UNIVERSITÁ DI PISA

Dipartimento di Informatica
Corso di Laurea Triennale in Informatica

TESI DI LAUREA

Temporal K-Motifs in Online Social Communities

RELATORI
Prof. Laura RICCI
Dott. Barbara GUIDI

Candidato
Giulia FOIS

ANNO ACCADEMICO 2017/2018

A tutti coloro
che mi sono stati vicini

Acknowledgements

Voglio ringraziare il fantastico gruppo di ricerca, composto dalla Prof.ssa Laura Ricci, dalla Dott.ssa Barbara Guidi, dal Dott. Andrea Michienzi e il Dott. Andrea De Salve. Mi hanno supportata in tutte le fasi di questo lavoro, a partire dalle prime ricerche del materiale, passando per la prima stesura fino ad arrivare ai più fini dettagli. La loro professionalità e disponibilità è stata vitale per la stesura di questa tesi.
Ringrazio la mia famiglia, che ha fatto enormi sacrifici per permettermi di trascorrere questo mio capitolo della vita a Pisa; ha gioito con me nei momenti felici e mi ha sempre dato la forza di andare avanti nei momenti difficili.
Ringrazio il mio fidanzato Carmine, che fino ad un anno e mezzo fa non faceva parte della mia vita, ma che me l'ha stravolta quando ci siamo conosciuti. Il suo amore e i suoi abbracci sono stati fondamentali per giungere a questo traguardo in modo sereno.
Ringrazio tutti i miei amici, la cui lista probabilmente prenderà tutta la pagina, perchè sono davvero tanti coloro che mi hanno accompagnata in questo percorso. Partendo da quelli di Genova, che mi hanno sempre accolta a braccia aperte quando tornavo, ringrazio Isabella, Beatrice, Bruk, Matteo, Teresa e Francesco. Prima di tornare a Pisa, faccio il giro largo e passo dalla Sicilia, magnifica regione da dove vengono quelli che con affetto chiamo *i Siciliani*: ringrazio Monica, Salvatore D.G., Salvatore F., Martina, Miriam e Claudio. Il mio giro panoramico dell'Italia si conclude a Pisa, dove ho avuto il piacere di condividere momenti di studio e soprattutto di risate con Laura, Francesco "Frank", Fabio, Francesco S., Alexandra, Gabriele "Uarisog", Edoardo, Eleonora, Sara F., Sara M., Giorgio, Daniele, Giulia, Maria Vittoria, Gabriele "Gersy", Marco Antonio, Elena, Laura R., Giorgia e Jessica.
Un ringraziamento speciale va alle due mie amiche "a distanza", distanza che non è mai stata mai percepita come un ostacolo, dato che l'hanno sempre saputa colmare. Ringrazio Elena da Madrid e Giulia da Perugia.
Grazie a tutti. Vi voglio bene.

Contents

Chapter 1

Introduction

Online Social Networks are cornerstones of today's communication. Their popularity is intrinsic in their usability and portability, since they can be nowadays accessed by most of the technological devices owned by the majority of people. Social Media like Facebook allow users to communicate with each other, establishing friendship bonds or just interacting with strangers.
This mechanism is particularly relevant in groups, smaller communities inside Social Networks. In groups people have the chance to share their ideas and reach out to a number of members that can vary from dozens to thousands of users; it is possible to discuss ideas, doubts or interests with such a big number of people that would be otherwise difficultly reachable in real life or with other traditional communication systems, that are mostly dedicated to private, end-to-end ways of messaging.

Such online communities can be modelled by the mathematical structure of graphs, that are much used in a lot of fields nowadays: Biology, Neurology, Information Technology are just some of the very large number of subjects that use graphs as models to represent complex structures like neural networks or, in fact, social communities. Social Networks, in particular, make use of temporal graphs. These kinds of graphs have the connotation not to be static structures with pre-determined sets of nodes and edges, but to dinamically evolve throughout time, being day after day enriched by new users and numerous interactions between existing ones.
Their fundamental additional feature with respect to traditional graphs is the fact that edges are labeled with temporal information, thus it is possible to reconstruct the ordered sequence of interactions that compose conversations. Users' behaviour is defined by the way they interact at a temporal level, other than what they share content-wise. Analyzing how people engage with others in online social contexts can

give us a picture of what the current behavioural tendencies are.
One possible path to follow is to investigate communication patterns, that are recurring ways of communicating in which users tend to interact. In literature, they are referred to as motifs, subgraphs that occur particularly frequently in networks [1]. They rely on the mathematical concept of graph isomorphism, and this makes trying to find them fall into the NP-complete class of problems. Motifs that can be found in temporal graphs are called temporal motifs. The peculiarity of temporal motifs is the additional temporal constraint that forces them to have the exact same order of interactions and to develop in definite time frames, called time windows.

In the state of the art, researchers have elaborated algorithms with the aim of finding temporal motifs made of sequences of interactions of pre-determined lengths. However, it is also interesting to investigate the matter in a parametric way, and see what size temporal motifs can reach. Counting how many instances of temporal motifs can be found in specific time windows and observing their evolution throughout time in terms of their length can reveal a lot about social networks' properties. Moreover, studying particular patterns can lead us to finding interesting figures in online social communities, such as, for example, influencers or spammers.

In this work we explore Facebook groups belonging to several categories. Our aim is to find what are the most frequent temporal motifs, and how much they expand in length. Some works in the state of art [2] [3] have analyzed Facebook datasets to find temporal motifs; our research specifically focuses on identifiying motifs in groups, that is a topic yet to be treated in literature. Users' behavioural tendencies in groups, where people can connect with strangers that have common interests or ideas, can be unique and deserve to be studied thoroughly. Furthermore, anoter innovative aspect is the parametric searching of the maximum lengths motifs can reach within group-specific amounts of time.
Works in literature [2] propose algorithms that may be efficient for a pre-determined set of temporal motifs with fixed lengths, but not flexible to find newly defined patterns. The results we found in relation to motifs' lengths are in some cases very interesting, and motifs' frequency in particular categories of groups is surprising.

This thesis is structured as follows. In **Chapter 2** we explain the mathematical background, defining the fundamental concepts that lead to the definitions of temporal graphs and temporal motifs. Then, we show the current state of the art regarding

this topic, summarising papers that tackled the problem by different points of view. In **Chapter 3** we give our formalization of some specific temporal motifs and give the problem our particolar imprint, placing great emphasis on how we parameterized our research. This allows us not to focus on temporal motifs of fixed lengths, but instead to be able to see and evaluate what the most frequent and bigger-sized communication patterns of this kind are present in our dataset, made up of groups that are centered on different topics; we then formulate and formalize our problem, and provide pseudocodes for the algorithms we wrote to solve it. The tools we made use of and some statistical concepts that came in handy to our analysis are described in **Chapter 4**. Finally, in **Chapter 5**, we expose the results found through our analysis, explaining them from different perspectives and giving graphical representations of them through plots.

Chapter 2

State of the Art

Abstract

In this chapter we provide the theoretical instruments that are the foundations of our research and analyze the state of the art for what concerns temporal motifs mining and their meaning in social networks, trying to stress the common features and the differences between them. We firstly introduce the mathematical background that is useful to fully comprehend the topics that are covered further on. Then, we go through the works related to motif mining, particularly concentrating on algorithms to do so, analyzing their functioning, performance and the specific temporal motifs they are capable of finding. Finally, we move our interest on works that use motif-related results to explain behavioural properties of online social communities.

2.1 Mathematical Background

In this section we present some mathematical definitions for the notions of *temporal graph* and *temporal window*, in order to be able to explain the concept of *temporal motif*, which is the core of our research.

2.1.1 Temporal Graphs

A graph G is a pair (V,E) of two sets: V is the set of vertices, that represent entities in this structure, and E is the set of edges, which represents relations between these objects. More specifically, an edge (v_i,v_j) is a link between two vertices; it can be directed or undirected. We thus talk of directed or undirected graphs. In this work we focus on a particular kind of graph, which is the *temporal graph*. First, though,

we need to define the concepts of time window and temporal edges, since they are used in the rest of the thesis.
A *time window* Δ_T is a temporal interval of length L_T. Given a fixed start time, t_{start}, the time window will thus be the interval $\Delta_T = \langle t_{start}, t_{start}+L_T \rangle$.
The peculiarity of temporal graphs is that edges represent interactions between nodes throughout time, and are thus extended with temporal information: for this reason they are called temporal edges. More in detail, a *temporal edge* e_i is a tuple $\langle s_i, d_i, t_i, \delta_i \rangle$ where $s_i, d_i \in V$ are respectively the source and the destination involved in the communication [3]. The temporal information is expressed through the attributes t_i and δ_i that are respectively the timestamp and the duration of the communication, such that the event connects the two nodes in the time interval which goes from t_i to $t_i + \delta_i$. Given the definitions of temporal edge and time window, a temporal graph T can thus be seen as a tuple $\langle V, E, \Delta_T \rangle$, where V is the set of vertices, E is a set of temporal edges and Δ_T is the temporal window of length L_T. The elements in E (also called *events*) $\{e_1,...,e_k\}$ hold the following properties [3]:
$\forall e_i \in E,\ 0 \leq i \leq k, \exists e_j \in E,\ j \neq i,$

1. $\{s_i, d_i\} \cap \{s_j, d_j\} \neq \emptyset$
2. $0 < t_i - t_j - \delta_j < L_T$ or $0 < t_j - t_i - \delta_i < L_T$

Other works base their definition of temporal graph on the concepts of Δ_T-*adjacency* [4] [5] and Δ_T-*connection* [4]. Kovanen et al. [4] call Δ_T-*adjacent* any couple of temporal edges e_i and e_j for which the properties 1 and 2 stated above are valid. Two edges are said to be Δ_T-*connected* if there exists a sequence of Δ_T-adjacent edges between them. A temporal graph, then, is a sequence of events, all Δ_T-connected between each other.
In Mellor's paper [5], instead, the Δ_T-*adjacency* property and his definition of temporal networks are used as the foundations for the concept of Δ_T-*Temporal Event Graph*. He defines a temporal network as a tuple $G = \langle V, E, T \rangle$. V is the set of nodes that interact with each other, T is a non empty and ordered set of interaction times and E is a set of temporal edges e_i of form $\langle s_i, d_i, t_i \rangle$. The source s_i and the destination d_i are nodes, and $t_i \in T$. He then proceeds and defines the Δ_T-Temporal Event Graph.
Given a temporal network G, the Δ_T-Temporal Event Graph, or Δ_T-*TEG*, is a directed graph $\mathscr{G} = \langle \mathscr{V}, \mathscr{E} \rangle$ where $\mathscr{V}$ coincides with the set of temporal edges of G and $\mathscr{E}$ is a subset of $\mathscr{V} \times \mathscr{V}$, and therefore contains elements of form $\langle (s_i, d_i), (s_j, d_j) \rangle$, and is built as follows. Given a vertex $\langle s_i, d_i \rangle$ of $\mathscr{V}$, let

$$S(s_i) = \{k|((s_i) \cap (s_k, d_k)\} \neq \emptyset) \wedge (0 < t_k - t_i < \Delta t)\}$$

be the set of Δ_T-adjacent nodes of s_i in V, and let $S(d_i)$ be the same set for d_i. $\mathscr{E}$ is then defined as

$$\{(e_i, e_j)|(j = min(S(s_i)) \vee (j = min(S(d_i))\},$$

which means that every element $\langle s_i, d_i \rangle$ in $\mathscr{V}$ is connected to the next Δ_T-adjacent event of s_i and d_i.

