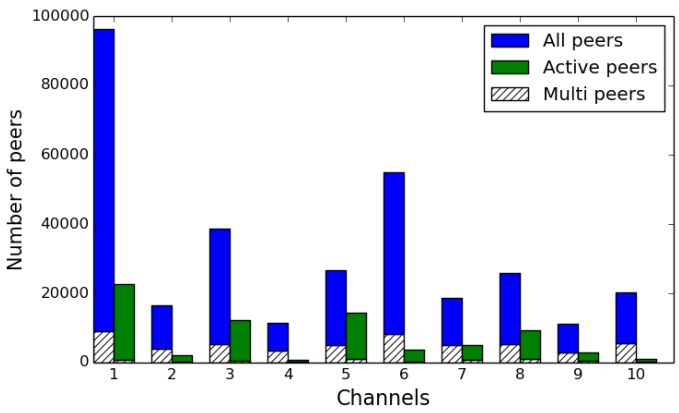



Figure B.2: Distribution of the peers over the channels.

are actively involved in video traffic

### B.4.2 Presence multi-channel peers

Turning now to the analysis of multi-channel peers, we start by studying the proportion of peers identified in several channels. It is worth remembering that the dataset consists of partial and independent measurements of the 10 channels (Section B.3). Thus, it is absolutely not guaranteed to detect such a behavior.

Figure B.2 presents the distribution of the peers over the 10 channels. For each channel, we show the number of *all peers* detected (left bar in blue) and the number of active peers (right bar in green) as defined above. In addition, we show for each of these quantities the fraction of the peers that are also detected in at least one other channel (bottom part of the bars with hatched lines). We will therefore refer to those peers as *multi-channel peers* (or multi peers for short).

The chart shows that the majority of the peers are concentrated in three channels. More importantly, it answers the first question raised in this paper which confirms that the measurement approach on which we rely on enables us to detect peers that appear in several channels. It turns out that 8% of the peers are multi-channel peers. Moreover, one can notice that this statement still stands even if we focus only on peers that exchange video contents, although

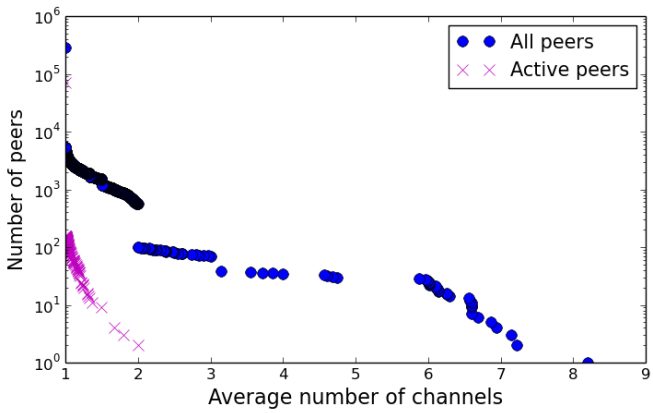



Figure B.3: Inverse CDF of the average multi-channel presence.

the ratio then drops to 0.8% of the total number of peers. However, the ratio of multi-channel peers actively watching a TV program still involve 3% of the active peers, thus revealing that a non negligible fraction of peers exchanging video content are involved in several channels. Note that this percentage could be overestimated if distinct active peers are recorded with a single identifier, which would be the case if they are behind a NAT. Although we did not investigate deeply this question in a systematic manner, we manually looked in detail the most susceptible peers of the dataset. Our preliminary results show that this is not the case and that those IPs detected on several channels depict a unique peer.

## B.5 Exploiting sliding time windows

The results presented in the previous section are interesting as they highlight the presence of multi-channel peers but aggregating all the information contained in the dataset prevents further refinement regarding the real behavior of the peers. In particular, it does not allow to distinguish a peer that switches between different channels (referred to further as a *switching peer*) from a peer that stops watching TV programs and starts watching another one way later on.

To overcome this issue we propose in the present sec-

tion to rely on the view provided by sliding time windows. More precisely, we sliced the whole dataset into non-overlapping windows of similar size and studied whether it enables an in-depth analyses of multi-channel peers. As one can expect, the size of the window becomes a key parameter in this approach. Since we focus on tracking the presence of switching peers, the size has to be short enough to discard peers that disconnect but it also has to be long enough to be able to detect the presence of the peers in several channels. Therefore we decided to use a 1 minute size window.

In the following sections, we will investigate how this approach enables us to distinguish between different types of peers (Section B.5.1) and different types of super-peers (Section B.5.2).

### B.5.1 Different peer behavior

By relying on short-time windows, we are now able to detect peers present *simultaneously* on different channels. In particular, we can determine the number of peers present in different channels for each slot of 1 minute. Besides, when such a peer is detected, one can track how many channels it is involved in.

