PROPER TIME WINDOW SIZE SELECTION FOR TIMELY EVOLVING DYNAMIC NETWORKS

(ZAMANLA DEĞİŞEN DİNAMİK AĞLAR İÇİN UYGUN ZAMAN PENCERESİ BOYU SEÇİMİ)

by

Serhat Çolak, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

in the

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

of

GALATASARAY UNIVERSITY

Supervisor: Assist. Prof. Keziban Orman

June 2021

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my thesis supervisor Assist. Prof. Keziban Orman for her unprecedented support and valuable guidance.

I would also like to give my special thanks to my family for providing me tenacity to complete this thesis.

At last but not least gratitude goes to my friends, Mahir Kutay Sivri, M.D. and Oğuz Kuzu, for critical readings they performed.

Table of Contents

List of Figures

List of Tables

ABSTRACT

Proper dynamic network extraction is a prominent problem for timely evolving systems' modelling. Dynamic network modelling of timely evolving complex systems allow to discover emerging properties of real-world facts. The main issue of such modelling is determining the proper time intervals for each member of network. The proper window size should be detected for extracting the most informative and least noisy time series of snapshot features. Existing solutions suffer from using only network topological properties as snapshot features, applying subjective methodologies needed user-dependent data labeling, and extracting snapshots with equal sized windows. We propose not only a new network similarity based compression ratio measuring the properness of studied window size, but also a window aggregation strategy allowing to extract a more informative dynamic network with less noisier structure by using the similarity metric and its statistical significance. The results on Enron, Haggle Infocom and Reality Mining data sets reveal that the proposed compression ratio is more effective for finding best window size than baseline, and aggregation strategy allow to capture important dynamic events which might be hidden in noise when using constant windows. Moreover, the statistical results show that the general topological characteristics of the network is mostly not affected after aggregation process. Finally, as a complementary result, aggregation process can be used in determining proper time interval for modelling.

ÖZET

Zamanla değişen karmaşık sistemler hemen hemen her alanda ortaya çıkmaktadır; bir lisenin dinamik arkadaşlık ortamda, şirket çalışanları arasındaki e-postalarda, bir ortamdaki elektronik cihazların sensör bağlantılarında vs. Örnekler ve uygulamalar çoğaltılabilir. Zamanla değişen karmaşık sistemlerin ortaya çıkan ve doğrusal olmayan dinamiklerini keşfetmek için bu sistemleri modellemek önemli bir konudur. Bu tür keşifler için, dinamik ağlar öne çıkan araçlardır (Holme & Saramäki, 2012). Çeşitli alanlardan birçok bilim insanı, dinamik ağ modellemesiyle ilgilenmişlerdir (Spiliopoulou, 2011; Blonder et al., 2012; Aggarwal & Subbian, 2014; Leskovec et al., 2007; Kempe et al., 2000; Rossi et al., 2013; Berger-Wolf & Saia, 2006). Olay listeleri, bağlantı akışları veya olay tabanlı sıralı graflar gibi birçok farklı dinamik ağ modeli tipi olmasına rağmen bu çalışmanın ana odağı zamana bağlı sıralı graflardır. Bir dinamik ağ, bir sistemin içerdiği bilgiyi ne kadar gürültüsüz ve temiz bir şekilde ifade ederse, bu ağ, sistemin analizinde o kadar faydalı olacaktır. Bilgilendirici bir model çıkarımının kilit noktası, sıralı graflar için uygun zaman aralığını belirlemektir. Bu durum, literatürde önemli bir konu olarak karşımıza çıkmaktadır (Rocha et al., 2017; Medo et al., 2018; Dakiche et al., 2018; Fish & Caceres, 2017). Şeçilen zaman aralığına bağlı olarak, modelin topolojisi değişir ve bu da topluluk tespiti (Medo et al., 2018; Dakiche et al., 2018), bağlantı tahminleme, özellik tahminleme ve değişim noktası tespiti (Fish & Caceres, 2017) gibi diğer tüm analizleri etkiler.

Biz bu çalışmada, etkili bir dinamik ağ modellemesi çıkarımı için uygun zaman aralığı bulmaya odaklanıyoruz. Uygun zaman aralıklarının özellikleri, ne ağı gürültülü bir hale getirecek kadar kısa ne de ağdaki zamana bağlı değişimleri gözardı edecek kadar uzun olmasıdır (Soundarajan et al., 2016; Sulo et al., 2010). Uygun zaman aralıklarını bulmanın yaygın deneysel yolu, en iyisini seçmek için aday pencere boyları ile çıkarılan farklı dinamik ağların kalitesini belirlemektir (Sulo et al., 2010; Uddin et al., 2017; Krings et al., 2012; K. Darst et al., 2016). İlk olarak, aday pencere boyları belirlenir ve her bir farklı pencere boyu için pencere kaydırma

yöntemi ile aday dinamik ağ oluşturulur. İkinci olarak, her bir aday ağ, çap, ortalama mesafe gibi (Sulo et al., 2010) çeşitli topolojik özelliklerin veya benzerlik metrikleri gibi akış özelliklerinin bir zaman serisi olarak ifade edilir. Ağ dizisinin kalitesi, ağ özelliklerinin zaman serisinin kalitesiyle ölçülür. Son olarak, en iyi zaman serisini ortaya çıkaran pencere boyu, uygun zaman aralığı olarak seçilir. Üretilen bir zaman serisinin sağlamlığı, genellikle zaman serilerinin varyansını ve sıkıştırma oranını istatistiksel olarak ölçerek (Sulo et al., 2010) veya denetimsiz (Uddin et al., 2017) veya denetimli (Fish & Caceres, 2017) öğrenmeyle bulunur. Sonuç olarak, en uygun dinamik ağ, en nitelikli özellik zaman serisini ortaya çıkaran ağdır. Önceki çalışmalarda bu ağdaki her bir üye eş zaman aralığı kullanılarak oluşturulmuştur.

Mevcut yaklaşımların temel sorunları, hem ağ dizisinin temsil edilmesi ile hem de kaliteyi belirleme yöntemleri ile ilgilidir. Bu yaklaşımlarda, ya kullanıcıya bağlı etiketlemeye başvurulmuştur (Fish & Caceres, 2017) ya da ölçekten bağımsız olmayan metrikler kullanılmıştır. (Sulo et al., 2010). Biz bu çalışmada, ikinci soruna bir çözüm öneriyoruz. İlgili çalışmada (Sulo et al., 2010), ağ dizisi, ağa bağlı topolojik özelliklerin bir zaman serisi olarak temsil edilmiştir. Ayrıca, zaman serilerinin gürültü ve sıkıştırma oranları, sırasıyla ölçekten bağımsız olmayan bir istatistik olan varyans ve karakter dizisi sıkıştırmaları için geliştirilmiş olan run-length kodlaması tabanlı metrik ile ölçülmüştür. Bu yaklaşımdaki ölçümlerin çoğu, objektif sınırlara veya ölçeklere sahip olmadığı için yanıltıcı olabilir.

İlk katkımız, ölçekten bağımsız olmayan metriklerin (Sulo et al., 2010) kullanımıyla ortaya çıkan problemin üstesinden gelmek adına yeni bir sıkıştırma oranı önermektir. İlk olarak, ağ dizisini, zaman geçtikçe sistemin kararlılığını yansıtan ölçekten bağımsız benzerlik ölçütleri ile temsil etmeyi ve ikinci olaraksa, sıkıştırma oranını karakter serisi temelli ölçümlerle değil, benzerlik ölçütlerinin istatistiksel olarak önemini ölçerek yapmayı öneriyoruz. Böylelikle, en uygun zaman aralığını objektif bir şekilde belirleyebiliriz. İkinci katkımız, sıralı üyeleri farklı sürelere sahip olabilecek daha bilgilendirici ve kompakt dinamik ağ oluşturabilen sıralı anlık görüntülerin bağlantı yapılarının istatistiksel önemine dayalı bir zaman aralığı toplama stratejisi önermektir. Yakın zamanda, uygun zaman aralığını bulabilmek adına ölçekten bağımsız ağ topolojisi benzerlik metriği tabanlı Jaccard benzerliğini kullanmanın etkinliği gösterilmiştir (Orman et al., 2021). Toplama aşamasında, eğer yeni anlık görüntü ile toplama aşamasında olan ilk anlık görüntünün benzerliği, beklediğimiz istatistiksel anlamlılık değerinden düşükse bu anlık görüntüleri

birleştiriyoruz çünkü istatistiksel olarak bu iki görüntü, kendisini tekrar eden topolojik bilgiye sahip olabilir. Üçüncü katkımız, bu stratejiyi kullanarak uygun zaman aralıklarının tespitidir. Halihazırda sunmuş olduğumuz sıkıştırma oranı ile uygun zaman aralığını belirleyebilmekteydik. Buna ek olarak, toplama stratejisinin de en iyi zaman aralığını seçebilmek konusunda yardımcı olabileceğini gösteriyoruz. Son katkımız, bu çalışmanın sunduklarını, daha önce de üzerine çalışılmış olan ve iyi bilenen üç veri seti üzerinde doğrulamaktır: Enron Email (Clauset & Eagle, 2012), Haggle Infocomm (Sulo et al., 2010) ve MIT Reality Mining (Sulo et al., 2010). Ayrıca, uygun pencere boyunu ve sonuçların sinyal kalitesini bulabilmek adına sabit pencere boyu kullanımıyla pencere toplama stratejisinin karşılaştırmalı bir analizini yapıyoruz.

Bu çalışmayı genel hatlarıyla anlatmak gerekirse; bölüm 1, dinamik ağlarda uygun zaman seçimi konusunda daha önce önerilmiş olan çözümleri ve bu çözümlerdeki problemlerin neler olduğunu anlattığımız bölümdür. Ayrıca bu çalışmanın da motivasyonunu oluşturan bu problemlere dair çözümlerimizden de ilk olarak bahsettiğimiz kısımdır. Bölüm 2, bu alandaki çalışmalardan bahsettiğimiz literatür araştırma bölümüdür. Bu bölümde, hem bizim çalışmamızla ilgili olan hem de genel hatlarıyla farklı ağ kıyaslamaları üzerine olan çalışmalar incelenmektedir. Bölüm 3, dinamik ağ çıkarımı için temel yöntemi, önerdiğimiz sıkıştırma oranını ve pencere biriktirmek için sunduğumuz toplama stratejisini açıkladığımız bölümdür. Öncelikle, çalışma boyunca kullandığımız kavramların tanımlarını veriyoruz. Ardından, en iyi pencere boyunu bulmak için kullanılan önceki yöntemlerin problemlerini açıklıyoruz. Sonrasında, bu problemin üstesinden gelmek için yeni bir sıkıştırma oranı öneriyoruz. Son olarak da, daha bilgilendirici ve kompakt dinamik ağlar oluşturan bir toplama algoritması öneriyoruz. Bölüm 4, deneylerimizde kullandığımız veri setlerini tanıttığımız ve bu veri setlerinin üzerinde çalışabilmek adına yaptığımız ön çalışmaları detaylı bir şekilde anlattığımız bölümdür. Son olarak da, toplama stratejimiz ile ve toplama stratejisi olmadan üretilmiş olan anlık görüntüleri karşılaştırmada kullanılacak olan benzerlik metriklerini tanımlıyoruz. Bölüm 5, yeni sıkıştırma oranı ve toplama stratejisinin etkinliğine de değinerek, iyi bilinen üç veri seti -Enron Email, Haggle Infocom ve MIT Reality Mining- üzerindeki deneylerimizin sonuçlarını ve yorumlarımızı içeren bölümdür. İlk olarak, hem toplama yöntemiyle hem de toplama yöntemi olmadan oluşturulan anlık görüntülerin topolojik özelliklerinin sonuçlarını veriyor ve yorumlarımızı yapıyoruz. Sonrasında, (Sulo et al., 2010) tarafından önerilmiş olan TWIN yönteminin sonuçları

ile önerdiğimiz bağlantı ve düğüm temelli sıkıştırma oranı metriğinin karşılaştırmalı sonuçlarını veriyoruz. Ayrıca, toplama stratejisinin de en iyi pencere boyunu bulmak için yardımcı olup olamayacağına değiniyoruz. Son olarak, önermiş olduğumuz toplama stratejisinin etkinliğini gösterebilmek adına, benzerlik metriklerinin sinyal sonuçlarını da detaylı bir şekilde yorumluyoruz. Son kısım olan bölüm 6 ile çalışmayı özetliyor, son sözlerimizi söylüyor ve çalışmanın gelecekteki yönlerini açıklıyoruz.

1 INTRODUCTION

Timely evolving complex systems are emerging in almost every domain; a dynamic environment of social friendship in a high school, emails among a company members, the connections of electronic devices via sensors in an area etc. The examples and applications are numerous. Modelling timely evolving complex systems for discovering their emergent and non linear dynamics is an important issue. Dynamic networks are the prominent tools for such discovery (Holme & Saramäki, 2012). Many scientists from various fields have become interested in dynamic network modelling (Spiliopoulou, 2011; Blonder et al., 2012; Aggarwal & Subbian, 2014; Leskovec et al., 2007; Kempe et al., 2000; Rossi et al., 2013; Berger-Wolf & Saia, 2006). Although there are several different types of dynamic network models such as event lists, link streams, or event-based sequential graphs, the main interest of this work is a time-based sequential graph representation. The more noiseless and clearer the dynamic network expresses the information contained by the system, the more useful it will be in its analysis. The key point of extracting an informative model is determining proper time intervals for sequence members. It arises as an important issue in the literature (Rocha et al., 2017; Medo et al., 2018; Dakiche et al., 2018; Fish & Caceres, 2017). Depending on the chosen time intervals, the topology of the model changes and it affects all further analyzes such as community detection (Medo et al., 2018; Dakiche et al., 2018), link prediction, attribute prediction, and change-point detection (Fish & Caceres, 2017).

In this work, we focus on finding proper time interval for extracting an effective dynamic network modelling. The properties of proper time intervals are reported as being neither small enough to make the network noisy nor large enough to ignore significant time-dependent effects on the network (Soundarajan et al., 2016; Sulo et al., 2010). The common empirical way of finding proper time intervals is to determine the quality of different dynamic networks which are extracted by candidate window sizes for choosing the best one (Sulo et al., 2010; Uddin et al.,

2017; Krings et al., 2012; K. Darst et al., 2016). First, different candidate dynamic networks are extracted for each window by sliding it. Second, each candidate network is expressed as a time series of various topological properties, i.e. diameter, average distance etc. (Sulo et al., 2010), or flow features, i.e. snapshot similarity metrics. The quality of the network sequence is measured by the quality of time series of network features. Finally, the window size which results the best time series is chosen as the proper time interval. The robustness of a generated time series is usually stated by measuring statistically the variance and compression ratio of the time series (Sulo et al., 2010), or unsupervised (Uddin et al., 2017) or supervised learning (Fish & Caceres, 2017). As a result, the most proper dynamic network is the one whose members have all the same time interval which is actually the best window size that results the most qualified feature time series.