Another possible way of representing a graph that is enriched by temporal information is the Link Stream formalization [6]. A Link Stream is a graph $G = \langle T, V, E \rangle$, where T is a set of temporal labels that can be given to temporal edges representing their order in the timeline of the graph, V is the set of nodes and $E \subseteq V \times V$ is the set of temporal edges. Each element of the Link Stream has the form $\langle t, s_i, d_i \rangle$ meaning that the t-th event connects the nodes s_i and d_i. In **Chapter 3** we use this definition as the basis for the formalization of the temporal motifs that are significant to our research.

Having fixed the concept of temporal graph, we now move forward and present what a temporal motif is and its relevance in the complex network analysis, with a special focus on Online Social Networks.

2.1.2 Temporal Motifs

A *temporal motif* is a class of equivalent temporal subgraphs [1]. Subgraphs that belong to a specific class need to be temporally isomorphic, i.e. they need to satisfy topological equivalence and their events have to occur in the same order. This definition can be also enriched by the requirement that the subgraphs in question are valid, that is they have to include all the consecutive events of the nodes that were originally present in the graph. Equivalence classes of subgraphs that have this property are called *temporal induced motifs*. This additional requirement reduces a lot the complexity of the problem of counting motifs. In fact, as stated by Zhao and Khan [7], enumerating non-induced ones is currently a computationally difficult task, since it grows very rapidly with the input size, and so far has been limited to small-scale networks, much more restricted than real-world ones.

One possible way to describe the various types of temporal motifs is that proposed in Paranjape et al.'s definition [2], that summarizes the most essential information about them in the number of nodes and edges they have, as well as the duration in which all their events take place. We thus have a k-nodes, l-edges, δ_T temporal motif. It is a k-node temporal graph having l edges that are temporally ordered and δ_T as

its associated time window. Any valid instance of this motif will have timestamps associated to its edges, that need to satisfy the temporal constraints given by the motif itself and fall into the δ_T interval.

2.2 Algorithms to mine motifs

The papers we go through in this section aim to introduce algorithms that are more efficient or scalable compared to the state of the art. We describe their structure and operation, and then, at the end of this part, confront their performance.

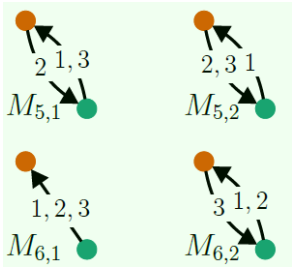


Figure 2.1: 2-nodes common motifs found by Paranjape et al. The bottom left indexing of each motif is due to the fact that these motifs, in Paranjape's paper, were organized in a bi-dimensional matrix.

Paranjape et al. [2] propose a specific framework that is optimal in the task of counting *3*-nodes, *3*-edge star and triangle motifs. They start by showing a more general algorithm, that is optimal for counting *2*-node motifs (linear in the edges' space) but inefficient even on *3*-node ones, like triangles, or generally on k-node ones, where $k > 3$.
The faster algorithm, instead, takes up a dynamic programming approach that takes advantage of the position of an edge inside the motif, by pinpointing three classes of counters according on the role of the edge as first, second or third in the motif. It takes a sequence of temporal edges and the length of a time window Δ_T as its input, and returns the number of motifs counted. It makes use of three subroutines, that are implemented differently depending on whether the algorithm is meant to count star or triangle motifs. This is a particularly relevant downside of the algorithm, since it is not possible to decide which motif to count, nor its dimension. Its efficiency is given by the fact that the choice of motifs taken into account is very limited.
The authors tested the faster algorithm on a multitude of datasets belonging to different contexts, for example emails, phone calls, Stack Overflow, the Bitcoin network and Facebook. We specifically concentrate on the latter because of its significance

and affinity with this work. Their Facebook dataset contains nearly 900.000 wall posts of over 45.000 users. Every edge $\langle v_i, v_j, t \rangle$ represents a post on v_j's wall made by v_i at time t. Differently from our research, they only take in consideration wall posts, ignoring comments and replies. They found that, among all the motifs of 2 and 3 nodes, the former are definitely more common. In fact, given a time window Δ_T of one hour, they counted almost half million *2*-nodes, *3*-edges cycles and ping-pongs against a peak of 35.000 *3*-nodes motifs. Figure 2.1 shows the motifs in question. Moreover, they distinguish motifs in relation to a property of their interactions. In motifs of the so-called blocking type, in fact, an user, before communicating with another one, typically waits for a response from an individual he interacted with beforehand. This strict condition doesn't apply to temporal motifs of the non-blocking type. In this paper, the authors show through their results how motifs of the blocking type tend to prevail: this means that users generally wait for their posts' recipient to respond before moving on and communicate with other friends.

Another algorithm, called COMMIT, has been developed and explained step by step in Gurukar et al. [8]. Differently from the previous algorithm, it can count all the temporal motifs that are present in the network that have a greater support than a certain threshold τ. The *support* of a motif is defined as the number of its embeddings in the temporal graph. It indeed takes as its input the just mentioned threshold and the length of a time window Δ_T.
COMMIT can be seen as a pipeline of different activities. First, it identifies all the temporal connected components in the network, i.e. the subgraphs of the network of biggest size that are temporally connected. Then it uses the concept of graph invariant to outdo the NP-complexity of the isomorphism problem, by mapping all the temporal connected subgraphs into a degree sequence of edges, where each edge is transformed in the pair of degrees of its source and destination nodes. An edge e_i is mapped into the couple (l(s_i),l(d_i)), where l(v) is the degree of the vertex v. A subgraph, thus, can be seen as a sequence of pair of degrees, which is indeed a degree sequence. Being the latter a graph invariant, two isomorphic graphs will have the same string of degrees. Then the third step consists in the so-called frequent subsequence mining phase, where, taken the collection of subsequences, the algorithm calculates the support of each of them, which is the number of their embeddings in the graph. It starts by counting the support of subsequences of one edge, and keeps on extending them with other edges (using a sub-procedure to discover the best candidate edges to do the extension) until a so-formed subsequence becomes not frequent: doing so

it identifies the most frequent subsequences. Eventually, COMMIT returns to the graph space in order to identify the subgraphs (i.e. the motifs) that correspond to the latter, and returns them with their frequencies.
To evaluate its scalability, COMMIT's performance has been compared to Naïve, a brute-force algorithm that enumerates all the possible subgraphs, and GRAMI [9], which represented the state of the art. They based the analysis of COMMIT's running time on the growth of three different parameters: the size of the time window Δ_T, k, which is the number of the most frequent motifs to be found, and the minimum support per-motif τ. The former depended on the specific case of application, since tests have been run on two big datasets derived from Facebook and Twitter, whose interactions are of different nature. Thus, it varied between the order of seconds (for Twitter) and hours (for Facebook). The second and the third parameters, on the other hand, were totally independent of the social networks. COMMIT doesn't present the limitations of the other two algorithms: Naïve, in fact, runs out of memory while mining motifs with size bigger than 4 because of its poor efficiency, while GRAMI fails to complete after several hours since it is only able to prune the search space on graphs with node labels, that were not present in COMMIT's authors test settings. In all cases, COMMIT's running time is significantly lower than the other algorithms', and it scales linearly on the growth of the parameters mentioned earlier.
Given the dataset on which this work is based, we consider particularly relevant the results obtained by running COMMIT on Facebook. Gurukar et al. [8] mined motifs on a temporal graph representing Facebook's wall posts between users; a directed edge from the vertex n_i to the vertex n_j, in this case, means that the user associated to the first node has posted on the second users' wall, jsut like what we said about Paranjape et al.'s analysis. As we can see in **Figure 2.2**, the top-3 motifs found by the algorithm fall into the *2*, *3* or *4*-nodes, *3*-edge category. The main tendency is for one user to repeatedly post on a friend's wall, which is a half ping-pong motif, that we later in this work define as *one-way couple*. This is followed by two other common behaviors that can be associated to the star temporal motif. In one case three different users all post on another users' wall: this situation can be easily linked to a person's birthday, when multiple friends write on his Facebook wall to wish him. The second behaviour corresponds to a situation where an user firstly posts once on a friend's wall, then twice on another's. Ultimately, these three motifs show that, on Facebook, people tend to interact with a restricted group of friends at a time.

Finally, Romijn, Nualláin and Torenvliet [10] implement another peculiar method to count motifs in their work; it is the Color-coding technique. It was firstly intro-

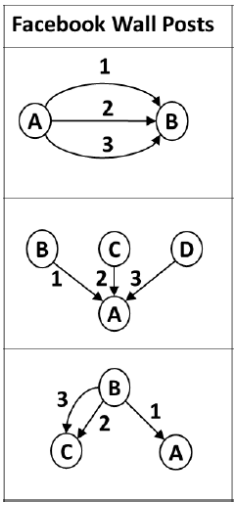


Figure 2.2: The top 3 motifs found by the COMMIT algorithm in a dataset of Facebook wall posts

duced by Alon, Yuster and Zwick [11], transformed in an algorithm by Gonen and Shavitt [12] and then used by Zhao et al. [7] in a parallel version to solve determinate cases of the subgraph isomorphism problem in polynomial time, which is a big conquest given the NP-completeness of the problem itself. Given a predetermined set of colors, the technique is based on a random color assignment to the nodes of a graph to find colorful paths or cycles. The term *colorful* means that each node needs to have a different color compared to all the others in the same path. The algorithm implemented by Romijn et al. [10] , called *Simple Path*, takes as its input a graph G, a vertex v and an integer l, which is the length of the paths it searches. In the main routine it has two nested loops. The internal one randomly colors the graph at each iteration, and uses a counter P_i to keep track of the number of colored paths (of length l) that contain v or are adjacent to it. The external one, on the other hand, calculates the mean Y_j of the values contained in the counters P_i, for each i. At the end, the algorithm returns the median of all the Y_j times the probability that a path of length l is colorful, i.e. the expected value of the number of colorful paths. This method has been exploited to count k-length paths in static graphs, and the process has also been conducted on a random, artificially created networks with the same nodes and edges sets' size, to notice the wider probability distribution of the paths in the nodes of the real network. The complex network on which they mainly conducted their experiments was an Irish community dataset, with over 36 million posts. A post consists in a reply to a thread, i.e. a specific discussion posted on a

forum of the website. The graph corresponding to the community is made by nodes that represent all the users that have an account and edges that connect two users *iff* they posted on the same thread. Since the Simple Path algorithm doesn't take into account the temporal information of the events, the results shown in the paper are not particularly relevant for this thesis.

All the algorithms confronted in this section are particularly efficient in solving the problems they tackle. In fact, the experiments made with COMMIT show that it takes up less than two hours for mining all motifs on large datasets. The Simple Path algorithm, being an approximate but accurate technique to deal with a NP-problem, grows linearly in the number of edges of the graphs it takes as its input. Finally, the algorithms designed by Paranjape et al. [2] are optimal as well, having a linear complexity on input size, but just for the restricted number of motif classes explained before. This is indeed their biggest limitation, that is not present in the other two. COMMIT, in fact, has the ability to support top-k queries, i.e. to identify the top-k frequent subgraphs, and thus is able to count the support for all the frequent motifs of all lengths in a network. On the other hand, Simple Path is too able to count the number of subgraphs of a certain dimension embedded in a graph, but doesn't consider at all the temporal information, and therefore is not the best choice for our aim.

2.3 Motifs to explain networks' properties

In this section we consider papers which, rather than going into detailed descriptions about the algorithms used to count motifs, show statistics that reveal characteristics about the motifs themselves and properties about the datasets employed, and consequently about the networks they examine.