Figure B.3 presents the inverse cumulative distribution of the average number of channels on which a peer is simultaneously present; for all the peers (plain circles) and for active peers (cross dots). In both cases, there is a large amount of peers involved in 1 channel only, then the value decreases smoothly between 1 and 2. However, when it comes to values higher than 2, active peers (cross dots) are no longer present. While in case of all peers (plain circles), a few peers are close to 4 or 5 channels in average. Finally, it is remarkable that some peers appear in almost all the channels (especially when computing the average value) when using such a short-time window. This indicates an unusual behavior and it is reasonable to consider such peers as *super-peers*, *i.e.* usually servers active the whole time whose purpose is to maintain the efficiency

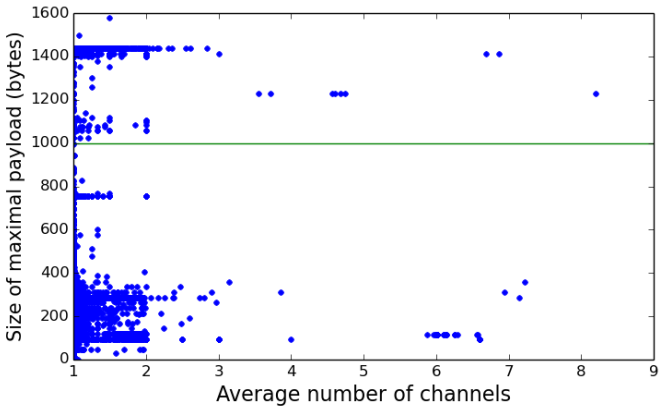



Figure B.4: Average multi-channel presence *v.s.* maximal payload size

of the infrastructure (see Section B.3).

### B.5.2 Different super-peers behavior

The detection of *video injectors* raises the question of the role of super-peers present on several channels in the infrastructure. In particular, do they all participate actively to the diffusion of video content or do they also regulate and control the traffic over the P2P infrastructure?

In order to have a better understanding of this question, we compare in Figure B.4 the average number of channels in which a peer is simultaneously present to the maximal size of a payload it sent/received. Each dot standing for a different peer, one clearly finds the super-peers detected previously but we can now refine their role: obviously, most of them only support control traffic since all the packets they exchange have a very low size. Thus, these super-peers are present only for administration and surveillance purposes, while the *video injectors* are clearly present in the top left part of the figure mixed with regular users. While the three super-peers in the top left who seem to produce video traffic, were only doing so in only 2 channels.

## B.6 Extending the analyses

In order to strengthen the results presented so far, we present in this section different analyses that allow to sustain our

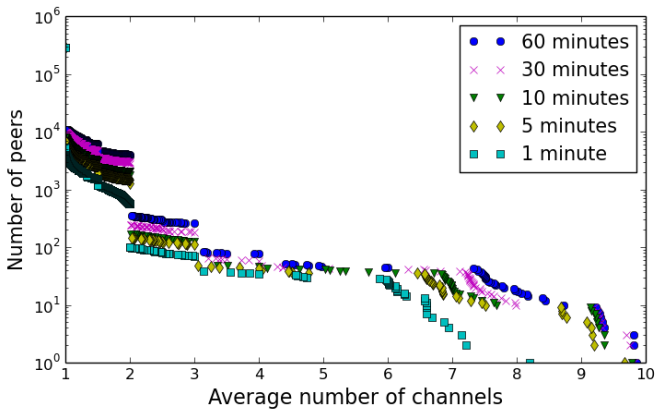



Figure B.5: Inverse CDF of the average multi-channel presence for different time windows.

former claims. In particular, we show that the size of the time window has little impact on the former conclusions (Section B.6.1). We also compare the results obtained on the 2015 dataset to the ones obtained on the 2013 dataset (Section B.6.2). This allows to give insights on how diffusion through P2PTV has evolved.

### B.6.1 Impact of the size of the window

All the results presented in Section B.5 are strongly related to our choice of using a 1 minute time window. Although we claim that this choice is reasonable regarding our main objective (tracking multi-channel peer), one might wonder whether another value would have altered our conclusions. We therefore present in Figure B.5 a figure similar to Figure B.3 but for different time windows (1, 5, 10, 30 and 60 minutes). The figure shows that the main effect of increasing the size of the time window is to increase the number multi-channel peers. This is completely expected since more times gives more opportunity to switch between channels. But if the size of the window impacts quantitatively the results, the overall observations remain qualitatively valid. In particular, all curves present a sharp breach around the value of 2 and another one after 3. It is worth noticing that increasing the size of the time window also reveals that some super-peers cover all of the ten channels. Thus, this confirms our assumption that they do

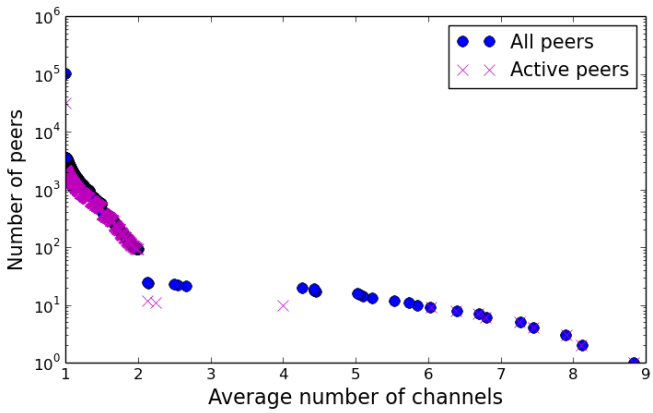



Figure B.6: Inverse CDF of the average multi-channel presence (2013 dataset).

not correspond to regular peers.