The main problems of existing approaches are both about the representation of network sequence and the methods of determining the quality. They either consult user-dependent labelling (Fish & Caceres, 2017) or use metrics which are not scale invariant (Sulo et al., 2010). In this work, we propose a solution for the latter problem. In (Sulo et al., 2010), the network sequence is represented as the time series of network-dependent topological properties. Moreover, the noise and compression ratio of the time series are measured by variance, which is not a scale invariant statistic, and run-length encoding based metric, which is developed for string compression, respectively. Most of the measurements in this approach can be misleading because of not having objective limits or scales.

Our first contribution is proposing a compression ratio to overcome the problem of using metrics which are not scale invariant (Sulo et al., 2010). We propose first to represent the network sequence by scale invariant similarity metrics which reflect the stability of the system as time goes by and second to measure the compression ratio not string based measures but statistically significance of similarity metrics. Thus, we determine the most proper window size more objectively. Our second contribution is proposing a window aggregation strategy based on the statistical significance of link structure similarities of consecutive snapshots that generates a more informative and compact dynamic network with the sequence whose members might have different duration. Recently, the effectiveness of the usage of scale invariant network topological similarity metrics based on Jaccard similarity for finding proper time intervals is shown in (Orman et al., 2021). We aggregate the

windows if the similarity of the new snapshot with the first snapshot of aggregation stage is lower than the expected statistical significant value because they might have repeated topological information statistically. Our third contribution is using this strategy in order to determine proper time window size. With the compression ratio we propose, we already determine proper time window size. In addition to this, we show that aggregation process can help to choose the best window size. Our last contribution is validating the proposal with the experiments on three well-known datasets: Enron (Clauset & Eagle, 2012), Haggle Infocomm (Sulo et al., 2010) and MIT Reality Mining (Sulo et al., 2010). We also perform a comparative analysis of window aggregation strategy with the usage of constant window size in terms of finding proper window size and quality of signal of results.

In section 3, we explain the baseline method for dynamic network extraction, our proposed compression ratio, and the aggregation strategy for accumulating the windows. We give proper definitions of the notions we use throughout this work. Then, we explain the problems of the previous methods in finding the best window size. After that, we propose our new compression ratio in order to overcome these problems. Lastly, we propose an aggregation algorithm which generates more informative and compact dynamic network. We explain the aggregation process, and our algorithm in details. In section 4, we introduce the datasets we use in our experiments, then we explain the proprocessing steps we apply to prepare the datasets we use in detail. Finally, we define the similarity metrics we use to compare snapshots generated with the aggregation method we propose to non-aggregated snapshots. In section 5, we give the results and our interpretations of the experiments on three well-known data sets: Enron Email, Haggle Infocom and Reality Mining by touching the effectiveness of new compression ratio and the usability of aggregation strategy. First, we give the results of the topological properties for both aggregated and non-aggregated snapshots. Following that, we give our interpretations of these results. Next, we compare the results of our link and node compression ratio with the results of the method proposed by (Sulo et al., 2010), i.e., TWIN. Moreover, we touch upon whether aggregation strategy itself can help to find the best window size. Lastly, we interpret the signal results of the similarity metrics to find out the effectiveness of the aggregation strategy we propose in details. Finally, we summarize the work, give our final remarks and explain future aspects in section 6.

2 LITERATURE REVIEW

Effective dynamic network modelling relies on selecting the most appropriate time intervals. When extracting a network, the time interval should be selected in a way that the chosen time interval must not be too large to miss significant time-dependent effects out on the network, and too small to generate noisy network (Fish & Caceres, 2015; Soundarajan et al., 2016; Sulo et al., 2010). In order to find proper time interval, a variety of solutions have been proposed. However, these solutions are either application specific, or data set specific. Usually, the process of determining proper time intervals for dynamic network extraction has been done in an empirical way. Initially, the authors specify a set of window sizes in order to find proper time interval among these window sizes. The process is as follows; the networks are extracted while sliding the windows for each size of them. Following that, the networks are expressed as time series generated from the features of its snapshots. Finally, the most robust time series is selected depending on the analysis, since the most appropriate time interval is the one that generated the most robust time series. There are two key points of this method; extracting a time series with appropriate snapshots' feature, and analyzing these time series to find the most robust one.

The topological properties are widely consulted for analyzing the extracted networks. In (Sulo et al., 2010), the authors make use of a variety of popular topological properties; diameter, density, radius etc. These topological properties are used to form time series in order to be analyzed by their compression ratio, and variance. They point out that there is a trade-off between compression ratio, and variance. As the window size increases so does the compression ratio, but variance decreases. Therefore, as the network becomes less noisy, the system information gets lost. Hence, the problem becomes a minimization problem. The authors also propose another work in which they represent the appropriate window size as a linear function of network links showing a steady behavior (Caceres et al., 2011). After that, in

(Uddin et al., 2017), the authors argue that network-level topological properties such as diameter, average path length etc. can be misleading. Any two snapshots of a dynamic network can be similar in terms of topological properties despite the fact that the roles of nodes are different in these snapshots. Therefore, in order to reflect the dynamicity of nodes, they suggest the usage of three node-level centrality measures; degree, closeness, and betweenness. They also use ARIMA and unsupervised learning with k-mean algorithm for time series analysis. The results of the works we review so far produce one window size. However, a single time interval might not reflect the dynamicity of a timely evolving complex system. For example, the authors propose a methodology called ADAGE (Soundarajan et al., 2016). ADAGE takes the node links as input, and produces a variety of window sizes. Therefore, this methodology, apart from the works we mentioned up to now, can generate different time intervals for different parts of the dynamic networks. Lastly, in (Fish & Caceres, 2017), the authors use supervised learning algorithms for the same problem. Their methodology is generally yields better results; however, it relies upon manual labelling, and the quality of the learning is determined by the training set.

In order to detect dynamic network changes in terms of structure, we can consult to topological properties. Nevertheless, if we only take the link changes, i.e., if the link is present or not, into consideration, we might miss some of the important changes in the system. Furthermore, a vast majority of the topological properties are not normalized. Thus, these topological properties cannot be utilized to get an objective perspective. There are studies that takes similarities of consecutive snapshots into account when it comes to extraction of time series. In (Clauset & Eagle, 2012), the authors create a metric called adjacency correlation coefficient to measure neighborhood similarity of consecutive snapshots' nodes. The adjacency correlation coefficient measures the correlation of adjacency matrix of two networks. They use both adjacency correlation coefficient and topological properties to analyze time series. Moreover, the authors of (K. Darst et al., 2016) remark the the usage of similarities for detecting time intervals. They propose a parameter-free methodology using Jaccard similarity of the events between snapshots by looking peak values, but they do not measure snapshots' similarity. On the other hand, in (Krings et al., 2012), the authors propose another approach that takes topological network properties and snapshots similarities based on Jaccard similarity into account. Nonetheless, instead of generating discrete snapshots, they aggregate them in the

process of window sliding. The result of this process generates a single network. They make use of link similarity, whereas we focus on node, link, neighborhood similarities as well as adjacency correlation coefficient.

Lastly, even though it is not directly related to our work, we would like to talk about a study. (Wills & Meyer, 2019) proposed a review regarding to comprehensive empirical performance evaluation for network comparison metrics. This study compares numerous different metrics based upon the adjacency matrix or spectral distance. Adjacency spectral distance is the most reliable metric for comparing two networks according to authors. However, in order to use adjacency spectral distance to compare two networks, the node number of these two networks must be equal. On the other hand, in a dynamic system, any two consecutive snapshots do not have to have the same node number. New members can join the system as nodes, or existing members might leave the system. Therefore, in our work, we consult topological similarity metrics based upon Jaccard similarity of nodes and links. Moreover, we use neighborhood of nodes, and adjacency correlation coefficient as well.

3 AGGREGATING SNAPSHOT ALGORITHM

3.1 Preliminaries

A timely evolving complex system $\mathcal{C}$ is a system of interacting objects in which each interaction occurs in a time point of the continuous time interval. Let us denote that system is defined between beginning and ending time points $[t_1, t_\theta]$. A dynamic complex network $\mathcal{G} = \langle G_1, \ldots, G_\theta \rangle$ is a representation of $\mathcal{C}$ for discovering its complex dynamics. It is a consecutive set of static networks ordered historically. Each member of this set, G_i $(1 \leq i \leq \theta)$, is called a snapshot where i represents a sub-interval in $[t_1, t_\theta]$. More clearly, $G_i = (V_i, L_i)$ is a static and plain network in which V_i defines its set of nodes and $L_i \subseteq V_i \times V_i$ defines its set of links. Our main aim is extracting an informative and proper $\mathcal{G}$ for modelling $\mathcal{C}$. As in reality $\mathcal{C}$ is defined in continuous interval, we should discrete $[t_1, t_\theta]$ for extracting $\mathcal{G}$ for fitting its definition. This problem can be divided into two sub-problems; first, we should determine the best window size w and second extract proper $\mathcal{G}$ for the best w. We concentrate on these two problems.

3.2 Evaluation of Window Size Selection

Up to now, extracting proper dynamic network problem is seen as only finding the best window size, w. The authors extract the proper dynamic network with the snapshots having equal duration. The problem of finding proper w is tackled as the problem of measuring the quality of candidate networks (Sulo et al., 2010; Uddin et al., 2017; Caceres et al., 2011) that are extracted for different candidate w. Usually, for such measurement, the dynamic network is represented under the form of its features' time series in those works. Indeed, two most critical points of these approaches are *choosing features* and *measuring the quality of time series* respectively. For the first point, the most common features are network topological

properties such as diameter, average distance etc. For the second one, different time series analysis techniques such as discrete Fourier transformation (Eagle & , Sandy), ARIMA (Uddin et al., 2017) or measuring quality by statistics (Sulo et al., 2010) is applied. Among them, the algorithm TWIN which is proposed in (Sulo et al., 2010) is one of the best solutions and it usually becomes the baseline for comparing the performance of other approaches. Our method is an extension of TWIN. That is why; we first explain TWIN and then we detail our proposed method for finding best w in the following part.

TWIN relies on the cross usage of two statistics for time series; *noise* and *compression*. For a given w and the corresponding $\mathcal{G}$, $F_w = \langle F_1, \ldots, F_t \rangle$ is time series of a feature. The noise of F_w is measured by its variance (3.1). It gives information about the change of the F_w over time. The larger the σ^2 the noisier the signal of studied topological feature.

$$\sigma^2 = \frac{1}{t} \sum_{i}^{t} [F_i - \mu(F_w)]^2 \tag{3.1}$$

The compression ratio of F_w is defined as the string compression by run-length encoding. It is a basic compression algorithm that represents sequentially repetitive characters with a single character with the repetition count. Let us say u is the length of string representation of F_w. Each value represents a single character in u; therefore, the repetitive values can be compressed as if they are characters of a string. The length of the compressed string representation of F_w is defined as c. Therefore, compression ratio can be calculated with the formula given in equation 3.2.

$$s = \frac{u}{c} \tag{3.2}$$

If the consecutive snapshots take steadily same value for studied topological property, the compression ratio becomes large. Sulo et al. indicates that σ^2 and s have opposite behaviours (Sulo et al., 2010). Thus, TWIN defines the problem of finding best w as a minimization problem. It analyzes σ^2 and s as functions of w. The best w is the one that σ^2 and s are close or equal to each other. There are two

important issues at this approach; first, network topological properties are not scale invariant. That is why, their usage for comparison can be misleading. Recently, the usage of network similarity metrics' is proven to be effective for this issue (Orman et al., 2021). At this work, Orman et al. shows the effectiveness of the usage of Jaccard similarity based graph comparison metrics instead of topological properties. Second, TWIN applies string compression for numerical data. Run-length encoding is only checking if consecutive members of time series is same or not. Although it can be applied for integers, it is meaningless for real values. Most of the topological properties, i.e. density, average degree etc., takes real values. That is why; one can use only limited number of topological properties such as diameter or radius in TWIN.

For measuring the diversity of F_w, we concentrate on consecutive repetition in the series as TWIN did for string compression. If similarity scores are high and also exhibit a long-lasting repetition, it means the network has redundant snapshots. Contrarily, non-repeating signals exhibit diversity of the snapshots. We use a measure, *String Diversity*, which is based on string compression by run-length encoding to represent this diversity. String diversity is the reciprocal of string compression and can be calculated with the formula given in (3.3).

$$d = \frac{c}{u} \tag{3.3}$$

String diversity suggests that more consecutive repetition in the string representation yields a smaller ratio so, smarter snapshot extraction should yield a larger value of d in comparison to the snapshots extracted with the constant w. We use string diversity to show that the aggregation strategy we propose, which we will explain in next sections, can also be used for selecting proper time window size.