Zhao et al.'s paper [3] is one of those. They extrapolate motifs from two datasets derived from CDR (i.e. *Call Detail Records* collected from a western city) and Facebook. The first contains more than 17,000,000 interactions, while the second one is a set made of 800,000 wall posts between the social network users. The authors find the most common motifs using a time window W of 4 hours. As shown in **Figure 2.3**, which is a table containing the frequencies of the top-6 motifs in the two datasets, the 2-chain and the 2-star are both in the first two positions of the ranking. Except for the 4-star, which is only listed in the Facebook column, the other motifs illustrated in the picture are the same, besides the different positioning. Many patterns are thus

shared by the two different networks, and this means that human communication behavior doesn't change that much, even in two such different contexts. They compare the most frequent ones, to find that many patterns are common in both the two different networks, showing that human communication behavior doesn't change a lot, even in such different contexts. Split the time window Δ_T in several smaller intervals δ_i, they also show how the support of each motif remains almost constant in the various δ_i, meaning that the communication patterns maintain the same frequency over time.

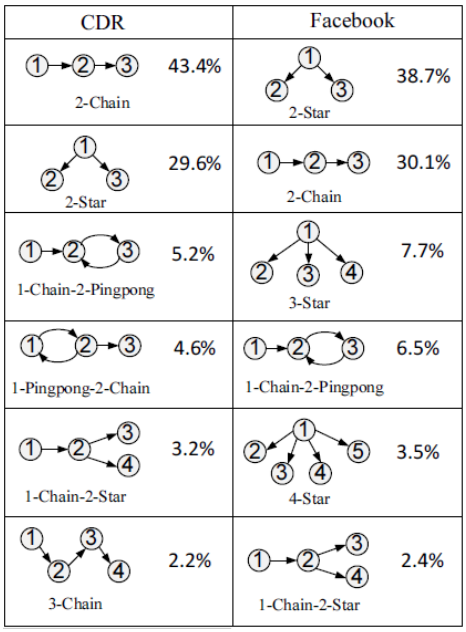


Figure 2.3: The most frequent motifs found in CDR and Facebook by Zhao et al.

Kovanen et al. [1] take advantage of the null model concept to show how factors like sex and age may have an influence on communication patterns. This property is called temporal homophily, i.e. the tendency of individuals with the same attributes to participate in the same communication patterns. In their paper, in addition to a dataset of mobile phone calls (represented by a network with nodes that have three attributes: gender, age interval and payment type), they generate synthetic data, shuffling the node types or the timing of the edges by applying a permutation function on the graph. Then they calculate the support of a determinate motif in the original network and in the fake one, and use those two data to find and check the values of the z-scores. Since the latter are broadly higher than 0 for the 35% of the motifs, the authors conclude that the null hypothesis is false, that is motifs' frequency truly depends on nodes' attributes.

Chapter 3

Temporal K-Motifs in Facebook Groups

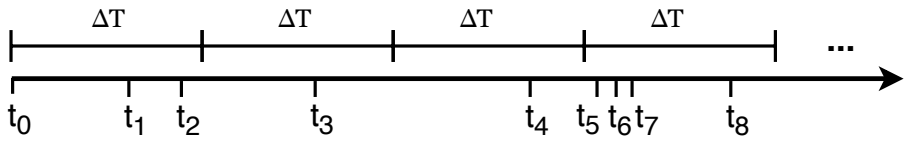
Abstract

In the previous chapters we went through important definitions to frame the matter this work focuses on, and then examined some related papers to learn methodologies and results found about it. We now give a brief explanation of our goals and how we want to achieve them. We describe in detail the structure of the two kinds of graphs our work is based on, that are the Post Graph and the User Graph. Then we describe and formalize the central temporal motifs for our research. Moreover, we provide pseudocode for the algorithms we used to count the temporal motifs.

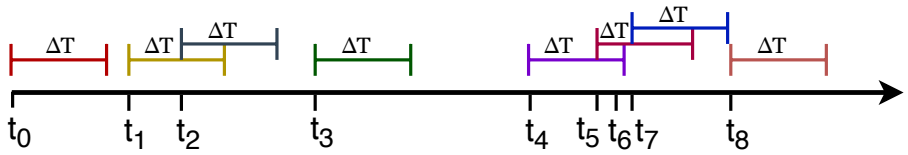
3.1 Description of the problem

The main focus of this thesis is to mine some significant kinds of temporal motifs in different Facebook groups. We want to see how frequent they are, in order to try to explain what are the most common communication patterns inside each group and what they mean by considering the time factor.
The definition of the motifs is based on the temporal graph formulation. Hence, the temporal aspect is tied to the presence of a temporal window Δ_T, that we previously defined in **Chapter 2**. The first approach we adopted, that we called the *snapshots* approach, was to divide the entire group timeline in consecutive time frames, each one of the same exact length as the time window Δ_T, and then working on those graph portions separately to find the motifs. This approach was quite simplistic: we

expected this adoption of time discretization to make us lose some interactions between snapshots, and thus to make us find motifs that cross two or more snapshots shorter than their real length. For this reason we decided to also pursue a second approach, that is the *time window sliding* approach. As its name suggests, it consists in sliding the time window over the events of our dataset: given a sequence of events $\{e_1, ...e_k\}$ having timestamps $t_1, t_2, ..., t_k$, we counted the motifs that lied in the interval $(t_1, t_1 + \Delta_T)$, the ones that lied in $(t_2, t_2 + \Delta_T)$ and so forth. Even though this way of proceeding was more sophisticated and surely let us find overall longer motifs compared to the other approach, it was also highly dependent on the interactions' density throughout time, and the extents of the time window shifts were very heterogeneous. In particularly active parts of a group interactions were seconds apart, while in others, where the events' frequency had gone off, they could be several minutes apart, in some cases hours. Both ways of treating the time window have their advantages and disadvantages: we decided to adopt the two approaches separately and confront the results obtained by using them. We will see later on in this work that the results fulfilled our expectations regarding the length of the temporal motifs. **Figure 3.1a** and **Figure 3.1b** are illustrative examples of the two time window approaches. The black bold arrow is the group timeline, and the ticks $t_0, ..., t_8$ are timestamps of eight specific and distinct temporal edges. As we can see from the figures, in the first case time is equally divided in portions as long as Δ_T, while in the second example the time window slides over the group timeline taking as its start each event, sometimes comprehending other interactions in its duration.



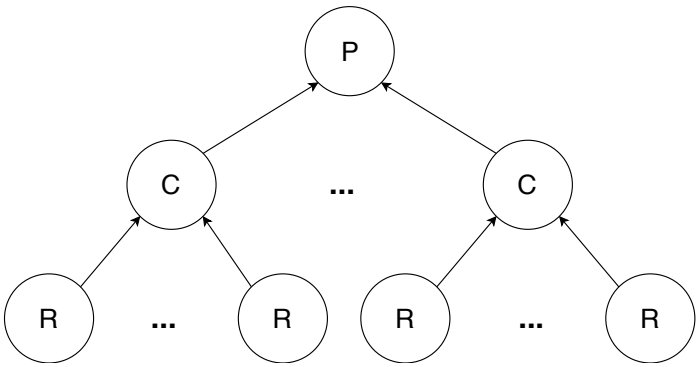
(a) A representation of the snapshots approach



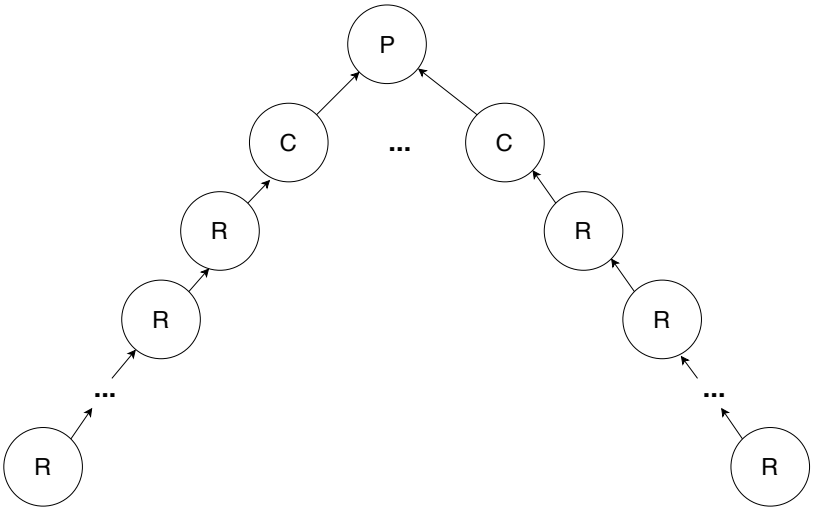
(b) A representation of the time window sliding approach

Figure 3.1

In order to give a defined structure to the Facebook groups in our dataset, we associated two kinds of graphs to each one of them, and we subsequently used both of them to find our temporal motifs from different perspectives. The first one is the Post Graph, where the focus is on the type of interaction that can happen in a group. In this kind of graph, the temporal k-motifs are mined in each post's scope, making our research of it post-centered. Searching for temporal motifs using this formalization is useful to comprehend how activity is structured under a specific post in a group.
A *Post Graph* is a tuple $\langle V, E, \Delta_T \rangle$, and is composed by as many separate subgraphs as the number of posts of the group it models. V, the set of vertices, is a set of couples $\langle u_i, a_i \rangle$, where u_i is one of the users of the group and a_i is an activity of his in a post. E contains directed temporal edges $\langle s_i, d_i, t_i \rangle$. It is important to notice how our temporal edges don't have any duration information δ_i, which makes our definition simpler than what stated in **Chapter 2**. Interactions, in this case, are immediate: the temporal aspect is only given by their timestamp. This is deductible from the fact that Facebook itself is composed by immediate interactions that only have timestamps, in relation to the moments in which they happen. Lastly, Δ_T is a temporal window, that we utilise, using one of the two approaches stated before, as a time frame to find our temporal motifs in the graph. The peculiarity of this graph is the fact that each node has a role. An user can be the one who writes a post, one who comments a post or one who replies to a comment. These are the fundamental parts that altogether contribute to the overall activity of the group. The definition of the edges is inherently bound to these roles. In fact, the edge $\langle s_i, d_i, t_i \rangle$ can only be interpreted either as *"s_i's comment under d_i's post at time t_i"* or *"s_i's reply under d_i's comment at time t_i"*. The nature of the temporal edge (comment or reply) is given by the roles of the nodes that take part to the edge.



(a) The 3-level tree representation of a Post Graph: replies, in this context, are seen as interactions only targeting comments



(b) The n-level tree representation of a Post Graph, where replies only have other replies as their targets, forming a chain under the original comment

Figure 3.2

The second kind of graph we used is the *User Graph*. This formalization is useful to look at a group's activity more broadly, focusing not on the type of interactions but on the users that generate it. This way we can find user-centered temporal motifs. A User Graph is a tuple $\langle V, E, \Delta_T \rangle$ where the set of vertices V is simply a subset of the users of the group and Δ_T is a temporal window. The set of edges E contains generic interactions between users, irrespective of the post under which they happen. If $\langle s_i, d_i, t_i \rangle = e \in E$, it means that s_i has interacted with d_i at time t_i with a semantics (comment or reply) that isn't relevant for us in this context. d_i could be then connected to another user for having interacted with him under the same post or a totally different one. Since two users can interact more than once in the scope

of the same time window, and thus their representing vertices can be connected by more than two edges, the User Graph is mathematically defined as a multigraph.
Figure 3.2a shows the general structure of a discussion under a post. It is a *3-level* tree where the root is the post's author, the second-level nodes are users who comment and the third-level ones are the users who reply to the comments. Since our information on replies' targets are ambiguous (a reply can either be directed to a comment or to another reply: Facebook doesn't distinguish one case from the other) the structure can also be seen as a *n-level* tree, where $n-2$ is the number of users that reply to the comment with the highest number of replies. This tree tends to be unbalanced towards the comments that generate more activity. This alternative vision is represented in **Figure 3.2b**. The Post Graph of a group with j posts is composed by j tree subgraphs, To count our temporal motifs, we decided to consider alternatively both the possible structures, depending on the motif to search for.
An example of a possible representation of the User Graph is given in **Figure 3.3**, where we can clearly see its multigraph topology. It is important to underline a fundamental fact about User Graphs. As we previously said, it is impossible to decide whether a reply targets a comment or is directed to another reply. In the Post Graph, when we encounter a reply, we treat it differently in accordance with the motif we are looking for. In the User Graph, however, we do not discern the type of interaction. Hence a reply is interpreted as an interaction between its author and the author of the comment it targets. What happens, then, when an user replies to its own comment, maybe triggered by another reply? In the Post Graph we can easily bypass the problem by considering the reply as an interaction which targets another reply. In the User Graph this is not possible, precisely because we do not take into consideration the nature of the events. Such interaction is inevitably viewed as one between a determined user and itself. For this reason, some intra post events are lost in the User Graph. In **Chapter 5** we see at a practical level how this affects the maximum length of some motifs.