### B.6.2 Comparisons of the datasets

We turn now to the comparison between the two datasets. Using the same exact method explained in Section B.5.1), Figure B.6 presents the inverse cumulative distribution of the average number of channels on which a peer is simultaneously present. Unlike Figure B.3, the breach between values 2 and 4 is more prominent. More interesting is that we can see that super-peers are also present on the active peer curve (cross dot). Thus, we can deduce that between 2013 and 2015 the roles of super-peers have changed. In 2013, a super-peer could have been responsible for injecting video contents as well as administrating the infrastructure. In contrast, in 2015 each of these two roles seems to be taken in charge by a specific super-peer.

Finally, Figure B.7 compares the average number of channels to the maximal payload size it has sent/received. We compare the figure with the one obtained on the 2015 dataset. Like in dataset 2015, the majority of super peers involved in several channels support only control traffic. However, in the 2013 dataset several super peers also support video traffic, which is in sharp contrast with what can be observed in the 2015 dataset. This is another indication of the important modification that took place between the two measurement campaigns.

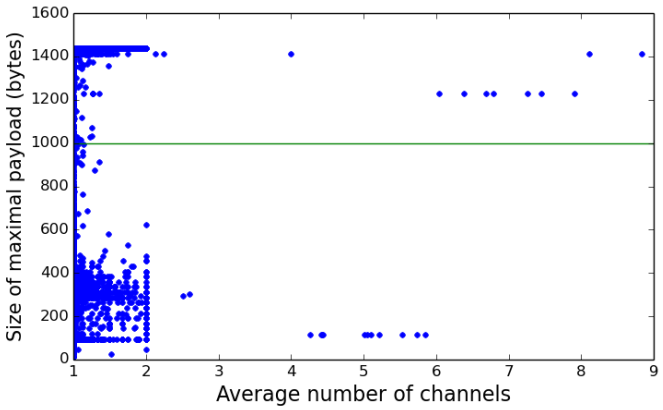



Figure B.7: Average multi-channel presence *v.s.* maximal payload (2013 dataset).

## B.7 Conclusion

In this paper, we used two datasets obtained by two measurement campaigns relying on respectively 10 and 12 points of measure associated to 10 and 12 channels on a P2PTV system. We investigated how much information we can retrieve on multi-channel peers and showed that although the obtained view is partial, such a non-invasive measurement approach yet enables to detect peers present in several channels. Indeed, we were able to detect that respectively 8% and 16% of the total number of peers were present on more than one channels during the measurement campaign. This number drops to 2% and 3% if we restrain to active peers, *i.e.* peers involved in video content traffic. In addition, conducting an analysis based on sliding time windows led to precisely track peers switching from one channel to another one as well as to identify super-peers and to qualify their role in the infrastructure.

These results are interesting to characterize the behavior of peers and extrapolate the behavior of users. This is important for content providers to adapt the video supply, and for network operators to optimize their infrastructure.

These results also allow to envision promising lines of research. Since the measurement approach is light and does not require to have privilege access to the application itself or its infrastructure, we intend to conduct several measurement campaign targeting both different P2PTV sys-

tems and different measurement points to detect more characteristics related to the applications or to the users behavior.

# Bibliography


[wir, ] Wireshark.

[Agarwal, 2007] Agarwal, S. (2007). A case study of large scale p2p video multicast. In *IP Multimedia Subsystem Architecture and Applications, 2007 International Conference on*, pages 1–5.

[Alessandria et al., 2009] Alessandria, E., Gallo, M., Leonardi, E., Mellia, M., and Meo, M. (2009). P2p-tv systems under adverse network conditions: A measurement study. In *INFOCOM 2009, IEEE*, pages 100–108.

[Alsayed and Higham, 2015] Alsayed, A. and Higham, D. J. (2015). Betweenness in time dependent networks. *Chaos, Solitons & Fractals*, 72:35–48.

[Bader et al., 2007] Bader, D. A., Kintali, S., Madduri, K., and Mihail, M. (2007). Approximating betweenness centrality. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 124–137. Springer.

[Batagelj and Praprotnik, 2016] Batagelj, V. and Praprotnik, S. (2016). An algebraic approach to temporal network analysis based on temporal quantities. *Social Network Analysis and Mining*, 6(1):28.