3.2.1 Link and Node Compression Ratio

In this work, we propose a new compression ratio specifically developed for dynamic networks for overcoming mentioned issues. The compression ratio is based on scale invariant similarity metrics. There are several different network similarity metrics

in the literature (Donnat & Holmes, 2018). Nevertheless, most of these metrics are dedicated to measure the similarity of two same sized networks because they use adjacency matrices. If the compared snapshots have different numbers of nodes, which is the most common case, their union networks matrices can be quite sparse. Thus, they suffer from the sparsity of the matrix. That is why we concentrate on Jaccard Similarity based metrics as it is explained in (Orman et al., 2021). In its general description, *Jaccard Similarity* quantifies the number of common parts of two different sets. We use Jaccard Similarity on link and node sets. We call this special form of Jaccard as *Link Similarity*, J_{link} (3.4) and *Node Similarity*, J_{node} (3.5). They measure the amount of repeating links or nodes of two consecutive snapshots to the amount of links or nodes that are seen at least in one of the consecutive snapshots, respectively.

$$J_{link}(t_i, t_{i+1}) = \frac{\mid L_{t_i} \bigcap L_{t_{i+1}} \mid}{\mid L_{t_i} \bigcup L_{t_{i+1}} \mid} \tag{3.4}$$

$$J_{node}(t_i, t_{i+1}) = \frac{\mid V_{t_i} \bigcap V_{t_{i+1}} \mid}{\mid V_{t_i} \bigcup V_{t_{i+1}} \mid} \tag{3.5}$$

The statistically significant limit of Jaccard Similarity is studied before by (Real & Vargas, 1996). They propose a probabilistic null-model related to the sizes of compared sets. The score of this model is giving the lower limit that two sets are similar by chance. In other words; if the Jaccard score of two sets is above the significant limit, they are statistically similar. This null model (3.6) measures the randomness of two sets by taking their common elements into consideration.

$$P = \frac{\sum_{x=0}^{C} \binom{A+B-x}{x}}{\sum_{x=0}^{min(A,B)} \binom{A+B-x}{x}} \tag{3.6}$$

The values A and B are defined as the numbers of the elements of each set, and C is the number of elements in common. Since the maximum number of elements in common can be the minimum value of A and B, P gives the limit of two consecutive sets having C elements in common by chance. In this study, A and B represent the

number of node sets (or link sets) of two network snapshots we compare. Similarly, C is the number of common elements in those sets. If compared snapshots are similar, their similarity must be higher than P value, since being more similar by chance suggests that the system stability is still ongoing. In this case, system can be explained by one of those snapshots statistically. Thus this part of the network can be compressed. Contrarily, if the similarity is less than P, the snapshots are not similar. They should be represented separately. We define a new compression ratio based on this idea. For given w, $S_w = \langle S_1, \ldots, S_t \rangle$ is time series of a similarity metric, $P_w = \langle P_1, \ldots, P_t \rangle$ is time series of statistical significance limits of the each member of S_w and $C_w = \langle S_1 - P_1, \ldots, S_t - P_t \rangle$ is time series of the difference between measured similarity and its statistical significance. We define a new compression metric as the ratio of the number of elements in uncompressed similarity time series S_w to the number of elements which is larger than zero of C_w. We call this new statistics as *link* and *node compression* for the different similarity scores used in S_w.

Because similarity scores are calculated for consecutive snapshots, they do not explain features of a snapshot but the stability of the system. That is why, we do not consult to the variance of similarity score. We determine the best w as the one where the node and link compression is the lowest. That window size allow to extract the most variant and not redundant snapshots. We show the correctness of node and link compression for finding the best w by comparing their results with TWIN as it is the ground-truth in this work.

3.3 Aggregation Based Algorithm

Most of the existing methods suffer from using fixed window size for all snapshots of the network. However, although the chosen w allow to represent the system with the most variable and non redundant snapshots, there still can be some periods where the system stay stable. As a result, several similar snapshots can be extracted. That is why; we propose a new window aggregation strategy for extracting a dynamic network with fewer number of snapshots having all critical and necessary information for network analysis. The main idea behind our aggregation strategy is that if a candidate snapshot is similar enough to already existing snapshots, we do not represent it separately in the final dynamic network but aggregate it into the existing model. The aggregation process is iterative and simple:

1. Determine window size, w, for entire network
2. Create first snapshot G_1 from first time point of aggregation with duration w
3. For each following snapshot G_i with duration w
   - If G_i is similar with G_1, aggregate and continue iteration
   - Else, end aggregation process and go back to Step 2 for beginning new aggregation process for the rest of time slices.

Algorithm 1 *Automatic Time Discretization for Dynamic Networks*

Require: $\mathcal{C}, w, t_1, t_\theta$
Ensure: $\mathcal{G}$
1: $G_{init} \leftarrow extract(\mathcal{C}, t_1, w)$
2: $G_{aggr} \leftarrow G_{init}$
3: $t_i \leftarrow t_1$
4: **while** $t_i < t_\theta$ **do**
5: $G_i \leftarrow extract(\mathcal{C}, t_i + w, w)$
6: $sim \leftarrow S(G_{init}, G_i)$
7: $limit \leftarrow P(G_{init}, G_i)$
8: **if** $sim < limit$ **then**
9: $\mathcal{G} \leftarrow add(\mathcal{G}, G_{aggr})$
10: $G_{init} \leftarrow G_i$
11: **else**
12: $G_{aggr} \leftarrow aggregate(\mathcal{C}, G_{aggr}, G_i)$
13: **end if**
14: $t_i \leftarrow t_i + w$
15: **end while**

Automatic time discretization algorithm for extracting proper dynamic networks is given in algorithm 1. Accordingly, the process requires timely evolving complex system $\mathcal{C}$, time window size w, beginning time point t_1 and ending time point t_θ and returns dynamic complex network $\mathcal{G}$. Lines #1,2 and 3 are dedicated to the initialization. In the beginning, initial snapshot is extracted from $\mathcal{C}$ at line #5. After that, similarity value, and the limit value are calculated at line #6, 7, respectively. As stated previously, the algorithm checks whether these two snapshots are similar enough to be aggregated, i.e., similarity score being larger than limit value. If the snapshots are not similar, the snapshots aggregated so far are added to $\mathcal{G}$ at line #9.

After that, in order to continue the process, current snapshot becomes the initial snapshot for the rest of the calculation from this point on (line #10). Otherwise, if the snapshots are similar, then they are aggregated to form a single snapshot that represents both of them at line #12. Finally, time slice is shifted by the window size at line #14, to continue calculating next iteration till the end of time series.

Let us remind that if w is too large, some critical time points where the system exhibits important change can be ignored. But, if w is too small, consecutive snapshots can be too similar especially if the system is changing slowly. Here, we use the link similarity and its statistical significance which are introduced in equations (3.4) and (3.6) respectively. If the system is changing slowly, comparing consecutive snapshots can be misleading because the changes occur slowly and they cannot be captured in short duration. In order to capture these smooth and unseen changes, we propose to measure the similarity of the current snapshot with the first snapshot that the aggregation process is started rather than comparing consecutive ones. One of the most important points for the aggregation process is to decide if the measured similarity is critical. If J_{link} of the snapshot under investigation G_i with G_1 is higher than P, we aggregate G_i into the already aggregated networks with G_1. Aggregated dynamic network can be seen as the compressed network including informative snapshots and not having redundancies. Although we use link similarity here, our aggregation process can be applied with any network similarity metric having a significance threshold.

3.4 Studied Topological Properties

In the experiments, we use eight topological properties which are commonly used in network analysis listed in following sections with their general formulas (Newman, 2003; Albert & Barabási, 2002; Freeman, 1978; Dorogovtsev & Mendes, 2002) to analyze the topology of the extracted networks. We observe the evolution of their values taken for different snapshots as time goes by under the form of time series. These snapshots are both aggregated and non-aggregated a.k.a constant window sized snapshots. As we will see in section 5, these topological properties are used to prove our claims regarding to aggregation strategy.

As we explained in section 3, for a dynamic complex system $\mathcal{C}$ whose members and

their interactions evolve in continuous time interval that is defined as time points in $[t_1, t_\theta]$, $\mathcal{C}$ can be represented as $\mathcal{G} = \langle G_1, \ldots, G_\theta \rangle$. Therefore, a member of this set, which we called snapshot, is expressed as $G_i = (V_i, L_i)$ where i represents a sub-interval in $[t_1, t_\theta]$.

3.4.1 Node and Link Number

We defined V_i as a set of nodes, and L_i as its set of links. The node number of a network is the total number of the nodes, i.e., the size of the V_i set. The formula of node number, n, is given in 3.7. Similarly, link number, m, is the size of L_i set. The formula of link number is also given in equation 3.8.

$$n = |V_i| \tag{3.7}$$

$$m = |L_i| \tag{3.8}$$

3.4.2 Network Density

Network density is another topological property we look into. It is the ratio of link number of the network to possible link number. Density gives the information about network sparseness. As D getting closer to 0, the network sparseness increases. In order to calculate the density of a network, one can use the equation 3.9

$$D = \frac{m}{\frac{n(n-1)}{2}} \tag{3.9}$$

3.4.3 Average Degree

For a given node, u, the degree of the node, k, is the total number of links connected to it. The average degree, $\langle k \rangle$, gives the overall average of degree distribution. The equation 3.10 is the formula for calculating average degree of a network.

$$\langle k \rangle = \frac{2m}{n} \tag{3.10}$$

3.4.4 Number of Connected Components

In graph theory, connected components are defined as a group of nodes that are connected to each other, bu disconnected from the rest of the graph. The total number of these connected components is a topological property of a network.

3.4.5 Diameter

For a given graph, let us say $d(u, v)$ is the distance between nodes u and v. If we calculate all the distances, i.e., *shortest paths*, of any two pair of nodes, diameter of a graph would be the longest of all the shortest paths. Diameter of a network can be found using the formula given in equation 3.11.

$$diam = max_{u,v \in V_i}(d(u, v)) \tag{3.11}$$

3.4.6 Average Path Length

We defined $d(u, v)$ as the distance between nodes u and v. If all the shortest paths for all possible pair of nodes are calculated, the average value of these distances is called average path length. Average path length is the property that tells how many steps it would take to get from one node to another node on average. It can

be confused with diameter; however, diameter is the maximal shortest path whereas average path length is the overall average of the shortest paths (Albert & Barabási, 2002). The formula of average path length of a network is given in equation 3.12

$$z = \frac{1}{n} \sum_{u,v \in V_i} d(u, v) \tag{3.12}$$

3.4.7 Transitivity

For a given node u, *transitivity* is the ratio of number of triangle connections to number of connected triples where u is linked. The transitivity of the whole network is the average of all transitivity values calculated for all nodes. In social networks, if a node u is connected to node v, and node v is connected to node z; then, u being connected to z is highly probable. This property can be seen as "the friend of my friend is also my friend" property in terms of social language (Newman, 2003). The formula to calculate the transitivity a.k.a. clustering coefficient can be seen in equation 3.13.

$$CC = \frac{1}{n} \sum_{u \in V_i} \frac{triangles(u)}{triples(u)} \tag{3.13}$$

4 EXPERIMENTAL SETUP AND COMPARATIVE ANALYSIS

In this section, we first introduce the datasets we use to conduct our experiments. Following that, we give samples of the raw datasets and explain the preprocessing steps we take to prepare the datasets in order to work on them. Finally, we give the definitions of the similarity metrics we use to compare aggregated and non-aggregated snapshots.

4.1 Dataset and Preprocessing

We conducted experiments on three well-known datasets; Enron Email, Haggle Infocom, and MIT Reality Mining datasets. These datasets are used in the studies dedicated to the same problem with us before (Clauset & Eagle, 2012; Sulo et al., 2010). The statistics of these sets are given in Table 4.1.

Table 4.1: Datasets

Data Set Name	**Size**	**Time Span**
MIT Reality Mining (Sulo et al., 2010; Uddin et al., 2017; Caceres et al., 2011; Clauset & Eagle, 2012; Eagle & , Sandy; Fish & Caceres, 2017; K. Darst et al., 2016)	90 users	9 months
Enron Email (Sulo et al., 2010; Fish & Caceres, 2017; K. Darst et al., 2016)	151 employees	53 months
Haggle Infocom conference (Caceres et al., 2011; Fish & Caceres, 2017)	41 participants	4 days

First dataset we use is Enron dataset which is an emailing set sent to or received by Enron employees between 1997 and 2003. Each email has a timestamp with

the content of the email. We used From and To fields of each email as nodes to generate networks with the timestamp. When there are multiple emails in the To field, we split these emails as if they are unique emails. Therefore, the clean dataset we generated only consists of Date field, a single email address for From field, and lastly a single address for To field. Each From and To fields represent a node in the networks we created. Thus, every transaction is a link between these nodes. Final preprocessed dataset we have formed is given in table 4.2. The networks we created are undirected networks since the direction information has no use in the scope of this study. For Enron, the duration less than 1 day results unrealistic snapshots having several unconnected components with sparse link structure. That is why we consider one window size $w = 1$ day.

Table 4.2: Enron Processed Dataset

Date	**From**	**To**
896283060	christopher.behney@enron.com	toni.schulenburg@enron.com
904674000	bruno.gaillard@enron.com	susan.mara@enron.com
904674000	bruno.gaillard@enron.com	mona.petrochko@enron.com
904674000	bruno.gaillard@enron.com	jeff.dasovich@enron.com
923259540	george.robinson@enron.com	larry.campbell@enron.com

MIT Reality Mining dataset consists of bluetooth discovery data suggesting social interaction between people (Eagle et al., 2009). The dataset is generated at the MIT Media Laboratory over a nine-month period. During the study, 90 participants are given Nokia 6600 cell phones, and various data have been collected such as bluetooth device discovery scans at five-minute intervals, voice and text messages, active applications. This study only used bluetooth discovery data in which source mac address, and target mac addresses -discovered mac addresses- captured with the timestamp. In our study, we represent each discovery as a unique event similar to Enron dataset. The source mac address, and each target mac address represented as a row in order to define a transaction. Therefore, each mac address is a node, and each transaction is a link between these nodes. Final preprocessed dataset we have formed is given in table 4.3. As we did with Enron dataset, we created networks with these transactions with various time windows. We created network snapshots with 3-hour window, 6-hour window, 12-hour window, and lastly 24-hour window. We calculated network measures for each network and compared each network snapshot

with the previous snapshot for each window in terms of similarity measures. Lastly, we applied our algorithm to find dynamic time intervals for the dataset as well.

Table 4.3: MIT Reality Mining Processed Dataset

Date	**Mac**	**Device Macs**
1090244740	422cda52043e0000	42252a62135e0000
1090244740	422cda52043e0000	426412c1347f8000
1090244740	422cda52043e0000	422cda546b420000
1090245087	422cda52043e0000	422cda546c620000
1090245087	422cda52043e0000	42252a62135e0000

Haggle Infocom dataset contains bluetooth discovery data very similar to MIT Reality Mining dataset. The dataset contains 98 unique participants which have discovered 4724 unique devices during Infocom 2006 in Barcelona. Special devices called iMotes are deployed throughout the area. Seventeen of them are long range static iMotes, three of them have been placed in the lift of the hotel, and the rest of the 98 are the participants of the workshop. Instead of the mac addresses, we deal with the IDs for this dataset. Therefore, we considered these IDs as network nodes when we generated our networks. Each row in the dataset we processed represents a discovery event. Hence, they are considered as links between nodes. The first column of the dataset is the timestamp of the discovery event. Since the experiment was done between April 24th to April 27th, 2006 and the iMotes are activated on April 23rd at 5:01 pm, we shifted the timestamp to match the correct dates. Final preprocessed dataset we have formed is given in table 4.4. Lastly, as we did with the Enron Email dataset, and MIT Reality Mining Dataset, we created networks with several different windows. We created these network snapshots every minute, every five minutes, every fifteen minutes, and every hour. After creating these snapshots, we calculated network measures similar to previous datasets, and also similarity measures with the previous snapshot for every windows. Finally, we calculated the dynamic time intervals for this dataset with the algorithm we suggested in this study.