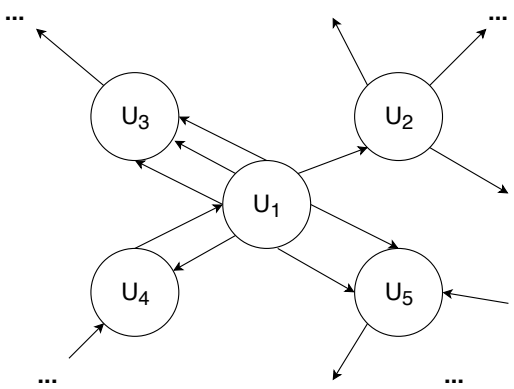


Figure 3.3: Representation of the User Graph

3.2 K-Motifs

We now define the temporal motifs in **Figure 3.4**, using the Link Stream formalism [6] we presented in **Chapter 2**. For each motif, we provide a description in the context of Facebook. A temporal motif is seen, in this context, as an isomorphism class of Link Stream graphs. Differently from many other works on this matter, that focused on temporal motifs with specific lengths, we decided to extend our research and to mine motifs with parametric length k. A parameterized research allowed us to find the maximum length of the motifs, that in some cases was impressive and unexpected. Moreover, this way we were able to produce statistics about the number of temporal motifs with different lengths.
Because of the relevance with this work, we chose some of the temporal motifs that in literature [3] are declared to be the most frequent ones in the Facebook network. In the following lines we give a structural description and a formalization of each motif.

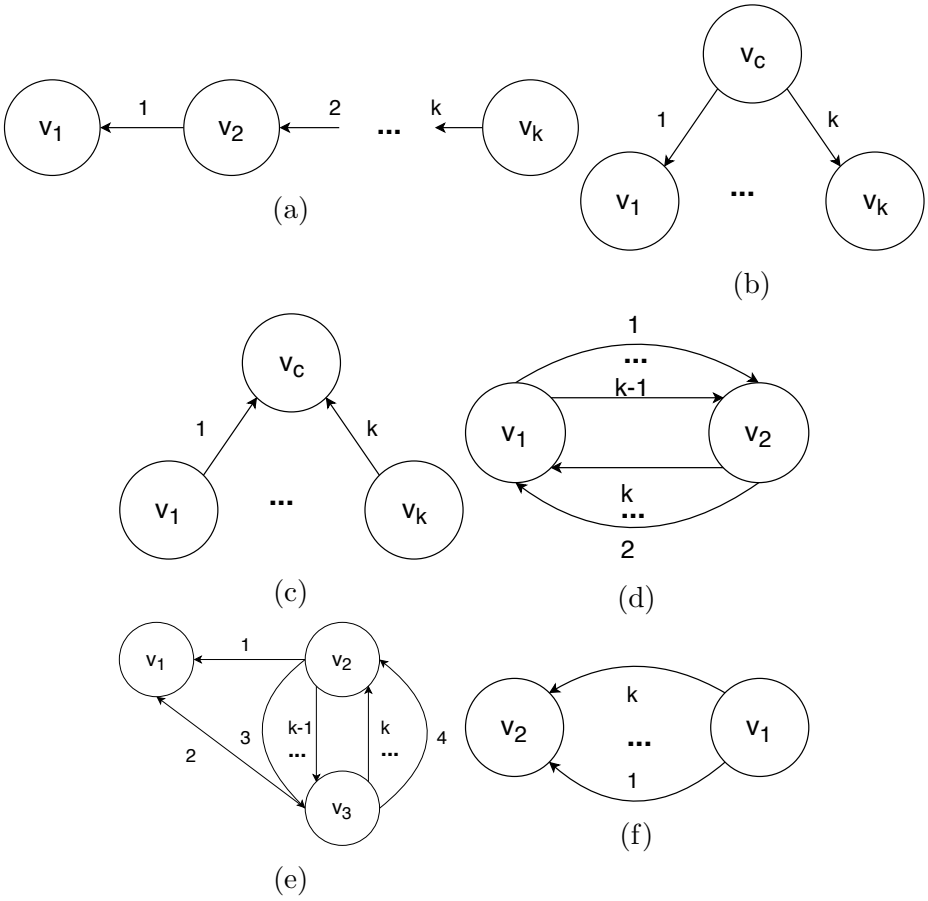


Figure 3.4: Significant motifs in Facebook groups

The first temporal motif, in **Figure 3.4a**, is the *chain*. It is composed by a set of nodes $\{v_1, ..., v_k\}$ and a list of directed temporal edges where the source of each edge corresponds to the destination of the previous edge and every couple of adjacent edges is also Δ_T-adjacent. Every vertex has exactly one ingoing edge and one outgoing one, with the exception of v_1, which only has an ingoing one, and v_k, that only has an outgoing edge. Moreover, its length t needs to be greater or equal to 3.
The chain motif represents a situation where the information is generated by the contribution of different users, that one after the other interact between themselves ending up to form what looks like an assembly line. In the Post Graph there are three possible patterns for the chain: it can be originated by a post, followed by a comment and by several replies, or it can start from a comment and continue with some replies, or be just composed by replies. As we said earlier, in the User Graph this connotation we give to events is not relevant. In this case the chain is simply made up of users that interact in a chain fashion in different contexts.

The chain can be expressed, with the LS formalism, as:

$$C = (T_C, V_C, E_C)$$

$$T_C = \{t \in \mathbb{N} \mid t \leq k\}$$

$$V_C = \{v_i \mid i \leq k+1 \wedge i \in \mathbb{N}\}$$

$$E_C = \{(i, v_{i+1}, v_i) \mid i \in T_C\}$$

The star is a temporal motif composed by a central node v_c, k peripherical nodes $\{v_1, ..., v_k\}$ and k-1 temporal edges. The edges all have the same direction, that is outgoing from the central node or ingoing in it. Depending on the role of the central node, that can either be the source or the destination of all the temporal edges, this motif can be in the form of an out star or an in star. **Figure 3.4b** and **Figure 3.4c** depict them.
In a Facebook context, an out-star motif describes a situation where an user spreads information towards different people. In order to be meaningful, its length needs to be greater or equal to 3.
The typical behaviour that may come to mind is the one of spammers. Spammers are users that want to disseminate a default message and typically act like machines, rapidly copying and pasting the same text with the aim of catching attention.
Both in the Post Graph and the User Graph, to generate an out-star motif, an user may comment different posts or reply to different comments in a short span of time. Considering our formal definition of the Post Graph, this motif is border-line, because the activity can be obviously scattered around different posts. Since the spamming attivity is better framed in a User Graph context, we decided to make an exception for what concerns its research in the Post Graph. In fact, we admitted its interactions to be scattered over multiple posts. The only difference is the fact that in the Post Graph we force the interactions of a determined out-star to be just comments or just replies, whereas in the User Graph there is no such constraint.
With the LS formalism, the out star can be formalized as follows:

$$OS = (T_{OS}, V_{OS}, E_{OS})$$

$$T_{OS} = \{t \in \mathbb{N} \mid t \leq k\}$$

$$V_{OS} = \{v_i \mid i \in T_{IS}\} \cup \{v_c\}$$

$$E_{OS} = \{(i, v_c, v_i) \mid i \in T_{OS}\}$$

The in-star motif is the opposite compared to the one just described. It portrays a situation where a user becomes the center of attention in a short amount of time. The considerations made about the out-star length are valid in this situation as well. Moreover, it is fundamental that a certain condition is fulfilled: that is the fact that all the authors of comments or replies that target the center user need to be different. The most common figure that frequently generates this kind of situation is the influencer. It is a very popular person in a social network and, narrowed to our situation, inside a group. Its name is known by most of the members of it and so is more prone to generating high responsiveness. Members of the group are led to commenting to his/her post or replying to his/her comments rather than doing so for people they don't know.
In the Post Graph this motif is one of the easiest to find, since it just requires the counting the comments under one post or the replies under one comment. Searching for an in-star in the User Graph, instead, might be more interesting. In fact, it allows us to picture the activity generated in the entire group, and in a short amount of time, by one single person.
The in star can be expressed as:

$$IS = (T_{IS}, V_{IS}, E_{IS})$$

$$T_{IS} = \{t \in \mathbb{N} \mid t \leq k\}$$

$$V_{IS} = \{v_i \mid i \in T_{IS}\} \cup \{v_c\}$$

$$E_{IS} = \{(i, v_i, v_c) \mid i \in T_{IS}\}$$

The temporal motif in **Figure 3.4d** is the ping pong. It is made up of two nodes, $\{v_1, v_2\}$, interacting with each other back and forth k times. In fact, for our list of directed temporal edges, we have the condition that if an edge has the form $e_i = \langle t_i, v_1, v_2 \rangle$, then the following one will be $e_{i+1} = \langle t_{i+1}, v_2, v_1 \rangle$. Moreover, in order for it to be a complete and significant ping pong, k must be an even number greater than 2. A ping pong represents a situation in which two users repeatedly communicate back and forth. A simple example of a ping pong could be a situation where two users start a lively discussion on a topic, and keep on debating about it. This is important to understand how frequently couple of users interact in small time windows. This means a debate between two users which can generate a general discussion. In the

Post Graph the situation is narrowed to a single post. In this case the motif may start with User A's post, proceed with User B's comment and then end with an alternate repartee of replies from both of them. It could also start with User A's comment and proceed with an alternation of replies.
In the User Graph, instead, neither the type of interaction or the specific post are taken into account. The two users might simply interact under one post and then rapidly move on and create another discussion under another one.
The LS formalization for the ping pong is:

$$PP = (T_{PP}, V_{PP}, E_{PP})$$

$$T_{PP} = \{t \in \mathbb{N} \mid t \leq k\}$$

$$V_{PP} = \{v_1, v_2\}$$

$$E_{PP} = \{(t, v_1, v_2) \mid t \in T_{PP} \wedge t\%2 == 1\} \cup \{(t, v_2, v_1) \mid t \in T_{PP} \wedge t\%2 == 0\}$$

In **Figure 3.4e** we have the p-triangle, that we called like this because it can be seen as a composition of a triangle and a ping pong. It is a motif yet to be treated in literature, and can be considered as a ping pong between two users, triggered by two previous interactions of them with a third user. The triangle shape is indeed given by the presence of three users. When the ping pong starts between two users, the third user just mentioned stops to be relevant. The p-triangle is thus composed by just three nodes $\{v_1, v_2, v_3\}$, and k+2 edges. Of those k, two have their source in v_2 and v_3 respectively and their destination in v_1, and are not counted in the overall length k of the motif. The remaining k edges, that are instead determinative to the length, are ping pong exchanges between v_2 and v_3. We considered this motif fitting best just in the Post Graph context, because the ping-pong motif arises from two previous interactions presumably in the scope of the same discussion under a post. A p-triangle situation would be of two users, B and C, interacting with the third one, A, with a comment or a reply (we want to stress the fact that we are in a Post Graph context, and so the type of interaction is meaningful). Then B and C, having seen each other's contribute to what A said, engage a lively discussion generating a ping pong. A common situation in which this motif may take place is one where two people give opposite opinions on a topic some other person initiate, and start arguing about their different point of views on the subject.