[Bavelas, 1950] Bavelas, A. (1950). Communication patterns in task-oriented groups. *Journal of the Acoustical Society of America*, 22:725 – 730.

[Bergamini et al., 2016] Bergamini, E., Borassi, M., Crescenzi, P., Marino, A., and Meyerhenke, H. (2016).

Computing top-k closeness centrality faster in unweighted graphs. In *2016 Proceedings of the Eighteenth Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 68–80. SIAM.

[Bergamini et al., 2014] Bergamini, E., Meyerhenke, H., and Staudt, C. L. (2014). Approximating betweenness centrality in large evolving networks. In *2015 Proceedings of the Seventeenth Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 133–146. SIAM.

[Bermolen et al., 2011] Bermolen, P., Mellia, M., Meo, M., Rossi, D., and Valenti, S. (2011). Abacus: Accurate behavioral classification of p2p-tv traffic. *Computer Networks*, 55(6):1394 – 1411.

[Bonacich, 1987] Bonacich, P. (1987). Power and centrality: A family of measures. *American Journal of Sociology*, 92(5):1170–1182.

[Borassi et al., 2015] Borassi, M., Crescenzi, P., and Marino, A. (2015). Fast and simple computation of top-k closeness centralities. *CoRR*, abs/1507.01490.

[Bracciale et al., 2014] Bracciale, L., Bonola, M., Loreti, P., Bianchi, G., Amici, R., and Rabuffi, A. (2014). CRAWDAD dataset roma/taxi (v. 2014-07-17). Downloaded from `https://crawdad.org/roma/taxi/20140717`.

[Braha and Bar-Yam, 2008] Braha, D. and Bar-Yam, Y. (2008). Time-dependent complex networks: dynamic centrality, dynamic motifs, and cycles of social interaction. In Gross, T. and Sayama, H., editors, *Adaptive networks: Theory, models and applications*, pages 38–50. Springer.

[Brandes, 2001] Brandes, U. (2001). A faster algorithm for betweenness centrality*. *Journal of mathematical sociology*, 25(2):163–177.

[Brandes and Pich, 2007] Brandes, U. and Pich, C. (2007). Centrality estimation in large networks. *International Journal of Bifurcation and Chaos*, 17(7):2303–2318.

[Cha et al., 2008] Cha, M., Rodriguez, P., Crowcroft, J., Moon, S., and Amatriain, X. (2008). Watching television over an ip network. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, IMC '08, pages 71–84, New York, NY, USA. ACM.

[Chaintreau et al., 2007] Chaintreau, A., Hui, P., Scott, J., Gass, R., Crowcroft, J., and Diot, C. (2007). Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620. (previously published in the Proceedings of IEEE INFOCOM 2006).

[Colman and Charlton, 2016] Colman, E. R. and Charlton, N. (2016). Separating temporal and topological effects in walk-based network centrality. *Physical Review E*, 94(1):012313.

[Daly and Haahr, 2007] Daly, E. M. and Haahr, M. (2007). Social network analysis for routing in disconnected delay-tolerant manets. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pages 32–40. ACM.

[Eppstein and Wang, 2001] Eppstein, D. and Wang, J. (2001). Fast approximation of centrality. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '01, pages 228–229, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

[Estrada, 2013] Estrada, E. (2013). Communicability in temporal networks. *Phys. Rev. E*, 88:042811.

[Estrada and Hatano, 2008] Estrada, E. and Hatano, N. (2008). Communicability in complex networks. *Phys. Rev. E*, 77:036111.

[Everett and Borgatti, 2005] Everett, M. and Borgatti, S. (2005). Ego network betweenness. *Social networks*, 27(1):31–38.

[Fenu and Higham, 2017] Fenu, C. and Higham, D. J. (2017). Block matrix formulations for evolving net-

works. *SIAM Journal on Matrix Analysis and Applications*, 38(2):343–360.

[Flores and Romance, 2018] Flores, J. and Romance, M. (2018). On eigenvector-like centralities for temporal networks: Discrete vs. continuous time scales. *Journal of Computational and Applied Mathematics*, 330:1041–1051.

[Fredman and Tarjan, 1987] Fredman, M. L. and Tarjan, R. E. (1987). Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615.

[Freeman, 1982] Freeman, L. (1982). Centered graphs and the structure of ego networks. *Mathematical Social Sciences*, 3(3):291–304.

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

[Gemmetto et al., 2014] Gemmetto, V., Barrat, A., and Cattuto, C. (2014). Mitigation of infectious disease at school: targeted class closure vs school closure. *BMC infectious diseases*, 14(1):695.

[Ghanem et al., 2017] Ghanem, M., Coriat, F., and Tabourier, L. (2017). Ego-betweenness centrality in link streams. In *The 7th Workshop on Social Network Analysis in Applications (workshop IEEE ASONAM 2017)*, Sydney, Australia.