Table 4.4: Haggle Infocom Processed Dataset

Date	**From**	**To**
1145817157	17	13
1145817283	13	17
1145817532	3	14
1145817620	16	53
1145817670	14	3

4.2 Comparative Analysis

The main focus of experiments is revealing the effect of aggregating snapshots with our proposal in comparison with non-aggregated snapshots. Hence, we compare the dynamic networks that are extracted with aggregation process with the ones extracted with constant window sizes in different perspectives. The most common way of such comparison is to analyze the quality of the time series generated with the topological properties of snapshots (Sulo et al., 2010). We compare topological properties qualitatively by their box plot graphics.

Recently, the usage of network similarity metrics' is proven to be effective not only for such comparison but also for determining the most proper window size (Orman et al., 2021). Thus, besides topological properties, we also use node similarity, link similarity, neighborhood similarity which we define in next section, and lastly adjacency correlation coefficient which we also define the section after neighborhood similarity section for comparison and finding the most proper window size.

In previous section, we have given the definition of *Link similarity* as $J_{link}(t_1, t_i)$ (3.4). Similarly, we define *Node Similarity*, $J_{node}(t_1, t_i)$, with equation 3.5. Next subsections, we will define neighborhood similarity, and adjacency correlation coefficient.

4.2.1 Neighborhood Similarity

Let $N_{t\_1}(v)$ and $N_{t\_i}(v)$ are the first order neighborhood of $v \in V_{t\_1} \bigcap V_{t\_i}$ at previous and next snapshots respectively. We define for a given node, v, its *neighbor stability*, $\delta(v, t\_1, t\_i)$ as Jaccard Similarity of $N_{t\_1}(v)$ and $N_{t\_i}(v)$ (equation 4.1).

$$\delta(v, t\_1, t\_i) \frac{\mid N_{t\_1}(v) \bigcap N_{t\_i}(v) \mid}{\mid N_{t\_1}(v) \bigcup N_{t\_i}(v) \mid} \tag{4.1}$$

Neighborhood Similarity, $S_{neighbor}(t\_1, t\_i)$ (equation 4.2) is the average neighbor stability of the common nodes in previous and next snapshots. $S_{neighbor}$ uses nodes as units. It reflects the average neighborhood stability over nodes while S_{link} concentrates directly on links. If network node numbers do not change too much, and network centralization is not large, S_{link} and $S_{neighbor}$ contains similar information. However, if there are considerable nodal changes, the result scores of those two metrics fall apart. For example, for a star-shaped network whose most of its non-central nodes leave as time goes by, it's S_{link} and $S_{neighbor}$ would be different.

$$S_{neighbor}(t\_1, t\_i) = \frac{1}{\mid V_{t\_1} \bigcap V_{t\_i} \mid} \sum_{v} \delta(v, t\_1, t\_i) \tag{4.2}$$

4.2.2 Adjacency Correlation Coefficient

We compare the proposed similarity metrics performance with *adjacency correlation coefficient*, γ. γ is also developed for measuring the similarity of consecutive snapshots by (Clauset & Eagle, 2012) for adjusting w. It is the correlation of adjacency matrices of consecutive snapshots. The authors define A as an $n\,x\,n$ adjacency matrix of a graph where n is the node number. If the nodes i and j is connected, $A_{i,j} = 1$. On the other hand, if they are disconnected, $A_{i,j} = 0$. They suggest a more accurate metric by calculating the correlation between the row elements of $A^{(x)}$ at time x and $A^{(y)}$ at time y, which are nonzero at least once in either of them. They define this union as $N(j)$. Therefore, the adjacency correlation for j is given in equation 4.3. Comparative performance analysis of similarity metrics

with γ by their sensitiveness to reflect the noise and diversity level for different w is explained in section 5.

$$\gamma_j = \frac{\sum_{i \in N(j)} A^{(x)}_{i,j} A^{(y)}_{i,j}}{\sqrt{(\sum_{i \in N(j)} A^{(x)}_{i,j})(\sum_{i \in N(j)} A^{(y)}_{i,j})}} \tag{4.3}$$

5 RESULTS

In this section, we first report the topological quality of the aggregated snapshots in comparison with snapshots extracted with constant w. Then, we touch upon how the method we proposed can help to choose the proper window size. Finally, we interpret time series of snapshot similarities by taking their probability density function analysis into account.

5.1 Topological Quality of Snapshots

We compare the network snapshots extracted with the method we proposed, which we call aggregated snapshots, with the networks extracted with constant w in terms of topological properties. In Fig. 5.1, topological properties of aggregated and constant snapshots for Enron dataset are shown as box plots. Accordingly, for almost all topological properties, aggregated snapshots have less noise. More clearly, their topological properties do not take extreme values in any snapshot. Moreover, they have fewer number of connected components with higher density. Their result values are also similar with well-known real-world networks (Newman, 2003). Contrarily, snapshots with constant w might have larger average distance or diameter values with high number of connected components. Briefly, we observe more consistent and realistic snapshots in the aggregated ones.

Similarly, box plots showing the topological properties of snapshot series for Haggle networks are shown in Fig. 5.2. We extracted four different sets of dynamic networks in respect to different w values. In general, aggregated snapshots' topology seems less noisy than it is for constant snapshots. Their number of connected components, diameters, and average distances are smaller. Node numbers, link numbers, average degrees, and densities are larger. Thus, the aggregation process seems to extract more compact and connected snapshots with higher transitivity. Considering w

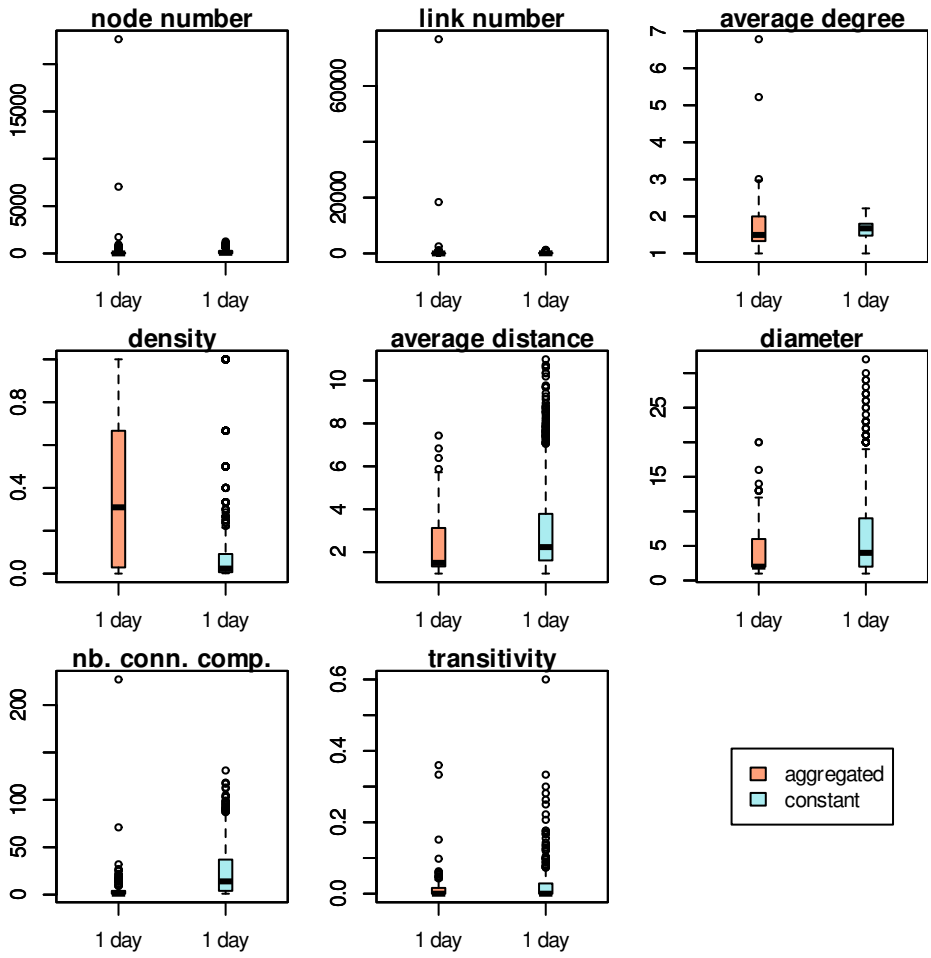

Figure 5.1: Topological Properties of Enron Snapshots extracted for $w = 1$ day

effect, the larger the w size, the more crowded, more transitive, and more connected the snapshots independent from their extraction strategy. We observe increase of the numbers of nodes and links, average degree, average distance, diameter, and transitivity as w increase in all snapshots. However, when $w >= 15$ minutes, aggregated snapshots node and link number with average degree, density, average distance, diameter, and transitivity increases clearly. Moreover, all its topological properties become less noisy. Nevertheless, we observe a few number of snapshots when $w >= 15$. Indeed, when the system is changing slowly, our strategy can result over aggregation. There is an obvious effect of chosen window size in the snapshots topology. In the next section, we concentrate on determining the most proper window size and the effect of aggregation strategy for the selection of w.

Lastly, MIT Reality Mining dataset, box plots showing the topological properties of snapshot series are given in Fig. 5.3.

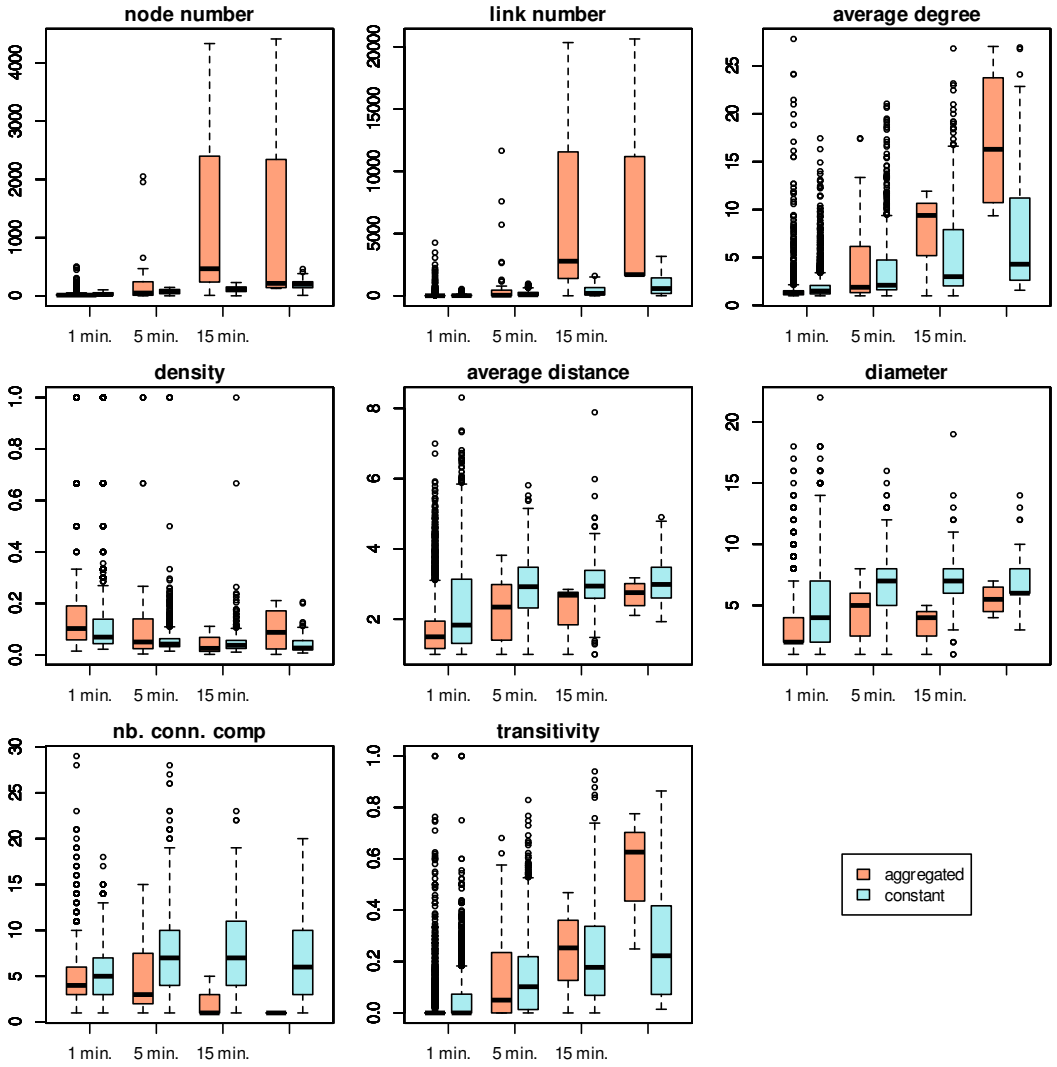

Figure 5.2: Topological Properties of Haggle Snapshots extracted for $w = 1, 5, 15, 60$ minutes

The results are quite similar to Haggle Infocom dataset results that are given in Fig. 5.2. We calculated the topological properties for four different sets of dynamic networks for different w values. The topological properties of aggregated snapshots are less noisy than the topological properties of constant snapshots. Also, we observe that the average distance is smaller for every window size meaning that the networks are more compact. Similar result can be extracted from diameter as well. Regardless of w, number of nodes, number of links and density values are larger. This results suggest that the aggregation process creates more dense networks, that contain more nodes and links, in which multiple snapshots can be represented successfully.