We can formalize a p-triangle of length k, with the LS formalism, as:

$$PTR = (T_{PTR}, V_{PTR}, E_{PTR})$$

$$T_{PTR} = \{t \in \mathbb{N} \mid t \leq k + 2\}$$

$$V_{PTR} = \{v_1, v_2, v_3\}$$

$$E_{PTR} = \{(t_1, v_2, v_1), (t_2, v_3, v_1)\}$$

$$\cup$$

$$\{(t, v_2, v_3) \mid t \in T_{PTR} \wedge t > 2 \wedge t\%2 == 1\}$$

$$\cup$$

$$\{(t, v_3, v_2) \mid t \in T_{PTR} \wedge t > 2 \wedge t\%2 == 0\}$$

Finally, **Figure 3.4f** depicts a motif that is not frequently dealt with in literature, which is what we call a one-way couple. This simple temporal motif is composed by a couple of vertices, $\{v_1, v_2\}$, and k unilateral temporal edges. This means that all edges have the same source and destination, and the length is determined by how many repeated interactions of such form take place within the time window. We study this motif because it shows a rather common situation, where a person seems obsessed with another one and keeps interacting with her despite not getting any feedback. In the Post Graph the interactions occur in the form of many comments of the same person under a post, or many replies under a comment. The User Graph more shows a stalker-like situation: an user, in fact, can not only confine himself under a discussion, but get to "chase his prey" in the context of other posts as well. We set its minimum length to 3, otherwise it could just be composed by two casual interactions.
The one-way couple can be expressed, with the LS formalism, as:

$$OWC = (T_{OWC}, V_{OWC}, E_{OWC})$$

$$T_{OWC} = \{t \in \mathbb{N} \mid t \leq k\}$$

$$V_{OWC} = \{v_1, v_2\}$$

$$E_{OWC} = \{(t, v_1, v_2) \mid t \in T_{OWC}\}$$

3.3 K-Motifs Algorithms

Having defined the environment we want to extract the motifs from, we can now move on to the algorithms we used to mine our motifs in the two kinds of graphs that are associated to the groups. Each motif is searched in relation to specific and group-dependent time windows we look into further in this work. Since we work on a dataset derived from Facebook groups, the application of our temporal motifs' formalization is adjusted to the fact that we have timestamps for the interactions. Thus, the Link Stream set T of time sorting labels is, for our practical usage in the algorithms, composed by timestamps. Moreover, as we already explained, we do not search motifs with a specific length. Our mining is parametric, and thus lets us find motifs with length k, that may vary from the minimum possible measure (that is motif-dependent) to an arbitrary length. **Algorithm 1** describes how we stored the information collected regarding the number of motifs of a certain length. The procedure UpdateMotifCounter is invoked in all the algorithms provided below, after finding the occurrence of a specific motif and its length. It updates a counter that is represented by a three-level hash map, with the graph where the motif has been found, the motif itself and its length as input parameters. Those are also the items that respectively serve as keys for the three levels of the map. Indeed, the first level has the two graphs as keys, and the motifs of various lengths as values; the second level has the specific motifs as keys, and their different lengths as values; finally, the third level has the particular lengths found for a motif as keys and the number of motifs of that length as values. The last parameter of the procedure, *min*, is simply an indicator for what is the minimum meaningful length for the motif given. This way, only the counters of bigger size motifs are increased.

Algorithm 1 UpdateMotifCounter

Input: *motif* is the temporal motif that is being searched, *min* is the minimum significant length for that motif, *length* is the length just found for that motif, *graph* is the type of graph (post or user) in which the motif has been searched

```
1: procedure UpdateMotifCounter(motif,min,length,graph)
2:     i ← length
3:     while i ≥ min do
4:         MotifCounter[graph][motif][i] ← MotifCounter[graph][motif][i] + 1
5:         i ← i − 1
```

For each motif, we provide the algorithms used to count its istances, one for each

kind of graph formulation. Since the research in the User Graph does not take account of the type of interaction, it is clear that the algorithms related to it are simpler. The Post Graph algorithms' names begin with the identifier *POST*, while the User Graph ones begin start with the keyword *USR*. It is also important to notice that we further divide the following subsections in two parts, in accordance to the two approaches used to treat the time window. For each motif, in fact, we present the algorithms based on the time window sliding approach and the ones based on the snapshots approach. The latter's names are identifiable by the keyword *SNAP*. The only differences in the two versions of the algorithms is the practical handling of the time window. In the algorithms related to the sliding approach, in fact, we proceed by scanning the entire list of the groups' interactions. For each temporal edge e_i, we consider as possible candidates for being a motif's component all the other temporal edges that fall into the timelapse that goes from t_i to $t_i + \Delta_T$. For what regards the other approach, instead, for each interaction encountered by the algorithm whe check the condition of being part of the current snapshot. If that is not the case, the temporal edge in question is cut out from the temporal motifs that start and end in that snapshot. Other than this simple but fundamental check, the algorithms related to the two approaches respectively do not differ. The instructions that are the core part for motifs' finding are the same, and the algorithms look specular.

3.3.1 Ping Pong Algorithms

The algorithms presented in this subsection are high-level pseudocode for the actual functions we used to find this motif. For each event, accordingly to its type, it finds interaction patterns shaped like described above. Then, if a ping pong is found and its length is greater than 2, the counter is updated. We only count an interaction if it's paired with another one, so the length of the ping pong is always an even number. The function in **Algorithm 2** is the core of this motif's mining, and is called by all the main algorithms related to it. It takes as its input a list of interactions and the couple of users we are currently seeking for a ping pong of, and returns the number of back and forth interactions between the two.

Algorithm 2

Input: *eventList* is a list of comments/replies, *authors* is a couple of user IDs

Output: *interactions* is the number of ping pong interactions between the two users

```
1: function CountInteractions(eventList, authors)
2:     searchedAuth ← 0, interactions ← 0
3:     for each e ∈ eventList do
4:         if e.author == authors[searchedAuth] then
5:             interactions ← interactions + 1
6:             searchedAuth ← (searchedAuth + 1) % 2
7:     return length
```

In **Algorithm 3** and **Algorithm 4** we show the algorithms used for counting ping pongs respectively in the Post Graph and in the User Graph, with the time window sliding approach. In the Post Graph, we consider the label of each temporal edge and according to that we run **Algorithm 2** to count the interactions between the source and the destination users of that edge. These interactions' labels need to respect the tree hierarchy of the Post Graph, so an event of the post type can only be followed by a comment and a sequence of replies; a comment can only be followed by a sequence of replies, and the same goes for a reply. In the User Graph we still count back and forth interactions between the source and a destination of each temporal edge, but we don't consider the type of the events, as anticipated.

Algorithm 3

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
1:  procedure POST_PINGPONGCOUNTER(G)
2:      for each e ∈ E do
3:          if e is a post then
4:              comList ← {c | c targets e ∧
                c.timestamp − e.timestamp ≤ Δ_T ∧ c ∈ E}
5:              length ← length + 1
6:              for each c ∈ comList do
7:                  authors ← (e.author, c.author)
8:                  repList ← {r | r targets c ∧
                r.timestamp − e.timestamp ≤ Δ_T ∧ r ∈ E}
9:                  interactNum ← COUNTINTERACTIONS(repList, authors)
10:                 if interactNum mod 2 ≠ 0 then
11:                     interactNum ← interactNum − 1
                                          ▷ The number of interactions must be even
12:                 UPDATEMOTIFCOUNTER(pingPong, 2, interactNum, postGraph)
13:         else if e is a comment then
14:             repList ← {r | r targets e ∧
                r.timestamp − e.timestamp ≤ Δ_T ∧ r ∈ E}
15:             for each r ∈ repList do
16:                 authors ← (e.author, r.author)
17:                 searchedAuth ← 0
18:                 interactNum ← COUNTINTERACTIONS(repList, authors)
19:                 if interactNum mod 2 ≠ 0 then
20:                     interactNum ← interactNum − 1
21:                 UPDATEMOTIFCOUNTER(pingPong, 2, interactNum, postGraph)
```

Algorithm 3 (cont'd)

```
22:     else if e is a reply then
23:         c ← e.target
24:         repList ← {r | r targets c ∧
            r.timestamp − e.timestamp ≤ Δ_T ∧ r ∈ E}
25:         for each r ∈ repList do
26:             authors ← (e.author, r.author)
27:             searchedAuth ← 0
28:             interactNum ← CountInteractions(repList, authors)
29:             if interactNum mod 2 ≠ 0 then
30:                 interactNum ← interactNum − 1
31:             UpdateMotifCounter(pingPong, 2, interactNum, postGraph)
```

Algorithm 4

Input: $G = (V, E, \Delta_T)$ is a temporal user graph

```
1: procedure Usr_PingPongCounter(G)
2:     for each e ∈ E do
3:         src ← e.source
4:         dest ← e.dest
5:         authors ← (src, dest)
6:         interactions ← {e_1 | e_1 ∈ E ∧
               e_1.timestamp − e.timestamp ≤ Δ_T}
7:         interactNum ← CountInteractions(repList, authors)
8:         if interactNum mod 2 ≠ 0 then
9:             interactNum ← interactNum − 1
10:        UpdateMotifCounter(pingPong, 2, interactNum, usrGraph)
```

The following algorithms, **Algorithm 5** and **Algorithm 6**, contain the pseudocode related to ping pong counting in the two types of graphs, with the snapshots approach.

Algorithm 5

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

11: **procedure** SNAP_POST_PINGPONGCOUNTER(G)
12: $startSnap \leftarrow E.get(0).timestamp$
13: $endSnap \leftarrow startSnap + \Delta_T$
14: **for each** $e \in E$ **do**
15: **if** $e.timestamp \geq endSnap$ **then**
16: $endSnap \leftarrow e.timestamp + \Delta_T$
17: **else**
18: **if** e is a post **then**
19: $comList \leftarrow \{c \mid$ c targets e $\wedge$ $c.timestamp \leq endSnap \wedge c \in E\}$
20: $length \leftarrow length + 1$
21: **for each** $c \in comList$ **do**
22: $authors \leftarrow (e.author, c.author)$
23: $repList \leftarrow \{r \mid$ r targets c $\wedge$ $r.timestamp \leq endSnap \wedge r \in E\}$
24: $interactNum \leftarrow$ COUNTINTERACTIONS$(repList, authors)$
25: **if** $interactNum \mod 2 \neq 0$ **then**
26: $interactNum \leftarrow interactNum - 1$
27: UPDATEMOTIFCOUNTER$(pingPong, 2, interactNum, postGraph)$
28: **else if** e is a comment **then**
29: $repList \leftarrow \{r \mid$ r targets e $\wedge$ $r.timestamp \leq endSnap \wedge r \in E\}$
30: **for each** $r \in repList$ **do**
31: $authors \leftarrow (e.author, r.author)$
32: $searchedAuth \leftarrow 0$
33: $interactNum \leftarrow$ COUNTINTERACTIONS$(repList, authors)$
34: **if** $interactNum \mod 2 \neq 0$ **then**
35: $interactNum \leftarrow interactNum - 1$
36: UPDATEMOTIFCOUNTER$(pingPong, 2, interactNum, postGraph)$

Algorithm 5 (cont'd)