[Ghanem et al., 2016] Ghanem, M., Fourmaux, O., Tarissan, F., and Miyoshi, T. (2016). P2ptv multi-channel peers analysis. In *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pages 1–6.

[Ghanem et al., 2018a] Ghanem, M., Magnien, C., and Tarissan, F. (2018a). Centrality metrics in dynamic networks: a comparison study (under minor revision). *IEEE Transactions on Network Science and Engineering*, pages 1–1.

[Ghanem et al., 2018b] Ghanem, M., Magnien, C., and Tarissan, F. (2018b). How to exploit structural properties of dynamic networks to detect nodes with high temporal closeness. In *CTW18: Cologne-Twente Workshop on Graphs and Combinatorial Optimization 2018)*, Paris, France.

[Ghosh et al., 2011] Ghosh, R., Kuo, T.-T., Hsu, C.-N., Lin, S.-D., and Lerman, K. (2011). Time-aware ranking in dynamic citation networks. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 373–380. IEEE.

[Ghosh and Lerman, 2012] Ghosh, R. and Lerman, K. (2012). Rethinking centrality: the role of dynamical processes in social network analysis. *arXiv preprint arXiv:1209.4616*.

[Ghosh et al., 2014] Ghosh, R., Lerman, K., Teng, S., and Yan, X. (2014). The interplay between dynamics and networks: Centrality, communities, and cheeger inequality. *CoRR*, abs/1406.3387.

[Girvan and Newman, 2002] Girvan, M. and Newman, M. E. J. (2002). Community structure in social and biological networks. *PNAS*, 99:7821–7826.

[Green et al., 2012] Green, O., McColl, R., and Bader, D. A. (2012). A fast algorithm for streaming betweenness centrality. In *Proceedings of the 2012 ASE/IEEE International Conference on Social Computing and 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust*, SOCIALCOM-PASSAT '12, pages 11–20, Washington, DC, USA. IEEE Computer Society.

[Grindrod et al., 2011] Grindrod, P., Parsons, M. C., Higham, D. J., and Estrada, E. (2011). Communicability across evolving networks. *Physical Review E*, 83(4):046120.

[Hei et al., 2007] Hei, X., Liang, C., Liang, J., Liu, Y., and Ross, K. W. (2007). A measurement study of a large-

scale p2p iptv system. *IEEE Transactions on Multimedia*, 9(8):1672–1687.

[Hei et al., 2008] Hei, X., Liu, Y., and Ross, K. W. (2008). Iptv over p2p streaming networks: the mesh-pull approach. *Communications Magazine, IEEE*, 46(2):86–92.

[Heymann et al., 2012] Heymann, S., Latapy, M., and Magnien, C. (2012). Outskewer: Using Skewness to Spot Outliers in Samples and Time Series. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 527–534. IEEE.

[Holme, 2015] Holme, P. (2015). Modern temporal network theory: a colloquium. *The European Physical Journal B*, 88(9):234.

[Holme and Saramäki, 2012] Holme, P. and Saramäki, J. (2012). Temporal networks. *Physics reports*, 519(3):97–125.

[Hoory et al., 2006] Hoory, S., Linial, N., and Wigderson, A. (2006). Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561.

[Hoßfeld and Leibnitz, 2008] Hoßfeld, T. and Leibnitz, K. (2008). A qualitative measurement survey of popular internet-based iptv systems. In *Communications and Electronics, 2008. ICCE 2008. Second International Conference on*, pages 156–161. IEEE.

[Huang and Yu, 2017] Huang, D.-W. and Yu, Z.-G. (2017). Dynamic-sensitive centrality of nodes in temporal networks. *Scientific Reports*, 7:41454.

[Hui et al., 2011] Hui, P., Crowcroft, J., and Yoneki, E. (2011). Bubble rap: Social-based forwarding in delay-tolerant networks. *Transactions on Mobile Computing*, 10(11):1576–1589.

[Isella et al., 2011] Isella, L., StehlÃĺ, J., Barrat, A., Cattuto, C., Pinton, J., and Van den Broeck, W. (2011). What's in a crowd? analysis of face-to-face behavioral networks. *Journal of Theoretical Biology*, 271(1):166–180.

[Jain et al., 2004] Jain, S., Fall, K., and Patra, R. (2004). *Routing in a delay tolerant network*, volume 34. ACM.

[Jia et al., 2007] Jia, J., Li, C., and Chen, C. (2007). Characterizing ppstream across internet. In *Proceedings of the 2007 IFIP International Conference on Network and Parallel Computing Workshops*, NPC '07, pages 413–418, Washington, DC, USA. IEEE Computer Society.

[Kas et al., 2013] Kas, M., Carley, K. M., and Carley, L. (2013). Incremental closeness centrality for dynamically changing social networks. In *IEEE/ACM International Conference on Advances in Social Netowrks Analysis and Mining (ASONAM)*, pages 1250–1258. IEEE.