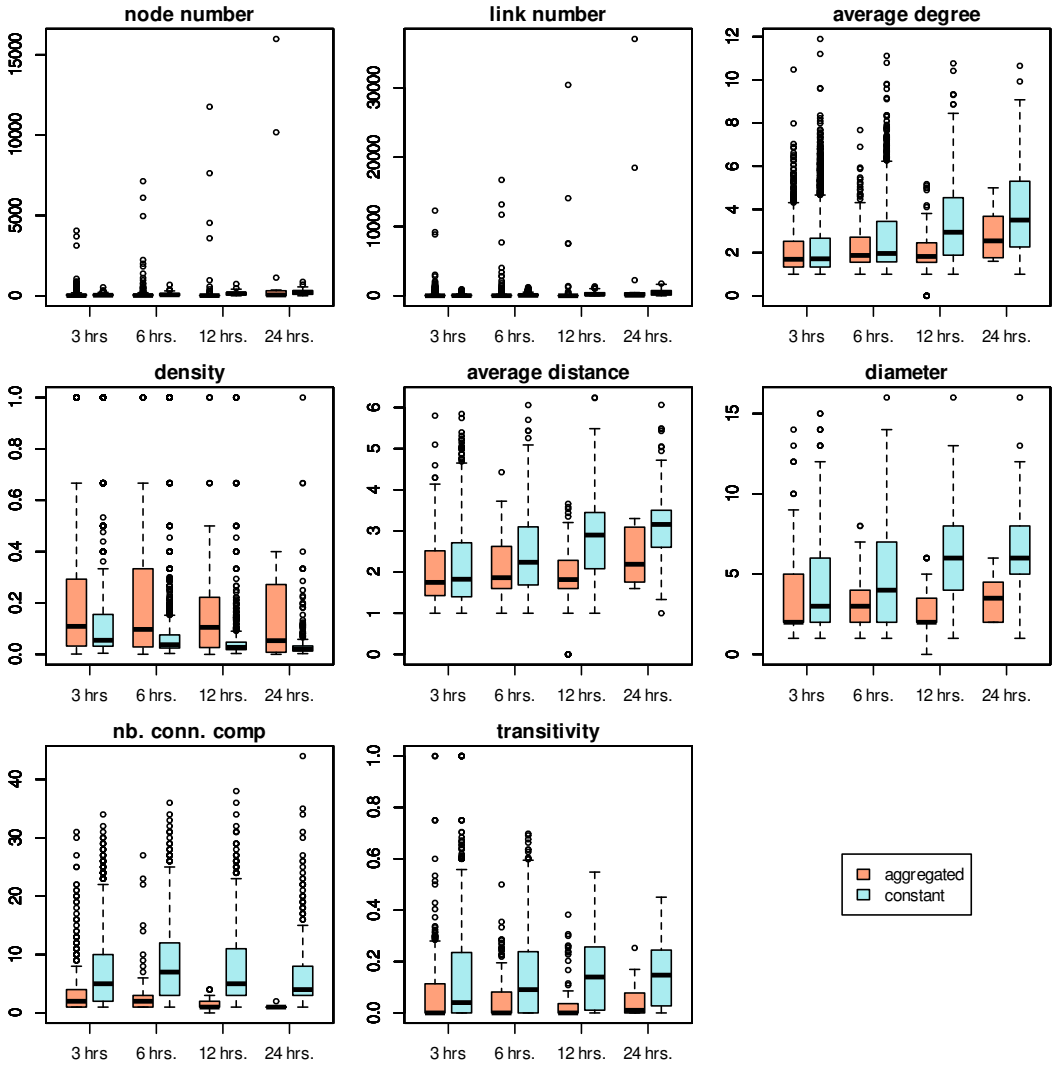

Figure 5.3: Topological Properties of Reality Mining Snapshots extracted for $w = 3, 6, 12, 24$ hours

5.2 Choosing The Best Window Size With Link And Node Compression Ratio

The best window size for three data sets are determined with TWIN, and link and node compression ratio we proposed. TWIN results are given in Fig. 5.4. In the figure, we show variance and compression ratio of the diameter for different w. The best w is the one that variance and compression ratio is the closest to each other. Accordingly, for Enron, the best w is 7 days. For Haggle Infocom, in fact, variance and compression ratio coincide on both 5 and 60 minutes. For Reality Mining, the best w is 3 hours. These results are mostly compatible with (Sulo et al., 2010).

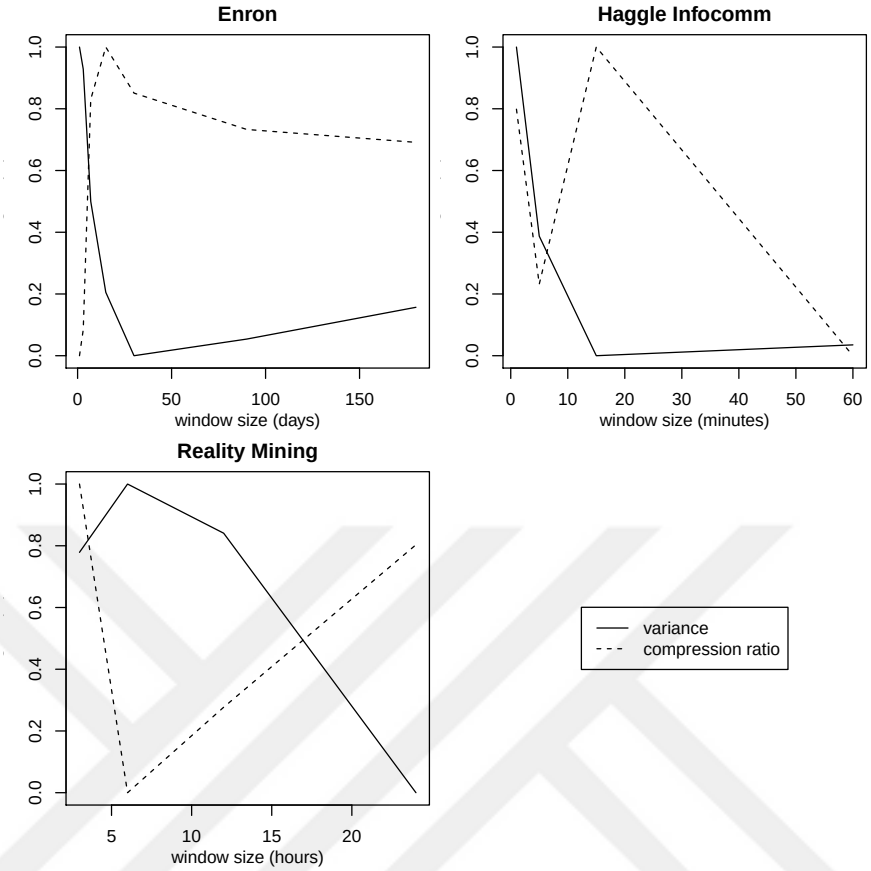

Figure 5.4: TWIN results calculated by variance and compression ratio of diameter for Enron, Haggle Infocom and Reality Mining. The most proper window sizes are 7 days, 5 minutes and 3 hours respectively.

We show the node and link compression results on Fig. 5.5. Accordingly, for Enron, we find 15 days which is a longer period for the best window size found by TWIN. Both node and link compression agree on this value. Our result is different from TWIN result. That's why; we examine in detail the diameter signal of Enron when w is 7 days and 15 days. The result is shown in Fig. 5.6. Accordingly; we found that the compression ratio for diameter is the same at 7 days and 15 days intervals. Since the variance of 7 days is higher than 15 days in calculations, TWIN determined 7 days as the best w. However, as we mentioned before, topological properties are not scale invariant. They can have different scores reflecting same information for two snapshots or vice versa. In our detailed analysis, we saw that the diameter scores found for these two w were different. When we look at the normalized standard deviation with the mean over the obtained values rather than simple variance, we found that the variation of the 15 days interval was higher. Thus, the measure can be misleading for explaining the noise of generated signals because of not being scale

invariant. Indeed, in the Fig. 5.6 the diameter signals for the two w are also visible. Accordingly, in the signal parts that coincided with the date of December 2001, when Enron announced his bankruptcy, the plot of $w = 7$ is both noisy and do not reflect the significance of the event with a critical line. Contrarily, we can see important details with less noise but in sufficient detail in the plot of $w = 15$. The critical diameter peak is also remarkable in the section coinciding with the bankruptcy date. More information can be retrieved from these signals. For example, George Bush named Kenneth Lay, who is Enron chairman and contributed more than $290,000 to George Bush's election campaign, as an adviser to his presidential transitional team in January 2000. We catch this event around in the plot of $w = 15$ where the snapshot number is around 32 with a critical spike. The diameter score is 15 for this snapshot. However, this event cannot be caught in the plot of $w = 7$. This event occurs around snapshot #67. When we take a look at the diameter score of this snapshot, it can be seen that the value is 10. The diameter score of the snapshots before and after of this snapshot is 9 and 8, respectively. Similarly, around February and March 2002, the hearing starts before Senate. This event is clearly visible in the plot of $w = 15$ around snapshot #85. The same event occurs between snapshot #174 and #180 in the plot of $w = 7$. We can see that the event reflected in the plot of $w = 7$ is represented by multiple snapshots. Therefore, the signal generated by these snapshots' diameter score are more noisy, whereas the event can be clearly captured by a single snapshot when the selected w is 15 days. Moreover, in June 15, 2002 Arthur Andersen is convicted. This event is captured by both $w = 7$ and $w = 15$. If we take a look at snapshot #91 in the plot of $w = 15$, we see that the diameter score increases from 6 to 9. The 50% increase suggests that a significant event occurred during the period of snapshot #91. The event as we said before was the conviction of Arthur Andersen. The same event occurs snapshot #192 for $w = 7$ snapshots. We can see the peak value of 13 in the plot of $w = 7$ around that snapshot as the last peak of the plot. We could catch many important events such as the resignation of chairman and chief executive of Enron at the peak points of the signal of $w = 15$. When we look at the overall picture, the window size found by node and link compression seems to be more informative for this system than the one that TWIN found.

For Haggle Infocom, link compression results that the best w is 5 minutes but node compression have opposite behavior. In fact, node similarity represents the stability of the actors of the system. If the system is changing slowly or the studied w sizes are

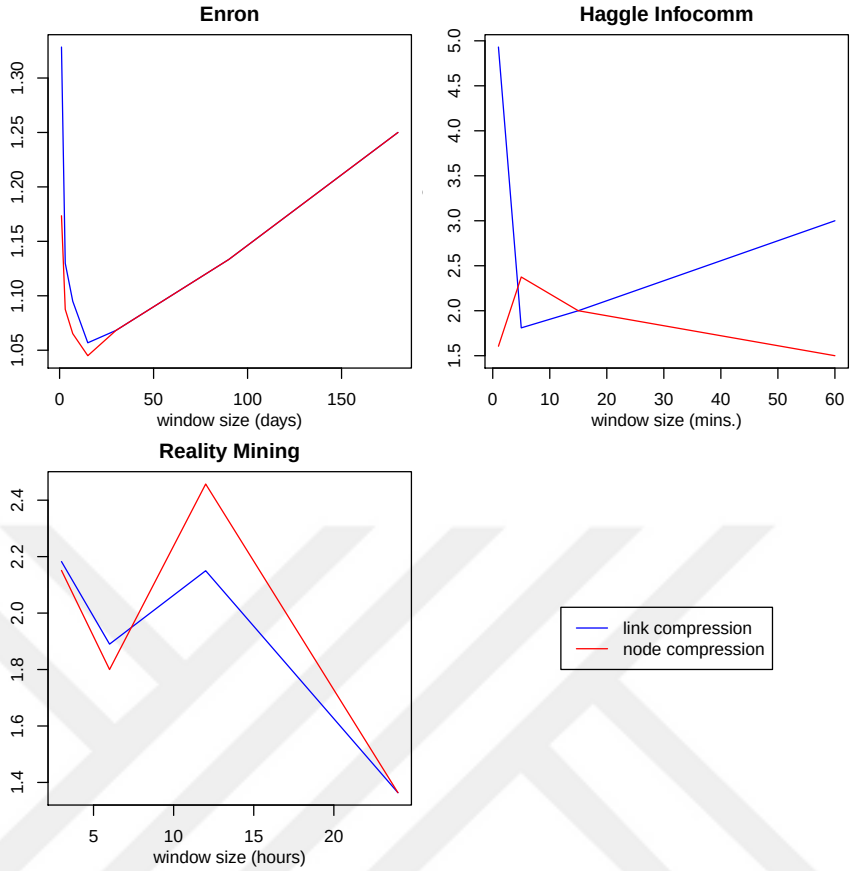

Figure 5.5: Link and Node Compression results for Enron, Haggle Infocom and Reality Mining. The most proper window sizes are 15 days, 5 minutes and 6 hours respectively.

too short, it can be expected that nodes of the system do not change considerably. Link similarity represents the stability of the connection of system actors. Although the system actors stay same, their connections can change. Let us remind that Haggle Infocom is a set of controlled experiments including Bluetooth connections on a predefined environment. Thus, in this system, we take into account the results of link compression. For Reality Mining, the lowest values of compression scores are obtained when w is 6 and 24 hours. Interestingly, these are two different best w explained in (Sulo et al., 2010) and (Eagle & , Sandy) respectively. Thus, if we want to capture sudden changes occurring daily, we extract network with $w = 6$. But if we want to observe the system with its macro changes and evaluate them, we use $w = 24$. TWIN finds 3 hours for the best w. The detailed analysis reveal that, it is because of the same reason that we explained for Enron.