```
37:         else if e is a reply then
38:             c ← e.target
39:             repList ← {r | r targets c ∧
            r.timestamp ≤ endSnap ∧ r ∈ E}
40:             for each r ∈ repList do
41:                 authors ← (e.author, r.author)
42:                 searchedAuth ← 0
43:                 interactNum ← CountInteractions(repList, authors)
44:                 if interactNum mod 2 ≠ 0 then
45:                     interactNum ← interactNum − 1
46:                 UpdateMotifCounter(pingPong, 2, interactNum, postGraph)
```

Algorithm 6

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
 1: procedure Snap_Usr_PingPongCounter(G)
 2:     startSnap ← E.get(0).timestamp
 3:     endSnap ← startSnap + Δ_T
 4:     for each e ∈ E do
 5:         if e.timestamp ≥ endSnap then
 6:             endSnap ← e.timestamp + Δ_T
 7:         else
 8:             src ← e.source
 9:             dest ← e.dest
10:             authors ← (src, dest)
11:             interactions ← {e_1 | e_1 ∈ E ∧
                e_1.timestamp ≤ endSnap}
12:             interactNum ← CountInteractions(repList, authors)
13:             if interactNum mod 2 ≠ 0 then
14:                 interactNum ← interactNum − 1
15:             UpdateMotifCounter(pingPong, 2, interactNum, usrGraph)
```

3.3.2 Out-Star Algorithms

The algorithms we used to count out-stars in the Post Graph and in the User Graph, by using the time window sliding approach, are respectively presented in **Algorithm 7** and **Algorithm 8**. For each temporal edge, we simply count how many other destination users are reached by the source node in the scope of the time window. In the Post Graph, as we previously said, we force the edges going outwards from the center vertex to be of the same type, whereas for the User Graph we do not have this requirement.

Algorithm 7

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

1: **procedure** POST_OUTSTARCOUNTER(G)
2: **for each** $e \in E$ **do**
3: **if** *e is a comment or e is a reply* **then**
4: $star \leftarrow \{e_1 \mid e_1 \in E \wedge e_1.type == e.type \wedge e_1.author == e.author \wedge e_1.timestamp - e.timestamp \leq \Delta_T\}$
5: Remove from *star* all the events with duplicate target
6: $starLen \leftarrow length(star)$
7: UPDATEMOTIFCOUNTER($outStar, 3, starLen, postGraph$)

Algorithm 8

Input: $G = (V, E, \Delta_T)$ is a temporal user graph

1: **procedure** USR_OUTSTARCOUNTER(G)
2: **for each** $e \in E$ **do**
3: $star \leftarrow \{e_1 \mid e_1 \in E \wedge e_1.author == e.author \wedge e_1.timestamp - e.timestamp \leq \Delta_T\}$
4: Remove from *star* all the events with duplicate target
5: $starLen \leftarrow length(star)$
6: UPDATEMOTIFCOUNTER($outStar, 3, starLen, usrGraph$)

In **Algorithm 9** and **Algorithm 10** we show the algorithms used to count out-stars in the Post Graph and in the User Graph using the snapshots approach.

Algorithm 9

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
1: procedure SNAP_POST_OUTSTARCOUNTER(G)
2:     startSnap ← E.get(0).timestamp
3:     endSnap ← startSnap + Δ_T
4:     for each e ∈ E do
5:         if e.timestamp ≥ endSnap then
6:             endSnap ← e.timestamp + Δ_T
7:         else
8:             if e is a comment or e is a reply then
9:                 star ← {e_1 | e_1 ∈ E ∧ e_1.type == e.type ∧ e_1.author == e.author
   ∧
                   e_1.timestamp ≤ endSnap}
10:                Remove from star all the events with duplicate target
11:                starLen ← length(star)
12:                UPDATEMOTIFCOUNTER(outStar, 3, starLen, postGraph)
```

Algorithm 10

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
1: procedure SNAP_USR_OUTSTARCOUNTER(G)
2:     startSnap ← E.get(0).timestamp
3:     endSnap ← startSnap + Δ_T
4:     for each e ∈ E do
5:         if e.timestamp ≥ endSnap then
6:             endSnap ← e.timestamp + Δ_T
7:         else
8:             star ← {e_1 | e_1 ∈ E ∧ e_1.author == e.author ∧
               e_1.timestamp ≤ endSnap}
9:             Remove from star all the events with duplicate target
10:            starLen ← length(star)
11:            UPDATEMOTIFCOUNTER(outStar, 3, starLen, usrGraph)
```

3.3.3 In-Star Algorithms

In **Algorithm 11** and **Algorithm 12** we show the algorithms used for detecting in-stars respectively in the Post Graph and in the User Graph, with the time window

sliding approach.We count, for each temporal edge, how many other edges have the same destination as its. In the User Graph there are no requirements for what concerns the label on the temporal edges, whereas for what concerns the Post Graph we take into consideration this factor.

Algorithm 11

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

1: **procedure** POST_INSTARCOUNTER(G)
2: **for each** $e \in E$ **do**
3: **if** e is a post **then**
4: $comList \leftarrow \{c \mid$ c is a comment $\wedge$ c targets e $\wedge$ $c.timestamp - e.timestamp \leq \Delta_T\}$
5: Remove from $comList$ all the comments with duplicate authors
6: $starLen \leftarrow length(comList)$
7: UPDATEMOTIFCOUNTER$(inStar, 3, starLen, postGraph)$
8: **else if** e is a comment **then**
9: $repList \leftarrow \{r \mid$ r is a reply $\wedge$ r targets e $\wedge$ $r.timestamp - e.timestamp \leq \Delta_T\}$
10: Remove from $repList$ all the replies with duplicate authors
11: $starLen \leftarrow length(repList)$
12: UPDATEMOTIFCOUNTER$(inStar, 3, starLen, postGraph)$

Algorithm 12

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

1: **procedure** USR_INSTARCOUNTER(G)
2: **for each** $e \in E$ **do**
3: $star \leftarrow \{e_1 \mid e_1$ targets e $\wedge$ $c.timestamp - e.timestamp \leq \Delta_T\}$
4: Remove from $star$ all the events with duplicate authors
5: $starLen \leftarrow len(star)$
6: UPDATEMOTIFCOUNTER$(inStar, 3, starLen, usrGraph)$

The following algorithms, **Algorithm 13** and **Algorithm 14**, are the algorithms related to in-stars counting in the two types of graphs, with the snapshots approach.

Algorithm 13

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

1: **procedure** SNAP_POST_INSTARCOUNTER(G)
2: $startSnap \leftarrow E.get(0).timestamp, endSnap \leftarrow startSnap + \Delta_T$
3: **for each** $e \in E$ **do**
4: **if** $e.timestamp \geq endSnap$ **then**
5: $endSnap \leftarrow e.timestamp + \Delta_T$
6: **else**
7: **if** e is a post **then**
8: $comList \leftarrow \{c \mid$ c is a comment $\wedge$ c targets e $\wedge$ $c.timestamp \leq endSnap\}$
9: Remove from $comList$ all the comments with duplicate authors
10: $starLen \leftarrow length(comList)$
11: UPDATEMOTIFCOUNTER$(inStar, 3, starLen, postGraph)$
12: **else if** e is a comment **then**
13: $repList \leftarrow \{r \mid$ r is a reply $\wedge$ r targets e $\wedge$ $r.timestamp \leq endSnap\}$
14: Remove from $repList$ all the replies with duplicate authors
15: $starLen \leftarrow length(repList)$
16: UPDATEMOTIFCOUNTER$(inStar, 3, starLen, postGraph)$

Algorithm 14

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

1: **procedure** SNAP_USR_INSTARCOUNTER(G)
2: $startSnap \leftarrow E.get(0).timestamp, endSnap \leftarrow startSnap + \Delta_T$
3: **for each** $e \in E$ **do**
4: **if** $e.timestamp \geq endSnap$ **then**
5: $endSnap \leftarrow e.timestamp + \Delta_T$
6: **else**
7: $star \leftarrow \{e_1 \mid e_1$ targets e $\wedge c.timestamp \leq endSnap\}$
8: Remove from $star$ all the events with duplicate authors
9: $starLen \leftarrow len(star)$
10: UPDATEMOTIFCOUNTER$(inStar, 3, starLen, usrGraph)$

3.3.4 Chain Algorithms

The algorithms we used to count chains in the Post Graph and in the User Graph, by using the time window sliding approach, are respectively presented in **Algorithm 15** and **Algorithm 16**. Starting from each temporal edge, in the Post Graph, we search for a chain of interactions that respect the Post Graph hierarchy within the time window. In the User Graph, instead, we count interactions of all kinds, as long as the source of an interaction is the same user as the destination of another one.

Algorithm 15

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

1: **procedure** POST_CHAINCOUNTER(G)
2: **for each** $e \in E$ **do**
3: **if** e is a post **then**
4: $comments = \{c \mid$ c is a comment $\wedge$ c targets e $\wedge$ $c.timestamp - e.timestamp \leq \Delta_T\}$
5: **for each** $c \in comments$ **do**
6: $replies = \{r \mid$ r is a reply $\wedge$ r targets c $\wedge$ $r.timestamp - e.timestamp \leq \Delta_T\}$
7: Remove from *replies* all the replies with duplicate authors
8: $chainLen \leftarrow len(replies) + 2$
▷ We add 2 because the chain includes the post and the comment
9: UPDATEMOTIFCOUNTER($chain, 3, chainLen, postGraph$)
10: **else if** e is a comment **then**
11: $replies = \{r \mid$ r is a reply $\wedge$ r targets e $\wedge$ $r.timestamp - e.timestamp \leq \Delta_T\}$
12: Remove from *replies* all the replies with duplicate authors
13: $chainLen \leftarrow len(replies) + 1$
▷ We add 1 because the chain includes the comment
14: UPDATEMOTIFCOUNTER($chain, 3, chainLen, postGraph$)
15: **else if** e is a reply **then**
16: $replies = \{r \mid$ r is a reply $\wedge$ $r.target == e.target$ $\wedge$ $r.timestamp - e.timestamp \leq \Delta_T\}$
17: Remove from *replies* all the replies with duplicate authors
18: $chainLen \leftarrow len(replies)$
19: UPDATEMOTIFCOUNTER($chain, 3, chainLen, postGraph$)

Algorithm 16

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
1:  procedure UsrChainCounter(G)
2:      for each e ∈ E do
3:          currEl ← e
4:          chainLen ← 1
5:          while true do
6:              searchedUsr ← currEl.dest
7:              currEl ← currEl.next
8:              if currEl.timestamp > e.timestamp then
9:                  break
10:             if currEl.src == searchedUsr then
11:                 chainLen ← chainLen + 1
12:         UpdateMotifCounter(chain, 3, chainLen, usrGraph)
```

Algorithm 17 and **Algorithm 18** are the algorithms used to count chains in the Post Graph and in the User Graph using the snapshots approach.

Algorithm 17

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
1:  procedure SnapPostChainCounter(G)
2:      startSnap ← E.get(0).timestamp, endSnap ← startSnap + Δ_T
3:      for each e ∈ E do
4:          if e.timestamp ≥ endSnap then
5:              endSnap ← e.timestamp + Δ_T
6:          else
7:              if e is a post then
8:                  comments = {c | c is a comment ∧ c targets e ∧
                    c.timestamp ≤ endSnap}
9:                  for each c ∈ comments do
10:                     replies = {r | r is a reply ∧ r targets c ∧
                        r.timestamp ≤ endSnap}
11:                     Remove from replies all the replies with duplicate authors
12:                     chainLen ← len(replies) + 2
13:                     UpdateMotifCounter(chain, 3, chainLen, postGraph)
```