[Katz, 1953] Katz, L. (1953). A new status index derived from sociometric index. *Psychometrika*, 18:39–43.

[Kendall, 1938] Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

[Kim et al., 2014] Kim, C.-M., Han, Y.-H., Youn, J.-S., and Jeong, Y.-S. (2014). A socially aware routing based on local contact information in delay-tolerant networks. *The Scientific World Journal*, 2014.

[Kim and Anderson, 2012] Kim, H. and Anderson, R. (2012). Temporal node centrality in complex networks. *Physical Review E*, 85:1–8.

[Kim et al., 2012] Kim, H., Tang, J., Anderson, R., and Mascolo, C. (2012). Centrality prediction in dynamic human contact networks. *Computer Networks*, 56(3):983–996.

[Kostakos, 2009] Kostakos, V. (2009). Temporal graphs. *Physica A: Statistical Mechanics and its Applications*, 388(6):1007–1023.

[Kourtellis et al., 2015] Kourtellis, N., Morales, G. D. F., and Bonchi, F. (2015). Scalable online betweenness centrality in evolving graphs. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2494–2506.

[Krings et al., 2012] Krings, G., Karsai, M., Bernhardsson, S., Blondel, V. D., and Saramäki, J. (2012). Effects of time window size and placement on the structure of an aggregated communication network. *EPJ Data Science*, 1(1):4.

[Kumar et al., 2016] Kumar, S., Spezzano, F., Subrahmanian, V., and Faloutsos, C. (2016). Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 221–230. IEEE.

[Kunegis, 2013] Kunegis, J. (2013). KONECT – The Koblenz Network Collection. In *Proc. Int. Conf. on World Wide Web Companion*, pages 1343–1350.

[Laflin et al., 2013] Laflin, P., Mantzaris, A. V., Ainley, F., Otley, A., Grindrod, P., and Higham, D. J. (2013). Discovering and validating influence in a dynamic online social network. *Social Network Analysis and Mining*, 3(4):1311–1323.

[Latapy et al., 2017] Latapy, M., Viard, T., and Magnien, C. (2017). Stream graphs and link streams for the modeling of interactions over time. *arXiv preprint arXiv:1710.04073*.

[Lee et al., 2016] Lee, M.-J., Choi, S., and Chung, C.-W. (2016). Efficient algorithms for updating betweenness centrality in fully dynamic graphs. *Inf. Sci.*, 326(C):278–296.

[Léo et al., 2015] Léo, Y., Crespelle, C., and Fleury, E. (2015). Non-altering time scales for aggregation of dynamic networks into series of graphs. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, page 29. ACM.

[Lerman et al., 2010] Lerman, K., Ghosh, R., and Kang, J. H. (2010). Centrality metric for dynamic networks. In *Proceedings of the Eighth Workshop on Mining and Learning with Graphs - MLG '10*, pages 70–77, New York, New York, USA. ACM Press.

[Liao et al., 2017] Liao, H., Mariani, M. S., Medo, M., Zhang, Y.-C., and Zhou, M.-Y. (2017). Ranking in evolving complex networks. *Physics Reports*, 689:1–54.

[Lim et al., 2011] Lim, Y.-s., Menasché, D. S., Ribeiro, B., Towsley, D., and Basu, P. (2011). Online estimating the k central nodes of a network. In *Network Science Workshop (NSW), 2011 IEEE*, pages 118–122. IEEE.

[Magaia et al., 2015] Magaia, N., Francisco, A., Pereira, P., and Correia, M. (2015). Betweenness centrality in delay tolerant networks: A survey. *Ad Hoc Networks*, 33:284–305.

[Magharei and Rejaie, 2006] Magharei, N. and Rejaie, R. (2006). Understanding mesh-based peer-to-peer streaming. In *Proceedings of the 2006 International Workshop on Network and Operating Systems Support for Digital Audio and Video*, NOSSDAV '06, pages 10:1–10:6, New York, NY, USA. ACM.

[Magnien and Tarissan, 2015] Magnien, C. and Tarissan, F. (2015). Time evolution of the importance of nodes in dynamic networks. In *Proceedings of the International Symposium on Foundations and Applications of Big Data Analytics (FAB), in conjunction with ASONAM, 2015.*, FAB '15, pages 1200–1207, New York, NY, USA. ACM.

[Maiya and Berger-Wolf, 2010] Maiya, A. S. and Berger-Wolf, T. Y. (2010). Online sampling of high centrality individuals in social networks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 91–98. Springer.

[Mantzaris and Higham, 2013] Mantzaris, A. V. and Higham, D. J. (2013). Dynamic communicability predicts infectiousness. In *Temporal Networks*, pages 283–294. Springer.

[Mendes et al., 2010] Mendes, J., Salvador, P., and Nogueira, A. (2010). P2p-tv service and user characterization. In *Computer and Information Technology*

*(CIT), 2010 IEEE 10th International Conference on*, pages 2612–2620.