We also compare the signal of topological properties of the extracted networks with

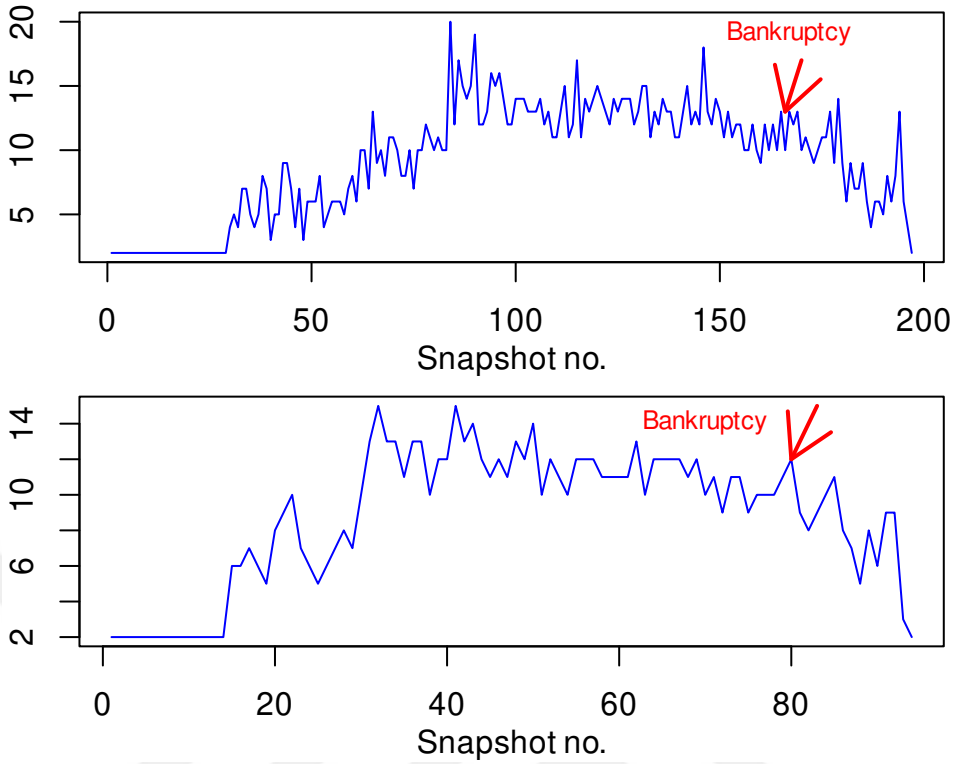

Figure 5.6: Top and bottom time series are Enron diameter scores for $w = 7$ and $w = 15$ days respectively. The biggest bankruptcy in US history happened at December 2001. The date is indicated with red arrow in both plots.

aggregated and constant windows for the best w. In figure 5.7, diameter and average distance signals for constant (top layout) and aggregated (bottom layout) windows for Haggle Infocom data set is shown. The selected window size for this analysis is $w = 5$. The signals of constant windows show a circadian rhythm, but they are too noisy. This data set is collected for four days. We clearly observe daily cycles between day and night in constant window usage with the periodic increase and decrease at the signals. This result is already expected before data analysis. These types of circadian rhythms are common and explained in detail in (Aledavood et al., 2015). That is why; obtaining circadian signals does not add value. However, although the signals generated for aggregated snapshots have no distinct trend, one can detect peak values and important changes in time points. Because circadian periods are aggregated and represented with fewer numbers of snapshots, we observe only important changes in those signals. The red circled zones in Fig. 5.7 correspond to the same period; the last night of conference. Indeed, the number of nodes and links decreases and diameter and average distance get shorter. Aggregation process merges many snapshots before that period. Because the diminishing at this period is

more considerable than others, they are represented as separate snapshots. However, we cannot notice the considerable diminish and sudden increase by using constant window. Last night of the conference seems like other nights for the extraction with constant window. We cannot understand its difference because of having too repetitive snapshots, i.e. being noisy signal.

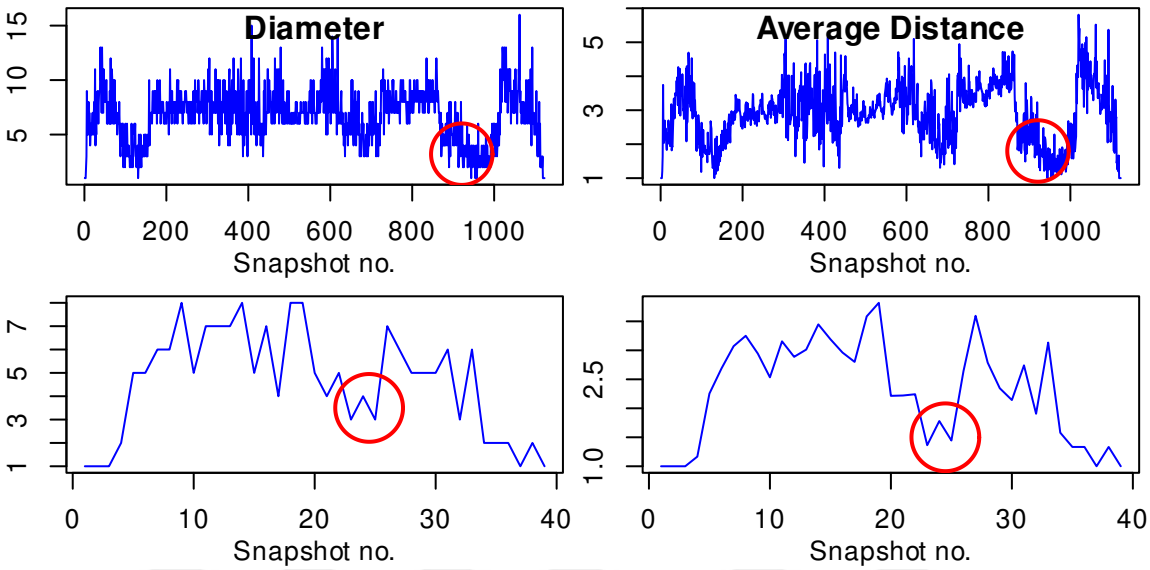

Figure 5.7: Diameter and Average Distance Signals for $w = 5$ minutes for Haggle Infocom. The plots of the snapshots extracted with constant window and aggregated windows are given in top and bottom layout, respectively. The red circles correspond to the same period of critical decrease for all plots.

5.3 Choosing The Best Window Size With Aggregation

In order to choose the most proper window size, we examine the noise and informativeness of the time series of extracted snapshots' similarities. We use σ and d of the time series as it is done in (Caceres et al., 2011).

In Fig. 5.8, the change of σ and d of four similarity metrics in function of w are given for both aggregated and constant snapshots for Haggle dataset. For constant w, the σ of node, neighborhood similarities and adjacency correlation coefficient decrease until a certain point between 5 and 15 minutes. Link similarity trend is different from others. String diversity increases until w is 5 to 15 minutes and then it stays stable. For both statistics, although the general trends of similarity metrics

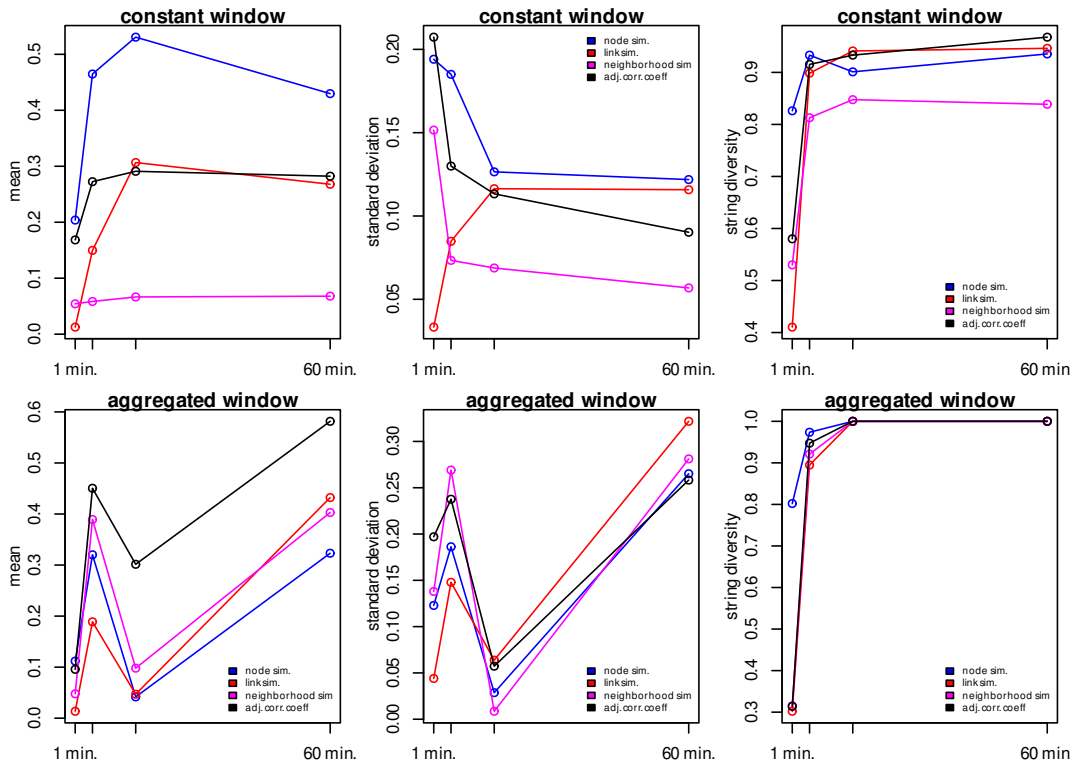

Figure 5.8: Mean (left), Standard Deviation (middle) and String Diversity (right) scores obtained for the time series of different consecutive snapshot similarities for Haggle Infocom dataset. The plots of the snapshots extracted with constant window and aggregated windows are given in top and bottom layout, respectively. Different colors of the lines are dedicated to different similarity metrics.

are similar, we do not observe a consensus at their values or at the peak points. Evaluating both statistics' results, the least noisy and the most informative time series are generated for the snapshots when w is between 5 and 15 minutes.

For aggregated snapshots, σ and d of all similarity metrics peak at 15 minutes. The larger the string diversity value, the more informative the snapshots are. Therefore, starting from 5 minutes, the aggregated snapshots are more informative than constant snapshots. All similarity metrics have a consensus on those two statistics. Comparing two snapshot extraction strategy, aggregated window usage allows us to determine the proper time interval independent from studied similarity metric. That is why; it is more similarity-metric independent. The most proper window size, w, is where low variance and high string diversity are achieved. Regardless of similarity metrics, one can see that the best window size is between 5 minutes and 15 minutes when we take both σ and d into account. Nevertheless, we remind that

for $w = 15$, our method over aggregates and results a few number of snapshots. By this cross interpretation, the most proper time interval can be 5 minutes for Haggle Infocom dataset.

When we look at the Fig. 5.9, these results can be seen more clearly. All of the similarity metrics are dropped significantly at 15 minutes mark for aggregated results. Also, for w being 15 minutes, all of the similarity scores are significantly less noisy. Even though, for w being 1 minute, similarity scores of aggregated snapshots seem to be more noisy, when we take a look at Fig. 5.8 we can see that 1 minute mark is the least informative window size for Haggle dataset. However, the significant drop for $w = 15$ is mostly because of over aggregation. Therefore, choosing 5 minutes for window size is the best option for Haggle dataset as we proved before since the second best similarity results are achieved $w = 5$ in Fig. 5.9.

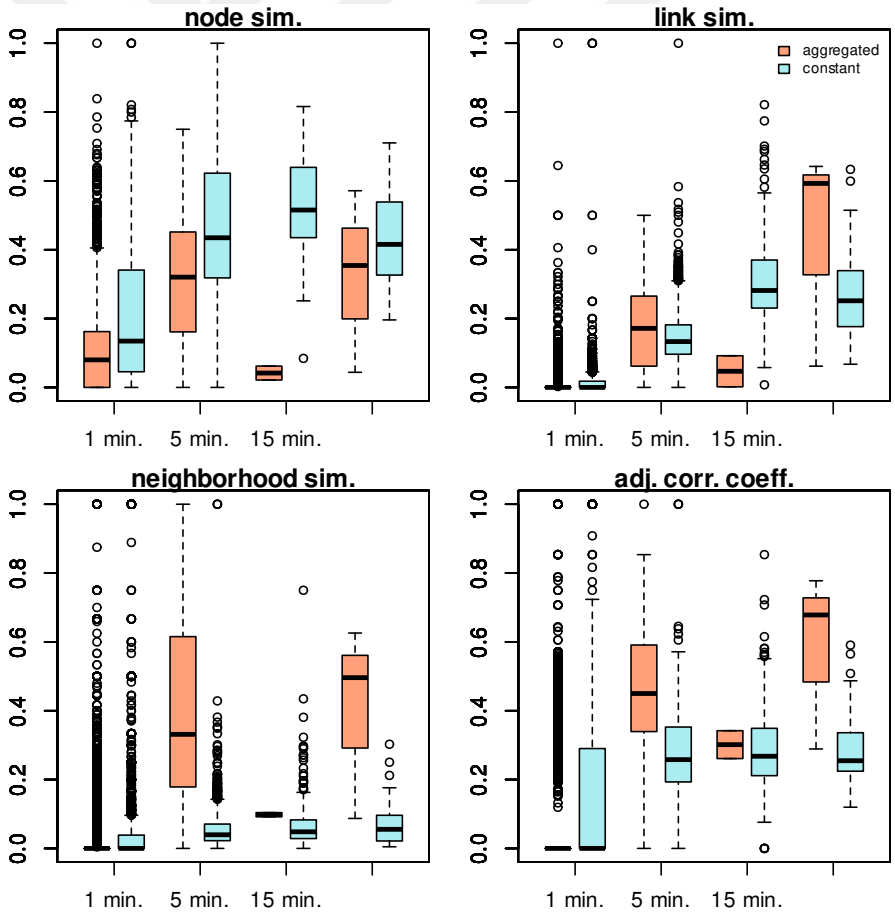

Figure 5.9: Similarities of Haggle Snapshots extracted for $w = 1, 5, 15, 60$ minutes

In Fig. 5.10, the change of σ and d of four similarity metrics in function of w are given for both aggregated and constant snapshots for MIT Reality Mining dataset. For

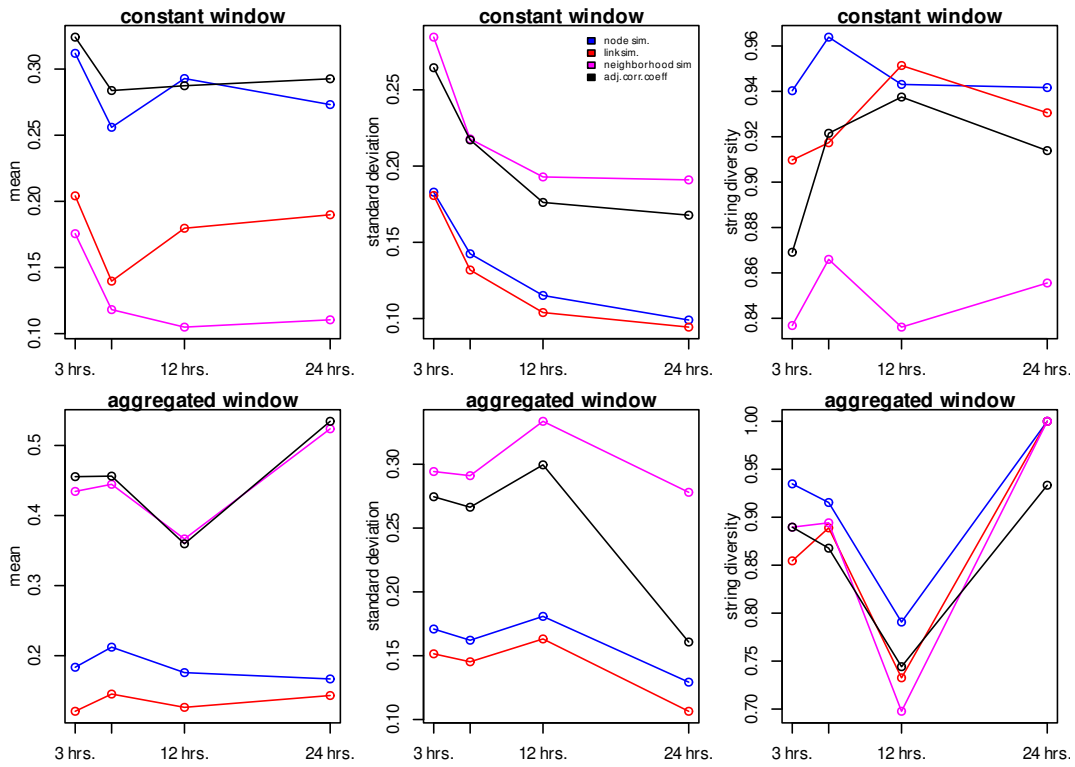

Figure 5.10: Mean (left), Standard Deviation (middle) and String Diversity (right) scores obtained for the time series of different consecutive snapshot similarities for Reality Mining dataset. The plots of the snapshots extracted with constant window and aggregated windows are given in top and bottom layout, respectively. Different colors of the lines are dedicated to different similarity metrics.

constant w, the σ of all similarity scores decrease. On the other hand, string diversity scores peak at w being 6 hours for node similarity and neighborhood similarity. However, for link similarity and adjacency correlation coefficient, string diversity scores peak at 12 hours mark. Contrarily, the least noisy time series are generated at $w = 24$. Therefore, similar to Haggle Infocom dataset analysis, we do not observe a consensus at their values or at the peak points.