Algorithm 17 (cont'd)

```
14:         else if e is a comment then
15:             replies = {r | r is a reply ∧ r targets e ∧
            r.timestamp ≤ endSnap}
16:             Remove from replies all the replies with duplicate authors
17:             chainLen ← len(replies) + 1
                                ▷ We add 1 because the chain includes the comment
18:             UpdateMotifCounter(chain, 3, chainLen, postGraph)
19:         else if e is a reply then
20:             replies = {r | r is a reply ∧ r.target == e.target ∧
            r.timestamp ≤ endSnap}
21:             Remove from replies all the replies with duplicate authors
22:             chainLen ← len(replies)
23:             UpdateMotifCounter(chain, 3, chainLen, postGraph)
```

Algorithm 18

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
 1: procedure Snap_Usr_ChainCounter(G)
 2:     startSnap ← E.get(0).timestamp, endSnap ← startSnap + Δ_T
 3:     for each e ∈ E do
 4:         if e.timestamp ≥ endSnap then
 5:             endSnap ← e.timestamp + Δ_T
 6:         else
 7:             currEl ← e
 8:             chainLen ← 1
 9:             while true do
10:                 searchedUsr ← currEl.dest
11:                 currEl ← currEl.next
12:                 if currEl.timestamp > endSnap then
13:                     break
14:                 if currEl.src == searchedUsr then
15:                     chainLen ← chainLen + 1
16:             UpdateMotifCounter(chain, 3, chainLen, usrGraph)
```

3.3.5 One-Way Couple Algorithms

In **Algorithm 19** and **Algorithm 20** we illustrate the procedures we followed to count this motif respectively in the Post Graph and in the User Graph, with the time window sliding approach. These simple algorithms count how many temporal edges share the same source and destination in the scope of the time window. In the Post Graph, as with every other temporal motif, the temporal edges all need to be of the same type label.

Algorithm 19

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
1:  procedure POST_ONEWAYCOUPLECOUNTER(G)
2:      for each e ∈ E do
3:          if e is a post then
4:              comments = {c | c is a comment ∧ c targets e ∧
                c.timestamp − e.timestamp ≤ Δ_T}
5:              for each c ∈ comments do
6:                  oneWayInteractions ← {c_1 | c_1 ∈ comments ∧ c_1 ≠ c ∧
                c_1 targets e ∧ c_1.author == c.author}
7:                  oWCLen ← length(oneWayInteractions)
8:                  UPDATEMOTIFCOUNTER(oneWayCouple, 3, oWCLen, postGraph)
9:          else if e is a comment then
10:             replies = {r | r is a reply ∧ r targets e ∧
                r.timestamp − e.timestamp ≤ Δ_T}
11:             for each r ∈ replies do
12:                 oneWayInteractions ← {r_1 | r_1 ∈ replies ∧ r_1 ≠ r ∧
                r_1 targets e ∧ r_1.author == r.author}
13:                 oWCLen ← length(oneWayInteractions)
14:                 UPDATEMOTIFCOUNTER(oneWayCouple, 3, oWCLen, postGraph)
```

Algorithm 20

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

1: **procedure** USR_ONEWAYCOUPLECOUNTER(G)
2: **for each** $e \in E$ **do**
3: $currSrc, currDest \leftarrow e.src, e.dest$
4: $oneWayInteractions \leftarrow \{e_1 \mid e_1 \in E \wedge e_1 \neq e \wedge$
$e_1.src == e.src \wedge e_1.dest == e.dest \wedge$
$e_1.timestamp - e.timestamp \leq \Delta_T$
5: $oWCLen \leftarrow length(oneWayInteractions)$
6: UPDATEMOTIFCOUNTER($oneWayCouple, 3, oWCLen, usrGraph$)

In the following algorithms, **Algorithm 21** and **Algorithm 22**, we show the algorithms related to one-way couples counting in the two types of graphs, with the snapshots approach.

Algorithm 21

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

1: **procedure** SNAP_POST_ONEWAYCOUPLECOUNTER(G)
2: $startSnap \leftarrow E.get(0).timestamp$
3: $endSnap \leftarrow startSnap + \Delta_T$
4: **for each** $e \in E$ **do**
5: **if** $e.timestamp \geq endSnap$ **then**
6: $endSnap \leftarrow e.timestamp + \Delta_T$
7: **else**
8: **if** e is a post **then**
9: $comments = \{c \mid$ c is a comment $\wedge$ c targets e $\wedge$
$c.timestamp \leq endSnap\}$
10: **for each** $c \in comments$ **do**
11: $oneWayInteractions \leftarrow \{c_1 \mid c_1 \in comments \wedge c_1 \neq c \wedge$
c_1 targets e $\wedge$ $c_1.author == c.author\}$
12: $oWCLen \leftarrow length(oneWayInteractions)$
13: UPDATEMOTIFCOUNTER($oneWayCouple, 3, oWCLen, postGraph$)

Algorithm 21 (cont'd)

```
14:         else if e is a comment then
15:             replies = {r | r is a reply ∧ r targets e ∧
            r.timestamp ≤ endSnap}
16:             for each r ∈ replies do
17:                 oneWayInteractions ← {r1 | r1 ∈ replies ∧ r1 ≠ r ∧
            r1 targets e ∧ r1.author == r.author}
18:                 oWCLen ← length(oneWayInteractions)
19:                 UpdateMotifCounter(oneWayCouple, 3, oWCLen, postGraph)
```

Algorithm 22

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
 1: procedure Snap_Usr_OneWayCoupleCounter(G)
 2:     startSnap ← E.get(0).timestamp
 3:     endSnap ← startSnap + ΔT
 4:     for each e ∈ E do
 5:         if e.timestamp ≥ endSnap then
 6:             endSnap ← e.timestamp + ΔT
 7:         else
 8:             currSrc, currDest ← e.src, e.dest
 9:             oneWayInteractions ← {e1 | e1 ∈ E ∧ e1 ≠ e ∧
                e1.src == e.src ∧ e1.dest == e.dest∧
                e1.timestamp ≤ endSnap
10:             oWCLen ← length(oneWayInteractions)
11:             UpdateMotifCounter(oneWayCouple, 3, oWCLen, usrGraph)
```

3.3.6 P-Triangle Algorithms

Since the p-triangle is based on a ping pong, the algorithm for its counting in the graphs can be derived from the ping pong ones. In fact, in order to find a p-triangle, it is sufficient that a ping pong is generated between two users that have commented or replied to a post/comment of a trigger event.

Algorithm 23 and **Algorithm 24** contain the algorithms used to count p-triangles using respectively the time window sliding approach and the snapshots approach. As we previously said, they are practically the same as the ping pong algorithms, apart from the initial research of the two triggering interactions.

Algorithm 23

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
1: procedure PTriangleCounter(G)
2:     for each e ∈ E do
3:         if e is a post then
4:             comments = {c | c is a comment ∧ c targets e ∧
               c.timestamp − e.timestamp ≤ Δ_T}
5:             for each pair of comments c_1, c_2 do
6:                 interactNum ← the length of a ping pong between c_1 and c_2 ù
7:                 if interactNum mod 2 ≠ 0 then
8:                     interactNum ← interactNum − 1
9:                 UpdateMotifCounter(pTriangle, 2, interactNum, postGraph)
10:        else if e is a comment then
11:            replies = {r | r is a reply ∧ r targets e ∧
               r.timestamp − e.timestamp ≤ Δ_T}
12:            for each pair of replies r_1, r_2 do
13:                interactNum ← the length of a ping pong between r_1 and r_2
14:                if interactNum mod 2 ≠ 0 then
15:                    interactNum ← interactNum − 1
16:                UpdateMotifCounter(pTriangle, 2, interactNum, postGraph)
```

Algorithm 24

Input: $G = (V, E, \Delta_T)$ is a temporal post graph

```
1: procedure SNAP_PTRIANGLECOUNTER(G)
2:     startSnap ← E.get(0).timestamp
3:     endSnap ← startSnap + Δ_T
4:     for each e ∈ E do
5:         if e.timestamp ≥ endSnap then
6:             endSnap ← e.timestamp + Δ_T
7:         else
8:             if e is a post then
9:                 comments = {c | c is a comment ∧ c targets e ∧
               c.timestamp ≤ endSnap}
10:                for each pair of comments c_1, c_2 do
11:                    interactNum ← the length of a ping pong between c_1 and c_2
12:                    if interactNum mod 2 ≠ 0 then
13:                        interactNum ← interactNum − 1
14:                    UPDATEMOTIFCOUNTER(pTriangle, 2, interactNum, postGraph)
15:            else if e is a comment then
16:                replies = {r | r is a reply ∧ r targets e ∧
               r.timestamp ≤ endSnap}
17:                for each pair of replies r_1, r_2 do
18:                    interactNum ← the length of a ping pong between r_1 and r_2
19:                    if interactNum mod 2 ≠ 0 then
20:                        interactNum ← interactNum − 1
21:                    UPDATEMOTIFCOUNTER(pTriangle, 2, interactNum, postGraph)
```

Chapter 4

Data and Methods

Abstract

In this chapter we focus on a real Facebook dataset that constituted the ground on which we applied our algorithms to find the temporal motifs discussed before. Moreover, we write about the programs we used to run our code, and describe the procedures through which we elaborated the most appropriate values for the time windows we included in our Temporal Graphs.

4.1 Description of the dataset

In this work we analyse communication patterns in 18 Facebook groups. Each one of them belongs to one of the categories listed in **Table 4.1**. The table also displays the number of groups for each category.
We have three files with tabular format available for each group. The first one contains identifiers for all the members of the group. The second one and the third one respectively contain information about the posts and the comments and replies of the group. For each post we have an identifier and details about the author, the timestamp and the number of shares. Other than the fields just listed, the comments and replies file also reports information about the type of interaction (it takes the value C for comments, R for replies), as well as the identifier of the event's target. On the other hand, it doesn't contain any information about the shares, since Facebook users aren't allowed to share comments or replies. The events timestamp field has the Unix Epoch format. **Figure 4.1** contains a representation of the Entity-Relationship Diagram of our dataset.