[Michalski et al., 2011] Michalski, R., Palus, S., and Kazienko, P. (2011). Matching organizational structure and social network extracted from email communication. In *Lecture Notes in Business Information Processing*, volume 87, pages 197–206. Springer Berlin Heidelberg.

[Mizutani et al., 2015] Mizutani, K., Miyoshi, T., and Fourmaux, O. (2015). *Traffic Analysis in Concurrent Multi-Channel Viewing on P2PTV*. Springer Berlin Heidelberg.

[Nasre et al., 2014] Nasre, M., Pontecorvi, M., and Ramachandran, V. (2014). Betweenness centrality – incremental and faster. In Csuhaj-Varjú, E., Dietzfelbinger, M., and Ésik, Z., editors, *Mathematical Foundations of Computer Science 2014*, pages 577–588, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Newman, 2010] Newman, M. (2010). *Networks: an introduction*. Oxford university press.

[Nicosia et al., 2013] Nicosia, V., Tang, J., Mascolo, C., Musolesi, M., Russo, G., and Latora, V. (2013). Graph metrics for temporal networks. In *Temporal networks*, pages 15–40. Springer.

[Okamoto et al., 2008] Okamoto, K., Chen, W., and Li, X.-Y. (2008). Ranking of closeness centrality for large-scale social networks. In *International Workshop on Frontiers in Algorithmics*, pages 186–195. Springer.

[Opsahl and Panzarasa, 2009] Opsahl, T. and Panzarasa, P. (2009). Clustering in weighted networks. *Social Networks*, 31(2):155–163.

[Page et al., 1999] Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

[Pan and Saramäki, 2011] Pan, R. K. and Saramäki, J. (2011). Path lengths, correlations, and centrality in temporal networks. *Physical Review E*, 84(1):016105.

[Potamias et al., 2009] Potamias, M., Bonchi, F., Castillo, C., and Gionis, A. (2009). Fast shortest path distance estimation in large networks. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 867–876. ACM.

[Prado et al., 2016] Prado, S. D., Dahmen, S. R., Bazzan, A. L. C., Mac Carron, P., and Kenna, R. (2016). Temporal Network Analysis of Literary Texts. *ArXiv e-prints*.

[Praprotnik and Batagelj, 2015] Praprotnik, S. and Batagelj, V. (2015). Spectral centrality measures in temporal networks. *ARS MATHEMATICA CONTEMPORANEA*, 11(1):11–33.

[Qiao et al., 2017] Qiao, L., Shi, Y., and Chen, S. (2017). An empirical study on the temporal structural characteristics of vanets on a taxi gps dataset. *IEEE Access*, 5:722–731.

[Riondato and Upfal, 2016] Riondato, M. and Upfal, E. (2016). Abra: Approximating betweenness centrality in static and dynamic graphs with rademacher averages. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1145–1154. ACM.

[Rossi et al., 2011] Rossi, D., Sottile, E., and Veglia, P. (2011). Black-box analysis of internet p2p applications. *Peer-to-Peer Networking and Applications*, 4:146–164.

[Rozenshtein and Gionis, 2016] Rozenshtein, P. and Gionis, A. (2016). Temporal pagerank. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 674–689. Springer.

[Santos et al., 2016] Santos, E. E., Korah, J., Murugappan, V., and Subramanian, S. (2016). Efficient anytime anywhere algorithms for closeness centrality in large and

dynamic graphs. In *Parallel and Distributed Processing Symposium Workshops, 2016 IEEE International*, pages 1821–1830. IEEE.

[Saxena et al., 2017] Saxena, A., Gera, R., and Iyengar, S. (2017). A faster method to estimate closeness centrality ranking. *arXiv preprint arXiv:1706.02083*.

[Scholtes et al., 2016] Scholtes, I., Wider, N., and Garas, A. (2016). Higher-order aggregate networks in the analysis of temporal networks: path structures and centralities. *The European Physical Journal B*, 89(3):61.

[Scholtes et al., 2014] Scholtes, I., Wider, N., Pfitzner, R., Garas, A., Tessone, C. J., and Schweitzer, F. (2014). Causality-driven slow-down and speed-up of diffusion in non-markovian temporal networks. *Nature communications*, 5:5024.

[Shamma et al., 2009] Shamma, D. A., Kennedy, L., and Churchill, E. F. (2009). Tweet the debates: Understanding community annotation of uncollected sources. In *Proceedings of the First SIGMM Workshop on Social Media*, WSM '09, pages 3–10, New York, NY, USA. ACM.

[Shetty and Adibi, 2005] Shetty, J. and Adibi, J. (2005). Discovering important nodes through graph entropy the case of Enron email database. In *Proceedings of the 3rd international workshop on Link discovery - LinkKDD '05*, pages 74–81, New York, New York, USA. ACM Press.