For aggregated snapshots, d of all similarity metrics peak at 24 hours. Since, larger d values mean more informative snapshots, $w = 24$ seems to be the most informative window size for snapshots. For σ, we are looking for a window size where the standard deviation is the smallest since the most proper window size, w, is where low variance and high string diversity are achieved. Similar to Haggle Infocom dataset, all similarity metrics have a consensus on those two statistics. These results validate our previous deduction which is using aggregated window allows us to determine the

proper time interval independent from studied similarity metric. Consequently, one can see that the best window size is $w = 24$.

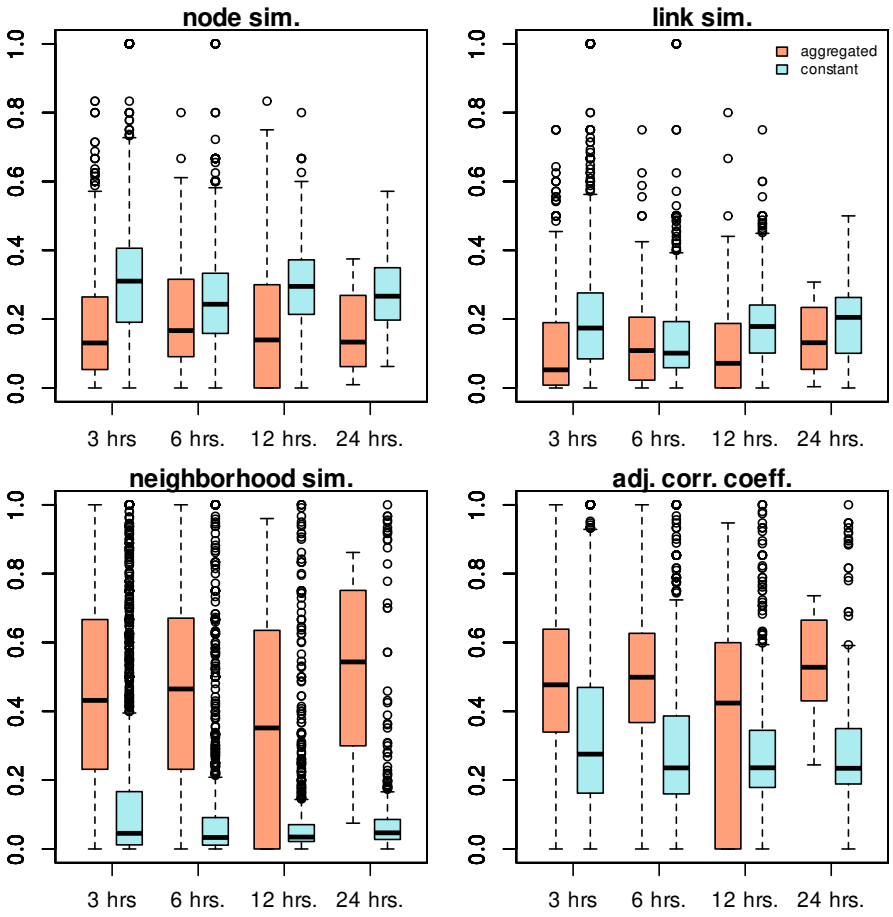

Figure 5.11: Similarities of Reality Mining Snapshots extracted for $w = 3, 6, 12, 24$ hours

When we look at the Fig. 5.11, other than similarity scores for aggregated snapshots being significantly less noisy, we cannot infer anything regarding to window size selection. Therefore, examining the noise and informativeness of the time series seems to be crucial for choosing best window size.

5.4 Aggregated vs Constant Window

In Fig. 5.12, the node similarity time series signal result is given for Enron dataset. Node similarity signal of snapshots with the constant w is difficult to analyze since there are no characteristic changes that give meaningful information about the system. The time series is a sort of a flat signal which is not informative. On

the other hand, one can detect meaningful changes on node similarity signal of aggregated snapshots. For example, the similarity is increased around snapshot #30. The similarity change can be detected around #55 as well. However, we cannot detect these changes on the signal of constant snapshots. Therefore, we detect instantaneous changes in the system with aggregation strategy. Moreover, we detect peak values which are the points where the system shows a considerable change. Link similarity signal also gives similar results in Fig. 5.13. Constant snapshots are quite similar to node similarity. It is mostly a flat signal giving hardly any information about the system. However, the signal of aggregated snapshots is informative compare to constant ones. Therefore, one can detect changes and determine peak values. Similarly, for neighborhood similarity, we can detect instantaneous changes in the system with aggregation strategy in Fig. 5.14. The signal is mostly very noisy, and hardly gives any information about the system. However, we can see that with aggregation strategy, we can infer significant changes in the system.

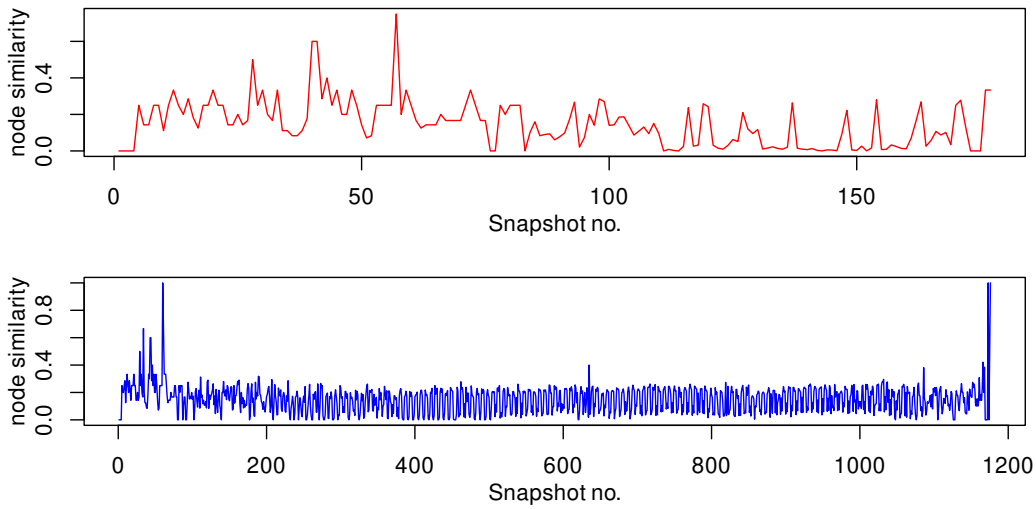

Figure 5.12: Node Similarity signals of Enron snapshots. The plots of the snapshots extracted with aggregated windows and constant window are given in top and bottom layout, respectively.

These remarks are also validated with Haggle dataset. We can see node similarity, link similarity, neighborhood similarity, and adjacency correlation coefficient signals for both constant snapshots and aggregated snapshots in Fig. 5.15. The selected window size for this analysis is $w = 5$. If we take a look at node similarity and adjacency correlation coefficient plots for constant snapshots, the signals show a circadian rhythm, but they are too noisy. This dataset is collected for four days. We clearly observe daily cycles in constant window usage. This result is already

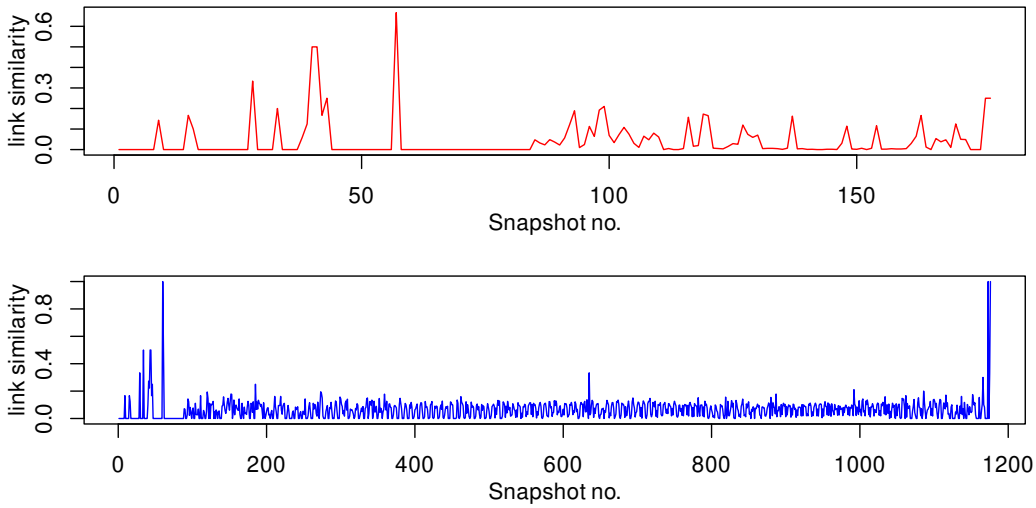

Figure 5.13: Link Similarity signals of Enron snapshots. The plots of the snapshots extracted with aggregated windows and constant window are given in top and bottom layout, respectively.

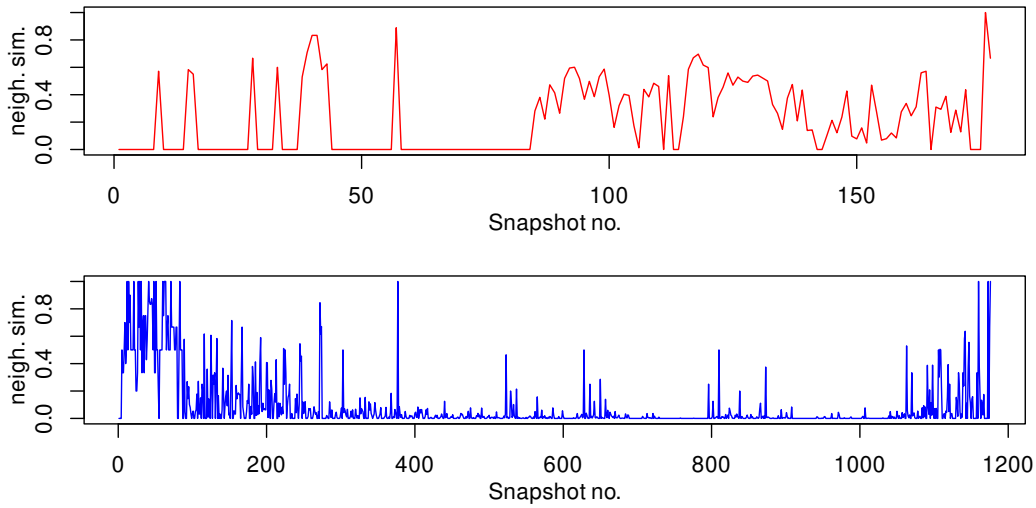

Figure 5.14: Neighborhood Similarity signals of Enron snapshots. The plots of the snapshots extracted with aggregated windows and constant window are given in top and bottom layout, respectively.

expected before data analysis. These types of circadian rhythms are common and explained in detail in (Aledavood et al., 2015). That is why; this result does not add value. However, although the signals generated for aggregated snapshots have no distinct trend, one can detect peak values and important changes in time points. Because circadian periods (a.k.a day time and night) are aggregated and represented

with few numbers of snapshots, we observe only important changes in those signals.

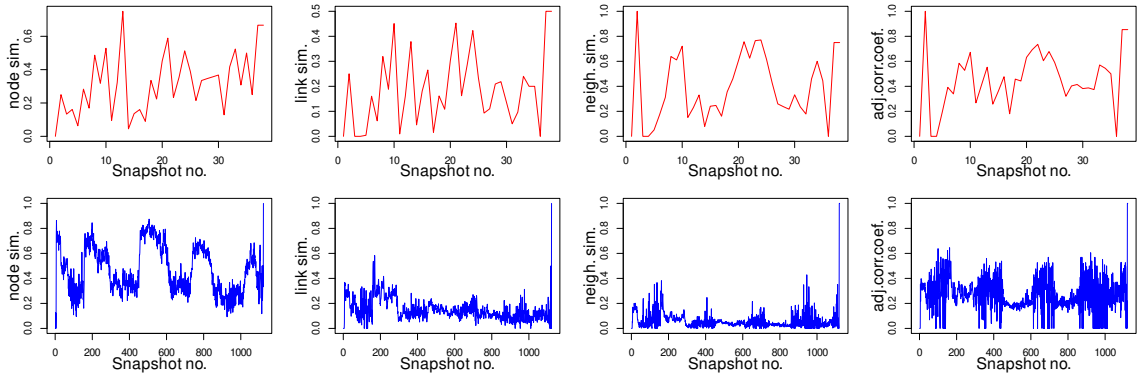

Figure 5.15: Node, Link, Neighborhood and Adjacency Correlation Coefficient Similarity signals of Haggle snapshots extracted for $w = 5$ minutes from left to right, respectively. The plots of the snapshots extracted with aggregated windows and constant window are given in top and bottom layout, respectively.

The results are more dramatic for Reality Mining dataset. The selected window size for this analysis is $w = 24$ since that is the best window size as we previously showed. In Fig. 5.16, we give the node similarity, link similarity, neighborhood similarity, and adjacency correlation coefficient signals. The bottom layout shows the constant window size, and the top layout shows the aggregated results. Node similarity, and link similarity signals are too noisy to be analyzed for constant window. These signals do not give any insights about the system. However, for aggregated window, we can see the behaviour of the system. The trend of the system is clearly visible. On the other hand, the neighborhood similarity signal of the constant window is mostly flat, even though, the aggregated window signal gives some information. Therefore, for constant window, when we look at the neighborhood similarity score, we cannot capture important changes since most of them are vanished. Similar result can be seen for adjacency correlation coefficient as well.

Finally, we analyze probability density functions for Haggle dataset with $w = 5$. The results are given in Fig. 5.17. All of the similarity metrics are noisier for the constant windows. Also, the distribution for the constant windows is a heavy-tailed distribution which is a lot harder to analyze compared to normal distribution. Moreover, the distribution of constant windows suggests that a large sum of the snapshots are too similar. On the contrary, the signals of similarity metrics, except for neighborhood similarity metric, of aggregated windows indicate that the distribution is a normal distribution suggesting that the similarity is low compare

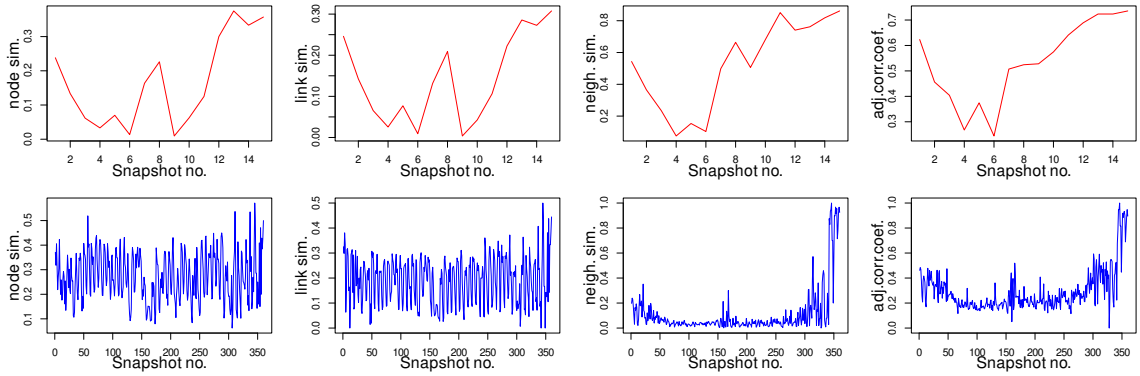

Figure 5.16: Node, Link, Neighborhood and Adjacency Correlation Coefficient Similarity signals of Reality Mining snapshots extracted for $w = 24$ hours from left to right, respectively. The plots of the snapshots extracted with aggregated windows and constant window are given in top and bottom layout, respectively.

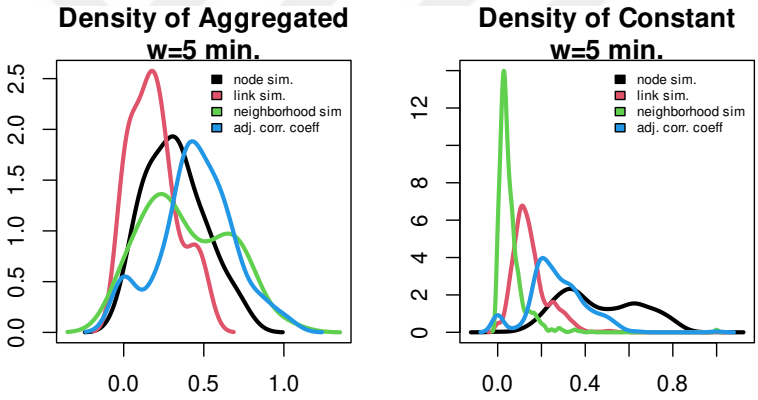

Figure 5.17: Probability Density Functions of the time series of different similarity metrics for Haggle snapshots extracted for $w = 5$ minutes. Snapshots extracted with aggregated windows and constant window are given in left and right layout, respectively.

to constant windows. It is symmetrical, easier to analyze, and consistent with the signal results we have.

6 CONCLUSION

We propose a method for determining proper window size and also an aggregation strategy for extracting informative dynamic networks for timely evolving complex systems analysis. The first method we propose is a compression ratio for evaluating the window size selection for the aggregation strategy. This method is based on scale invariant similarity metrics we also define; link similarity and node similarity. These similarities are based on Jaccard Similarity. We compare the performance of our method with TWIN which is one of the most accurate method in the literature for finding best window size. The results reveals that the topological analysis of the networks that are extracted with the window size that our method has found is more informative than the one of TWIN. TWIN suffers from using topological properties. It does not seem to catch some important events. TWIN has two issues; network topological properties are not scale invariant, and the compression ratio TWIN uses is meaningless for real values. Besides detecting window size more properly, we also propose an intelligent strategy for extracting proper dynamic networks. This method is based on the detection of the stability of the system by using the statistical significance of scale invariant similarity scores of consecutive networks. We determine a windows size, w, and shift the window in time while deciding whether we aggregate the snapshots or not. The criteria that we consult for aggregation is first measuring the topological similarity of snapshots, second deciding if this similarity is lower than expected randomness. Rather than using scale variant topological properties, we concentrate on Jaccard similarity based link similarity.

The main results are four folds. First, our aggregation strategy yields more realistic topological properties especially in terms of connectivity and compactness of the extracted snapshots than using constant windows. Furthermore, it allows to catch important events which are hidden in the noise of constant window sliding. Hence, we build an aggregation strategy both objective and simple. Our proposal is validated on the experiments with three well-known systems; Enron Email, Haggle

Infocom and MIT Reality Mining. Moreover, since the aggregation process generates fewer snapshots and as we showed that the process does not suppress the general characteristics of the network, it can be used for modelling the network. Second, it facilitates choosing the proper window size independent from the feature of generated time series. This was the main issue we were trying to solve in this work. Third, the signals of extracted snapshots features and their probability distributions are more informative and interpretable. Lastly, our proposed link and node compression ratio metric can be used to determine window size better than TWIN. Using this metric, we show that the more proper window size can be selected. Therefore, the aggregation process can be more effective, and the effects of the contributions regarding to aggregation strategy we explained previously can be more evident. The main drawback of the aggregation strategy is that if the system is slowly changing, it causes over aggregation. In this case, we might have a few numbers of snapshots and miss the time effect on the system at our analysis.

This work can be extended into different paths. First; the aggregation strategy can be improved in a way to overcome of over aggregation problem. Slowly changing systems can generate very few snapshots that conceal the general characteristics of the system. In order to improve the aggregation process to overcome this problem, a variety of approaches that takes changing rate into account can be proposed. For example, a method would be adjusting the null model that limits the similarity during the aggregation process. That way, the small information change in the system can be captured as well. An entropy based method could be used for adjusting null model. Second, new similarity metrics or network embedding methods for calculating similarity, which will measure the system stability can be proposed for aggregation strategy. Network embedding is a general term for methods that map networks into a low-dimensional representation of nodes while preserving the topological properties. As we suggested before, these new similarity metrics or embedding methods can also take changing rate into consideration when it comes to measuring the system stability. That way, one can capture whether there is an over aggregation problem that emerges during aggregation process rather than capturing it by interpreting the results. Third, the experiments can be augmented by using different data sets from different domains. We have mentioned quite a number of studies which use different data sets in their experiments. Those data sets can be use to expand this work if they are available to work on. Moreover, generating scale free networks using Barabasi and Albert Model algorithm can be employed to expand

data sets pool to augment the experiments even further. In (Sulo et al., 2010), the authors have used Barabasi and Albert Model to generate random graph to create a ground truth which they know the changing rate of the networks. A similar work can be done in the future works as well.

REFERENCES

Aggarwal, C., & Subbian, K. (2014). Evolutionary network analysis: A survey. *ACM Computing Surveys*, *47*(1), 10:1–10:36.

Albert, R., & Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, *74*(1), 47–97.
URL http://dx.doi.org/10.1103/RevModPhys.74.47

Aledavood, T., López, E., Roberts, S. G. B., Reed-Tsochas, F., Moro, E., Dunbar, R. I. M., & Saramäki, J. (2015). Daily rhythms in mobile telephone communication. *PLOS ONE*, *10*(9), 1–14.
URL https://doi.org/10.1371/journal.pone.0138098

Berger-Wolf, T. Y., & Saia, J. (2006). A framework for analysis of dynamic social networks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, (pp. 523–528). New York, NY, USA: ACM.

Blonder, B., Wey, T. W., Dornhaus, A., James, R., & Sih, A. (2012). Temporal dynamics and network analysis. *Methods in Ecology and Evolution*, *3*(6), 958–972.

Caceres, R. S., Berger-Wolf, T., & Grossman, R. (2011). Temporal scale of processes in dynamic networks. In *2011 IEEE 11th International Conference on Data Mining Workshops*, (pp. 925–932).

Clauset, A., & Eagle, N. (2012). Persistence and periodicity in a dynamic proximity network. *CoRR*, *abs/1211.7343*.

Dakiche, N., Tayeb, F. B., Slimani, Y., & Benatchba, K. (2018). Sensitive analysis of timeframe type and size impact on community evolution prediction. In *2018 IEEE International Conference on Fuzzy Systems*, (pp. 1–8).

Donnat, C., & Holmes, S. (2018). Tracking network dynamics: A survey using graph distances. *The Annals of Applied Statistics*, *12*(2), 971 – 1012.
URL https://doi.org/10.1214/18-AOAS1176

Dorogovtsev, S. N., & Mendes, J. F. F. (2002). Evolution of networks. *Advances in Physics*, *51*(4), 1079–1187.
URL http://dx.doi.org/10.1080/00018730110112519

Eagle, N., Pentland, A. S., & Lazer, D. (2009). Inferring social network structure using mobile phone data. In *PROCEEDINGS OF NATIONAL ACADEMY OF SCIENCES*, (pp. 127–136).

Eagle, N., & (Sandy) Pentland, A. (2006). Reality mining: Sensing complex social systems. *Personal Ubiquitous Comput.*, *10*(4), 255–268.

Fish, B., & Caceres, R. S. (2015). Handling oversampling in dynamic networks using link prediction. In *Machine Learning and Knowledge Discovery in Databases*, (pp. 671–686). Cham: Springer International Publishing.

Fish, B., & Caceres, R. S. (2017). A supervised approach to time scale detection in dynamic networks. *CoRR*, *abs/1702.07752*.
URL http://arxiv.org/abs/1702.07752

Freeman, L. C. (1978). Centrality in social networks conceptual clarification. *Social Networks*, *1*(3), 215–239.
URL https://www.sciencedirect.com/science/article/pii/0378873378900217

Holme, P., & Saramäki, J. (2012). Temporal networks. *Physics Reports*, *519*(3), 97–125.

K. Darst, R., Granell, C., Arenas, A., Gomez, S., Saramäki, J., & Fortunato, S. (2016). Detection of timescales in evolving complex systems. *Scientific Reports*, *6*.

Kempe, D., Kleinberg, J., & Kumar, A. (2000). Connectivity and inference problems for temporal networks. In *Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing*, STOC '00, (pp. 504–513). New York, NY, USA: ACM.

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

Leskovec, J., Adamic, L. A., & Huberman, B. A. (2007). The dynamics of viral marketing. *ACM Trans. Web*, *1*(1).

Medo, M., Zeng, A., Zhang, Y., & Mariani, M. S. (2018). Optimal timescale of community detection in growing networks. *CoRR*, *abs/1809.04943*.

Newman, M. (2003). The structure and function of complex networks. *SIAM review*, *45*(2), 167–256.

Orman, G. K., Türe, N., Balcisoy, S., & Boz, H. A. (2021). Finding proper time intervals for dynamic network extraction. *Journal of Statistical Mechanics: Theory and Experiment*, *2021*(3), 033414.
URL https://doi.org/10.1088/1742-5468/abed45

Real, R., & Vargas, J. M. (1996). The probabilistic basis of jaccard's index of similarity. *Systematic Biology, JSTOR*, *45*(3), 380–385.
URL www.jstor.org/stable/2413572

Rocha, L. E. C., Masuda, N., & Holme, P. (2017). Sampling of temporal networks: Methods and biases. *Phys. Rev. E*, *96*, 052302.
URL https://link.aps.org/doi/10.1103/PhysRevE.96.052302

Rossi, R. A., Gallagher, B., Neville, J., & Henderson, K. (2013). Modeling dynamic behavior in large evolving graphs. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, (pp. 667–676). New York, NY, USA: ACM.

Soundarajan, S., Tamersoy, A., Khalil, E. B., Eliassi-Rad, T., Chau, D. H., Gallagher, B., & Roundy, K. (2016). Generating graph snapshots from streaming edge data. In *Proceedings of the 25th International Conference Companion on World Wide Web*, WWW '16 Companion, (pp. 109–110). Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee.
URL https://doi.org/10.1145/2872518.2889398

Spiliopoulou, M. (2011). Evolution in social networks: A survey. In *Social Network Data Analytics*, chap. 6, (pp. 149–175). Springer.

Sulo, R., Berger-Wolf, T., & Grossman, R. (2010). Meaningful selection of temporal resolution for dynamic networks. In *Proceedings of the Eighth Workshop on Mining and Learning with Graphs*, MLG '10, (pp. 127–136). New York, NY, USA: ACM.

Uddin, S., Choudhury, N., M. Farhad, S., & Rahman, M. (2017). The optimal window size for analyzing longitudinal networks. *Scientific Reports*, *7*.

Wills, P., & Meyer, F. (2019). Metrics for graph comparison: A practitioner's guide. *CoRR*.

BIOGRAPHICAL SKETCH

Serhat ÇOLAK was graduated from Pertevniyal Anatolian High School in 2009, he has studied Control and Automation Engineering at Istanbul Technical University. He graduated in 2014, and after that he enrolled in M.Sc. program in Computer Engineering department at Galatasaray University.

Outside of academics, he has been working as Android Developer for the last 3 years.

He published a conference paper for M.Sc. thesis, titled “Similarity Based Compression Ratio for Dynamic Network Modelling” supervised by Assist. Prof. Keziban Orman was presented at “IEEE 19th International Conference on Smart Technologies (EUROCON), 2021, Ukraine” conference.