[Silva et al., 2017] Silva, A., Singh, A., and Swami, A. (2017). Spectral algorithms for temporal graph cuts. *arXiv preprint arXiv:1702.04746*.

[Silverston et al., 2009] Silverston, T., Fourmaux, O., Botta, A., Dainotti, A., Pescapé, A., Ventre, G., and Salamatian, K. (2009). Traffic analysis of peer-to-peer iptv communities. *The International Journal of Computer and Telecommunications Networking*, 53(4):470–484.

[Singh et al., 2015] Singh, R. R., Goel, K., Iyengar, S., and Gupta, S. (2015). A faster algorithm to update between-

ness centrality after node alteration. *Internet Mathematics*, 11(4-5):403–420.

[Spoto et al., 2009] Spoto, S., Gaeta, R., Grangetto, M., and Sereno, M. (2009). Analysis of pplive through active and passive measurements. In *Parallel & Distributed*.

[Takaguchi et al., 2016] Takaguchi, T., Yano, Y., and Yoshida, Y. (2016). Coverage centralities for temporal networks. *The European Physical Journal B*, 89(2):35.

[Tang et al., 2010] Tang, J., Musolesi, M., Mascolo, C., Latora, V., and Nicosia, V. (2010). Analysing information flows and key mediators through temporal centrality metrics. In *Proceedings of the 3rd Workshop on Social Network Systems*, pages 1–6. ACM.

[Tang et al., 2009] Tang, S., Lu, Y., Hernández, J. M., Kuipers, F., and Van Mieghem, P. (2009). Topology dynamics in a p2ptv network. In *NETWORKING 2009*, pages 326–337. Springer.

[Taylor et al., 2017] Taylor, D., Myers, S. A., Clauset, A., Porter, M. A., and Mucha, P. J. (2017). Eigenvector-based centrality measures for temporal networks. *Multiscale Modeling & Simulation*, 15(1):537–574.

[Tournoux et al., 2009] Tournoux, P. U., Leguay, J., Dias de Amorim, M., Benbadis, F., Conan, V., and Whitbeck, J. (2009). The Accordion Phenomenon: Analysis, Characterization, and Impact on DTN Routing. In *Proceedings of the 28rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1116–1124. IEEE.

[Tretyakov et al., 2011] Tretyakov, K., Armas-Cervantes, A., García-Bañuelos, L., Vilo, J., and Dumas, M. (2011). Fast fully dynamic landmark-based estimation of shortest path distances in very large graphs. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1785–1794. ACM.

[Uddin et al., 2016] Uddin, S., Khan, A., and Piraveenan, M. (2016). A set of measures to quantify the dynamicity of longitudinal social networks. *Complexity*, 21(6):309–320.

[Uddin et al., 2013] Uddin, S., Piraveenan, M., Chung, K. S. K., and Hossain, L. (2013). Topological analysis of longitudinal networks. In *2013 46th Hawaii International Conference on System Sciences*, pages 3931–3940.

[Valenti et al., 2009] Valenti, S., Rossi, D., Meo, M., Mellia, M., and Bermolen, P. (2009). Accurate, fine-grained classification of p2p-tv applications by simply counting packets. In *Traffic Monitoring and Analysis*, pages 84–92. Springer.

[Viswanath et al., 2009] Viswanath, B., Mislove, A., Cha, M., and Gummadi, K. P. (2009). On the evolution of user interaction in Facebook. In *Proc. Workshop on Online Social Networks*, pages 37–42.

[Wang et al., 2013] Wang, M., Fourmaux, O., Nakamura, Y., and Miyoshi, T. (2013). Network impact of p2p-tv zapping. IEEE/ACIS-SNDP.

[Wang et al., 2017] Wang, Z., Pei, X., Wang, Y., and Yao, Y. (2017). Ranking the key nodes with temporal degree deviation centrality on complex networks. In *2017 29th Chinese Control And Decision Conference (CCDC)*, pages 1484–1489.

[Williams and Musolesi, 2016] Williams, M. J. and Musolesi, M. (2016). Spatio-temporal networks: reachability, centrality and robustness. *Royal Society open science*, 3(6):160196.

[Wilmet et al., 2018] Wilmet, A., Viard, T., Latapy, M., and Lamarche-Perrin, R. (2018). Degree-based outliers detection within ip traffic modelled as a link stream. In *Network Traffic Measurement and Analysis Conference (TMA), 2018*. IEEE.

[Wu et al., 2009] Wu, D., Liu, Y., and Ross, K. (2009). Queuing network models for multi-channel p2p live streaming systems. In *INFOCOM 2009, IEEE*, pages 73–81.

[Zhu et al., 2013] Zhu, Y., Xu, B., Shi, X., and Wang, Y. (2013). A survey of social-based routing in delay tolerant networks: Positive and negative social effects. *IEEE Communications Surveys & Tutorials*, 15(1):387–401.