Group Categories	
Category	No. of groups
Education	3
Entertainment	4
News	3
Sport	4
Work	4

Table 4.1: The categories to which the Facebook groups in our datasets belong

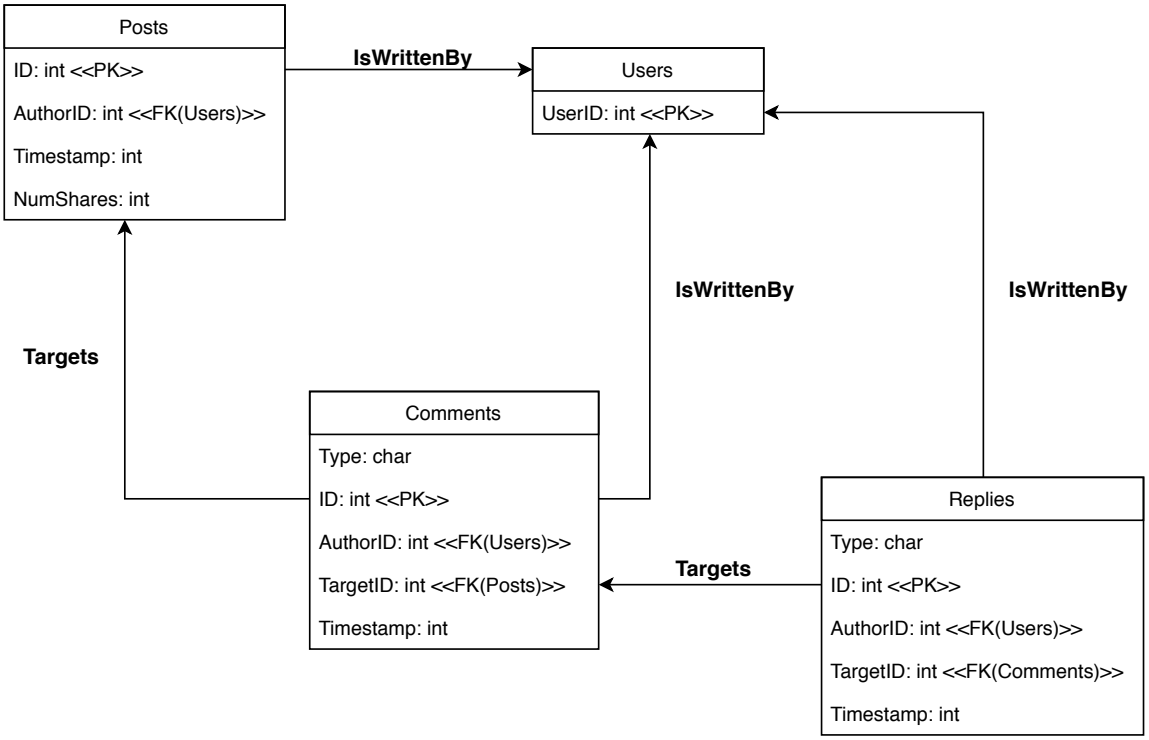


Figure 4.1: Entity-Relationships Diagram of our dataset

We gave each group a shorter name, in order for us to refer to them in an easier way. **Table 4.2** reports these associations.

Table 4.3 shows some data about the groups. We report attributes that are immediately extractable from the files, which are the number of members, the number of posts and comments, and the first/last date when we started/ended monitoring their

activity. Lastly, the table also includes, for each group, the average daily number of posts and comments and the average time (in seconds) that elapses between an event and the next one. As we can see from the table, in the face of a similar number of posts and comments (with the same order of magnitude for each group, with the exception of a couple ones), the number of members is very different, ranging from a few hundreds to tens of thousands. The number of monitored days is quite heterogeneous as well. Some groups have been observed for less than half year, others for more than a year.

Original Name	**Shorter Name**
Programmatori Need for Nerd - Code Community	Edu1
Studenti in Crisi	Edu2
AIRI	Edu3
Thriller storici e dintorni	Ent1
Horror House	Ent2
Scimmie Cinefile Official	Ent3
Chitarra Acustica	Ent4
Gossip, Anticipazioni, News: il meglio della TV	News1
Discutiamo di politica e attualitá	News2
Quick Books Tips and Tricks	News3
La vetrina del tennis	Sport1
Pub del nuoto	Sport2
Palestra Gruppo Ufficilale	Sport3
Psiche e Benessere	Sport4
Italian Startup Scene	Work1
Malta Temp Work	Work2
Discussion Cafe @ Jobs	Work3
Sistemisti ed Amministratori di Reti-Italiani	Work4

Table 4.2: The full names we gave to the Facebook groups in our dataset and the shorter names we gave them in this work for better understanding

Group	**#M**[1]	**#P**[2]	**#C**[3]	**Avg. dP**[4]	**Avg. dC**[5]	**Min date**	**Max date**	**Avg.** δ_t[6]
Edu1	8383	3555	63350	9	162	01/01/2017	24/01/2018	528
Edu2	27079	5271	77933	17	244	06/04/2017	18/02/2018	337
Edu3	820	5060	41480	13	105	25/01/2017	22/02/2018	818
Ent1	6941	5009	65205	38	493	30/09/2017	08/02/2018	173
Ent2	3458	3777	32235	30	257	22/10/2017	23/02/2018	331
Ent3	2324	4904	72631	40	595	02/01/2018	03/05/2018	143
Ent4	9392	3543	33098	20	184	09/09/2017	06/03/2018	465
News1	49762	156	9777	1	85	07/10/2017	28/01/2018	998
News2	11663	3466	282358	38	3069	08/11/2017	07/02/2018	28
News3	3992	1133	5476	3	13	02/01/2017	12/02/2018	6407
Sport1	376	5383	3823	21	15	21/05/2017	27/01/2018	5674
Sport2	117	708	3421	2	9	04/02/2017	09/02/2018	9340
Sport3	4321	6353	79998	211	2666	13/02/2018	14/03/2018	31
Sport4	25849	5588	162283	22	649	27/09/2017	03/05/2018	132
Work1	26496	1444	19007	4	46	02/01/2017	12/02/2018	1843
Work2	4592	945	16891	2	40	04/01/2017	27/02/2018	2142
Work3	4056	4809	3296	15	10	13/06/2017	27/04/2018	8307
Work4	11842	2651	2382	5	5	03/01/2017	04/05/2018	17550

Table 4.3: Some quantitative data about our groups

4.2 Methods

All our experiments were conducted using the programming language Python 3.6.5, in the Integrated Development Environment PyCharm 2018.1.3 (Community Version). All the Python scripts have been executed on a machine with Microsoft Windows 10 as operating system. Python was considered being the best option because

[1]No. of members
[2]No. of posts
[3]No. of comments
[4]Avg. posts per day
[5]Avg. comments per day
[6]Avg. time between interactions (in seconds)

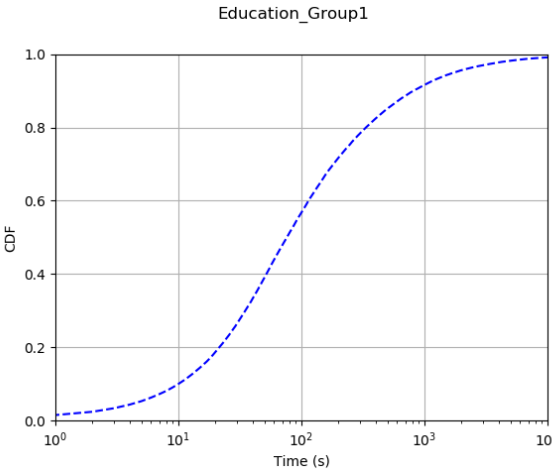
of its versatility and the vast amount of libraries available for data analysis, plotting and handling tabular files. Among the libraries used for our experiments, the most useful ones were for sure *csv*, *numpy* and the *pyplot* framework from *matplotlib*. Python complete documentation is available online at the following address: `https://docs.python.org/3.6/`. To elaborate the Correlation Matrixes we used KNIME Analytics Platform 3.7.0, an open source software useful to derive statistics on data.

As we previously said, our aim was to find how many motifs of different kinds develop in group-dependent time frames. We first describe the way we found the most appropriate time windows for each one of our groups. The choices we made were based on the fact that the different groups tend to have differing speeds of information spreading. For this reason, the times between events could be predominantly higher for one group and lower for others. Adopting the same time parameter for each group without taking into consideration this issue would have been a mistake. Therefore, the decision we took was to find, for every group, the different quartiles for the interaction time intervals. This way, we could capture within what time frames the 25%, 50% and 75% of the time lengths between events are included. In order to find the quartiles, we ordered the events belonging to each group by timestamp and produced a list of intervals. Each interval has been calculated as the time difference between an event and the previous one. We then sorted this list and extracted the three quartiles — each one being the amount of time by which the first, second and third quarter of interaction times stand. **Table 4.4** shows the quartiles for each group. As we can see, for most of the groups the first quartile ranges around a few tens of seconds, apart from two groups in the Sport category and two in the Work one, that are in the hundreds. The second quartile is still quite low, and for half of the groups still stays under a hundred seconds. In the other half, for three groups the second quartile rises above two thousand seconds. Regarding the third quartile, it is in the range of thousand seconds for six out of eighteen groups: this means that 75% of the interactions, in those groups, take place by a time that goes from around fifteen minutes to (in one case) more than three hours from the previous event. On the other hand, it is impressive that for three out of eighteen groups the third quartile has a value that is below one hundred seconds. In those groups the activity is thus very fast-paced, and the interactions take place in intervals of around one minute from each other.

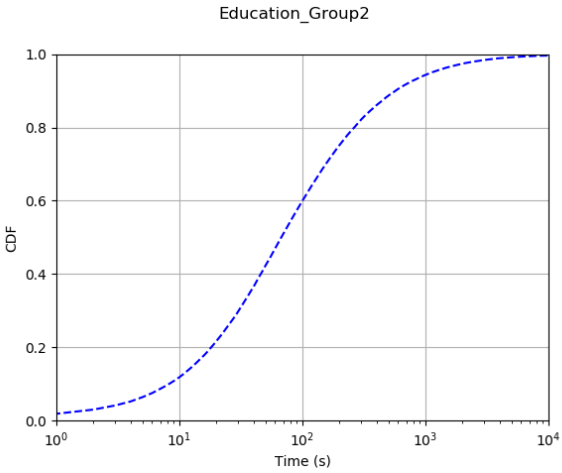
Group	**1st Quartile**	**2nd Quartile**	**3rd Quartile**
Edu1	28	77	243
Edu2	24	68	198
Edu3	58	172	582
Ent1	26	63	152
Ent2	35	95	266
Ent3	12	30	71
Ent4	47	129	375
News1	18	94	438
News2	4	11	26
News3	89	338	3187
Sport1	551	2158	6211
Sport2	113	653	3617
Sport3	5	14	30
Sport4	17	45	109
Work1	75	265	1045
Work2	65	213	993
Work3	316	2438	8479
Work4	264	2251	13586

Table 4.4: The three quartiles that we used as time windows lengths for our groups

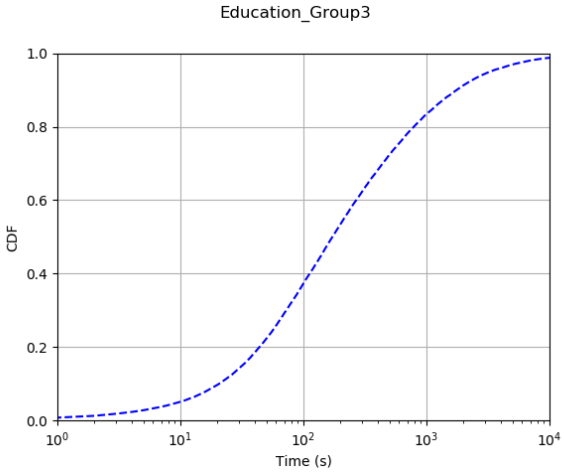
To obtain a graphical representation of the interactions percentages we just discussed, we also elaborated the CDF (cumulative distribution function) of the time between interactions. **Figure 4.2** shows, for each group, the plots of the function, that are clearly in line with the values we found for the quartiles.



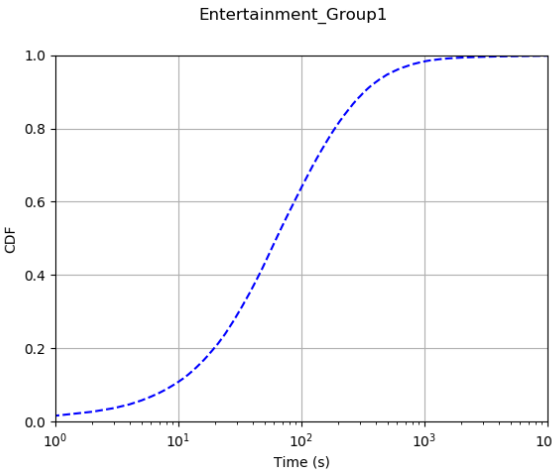
(a) Edu1



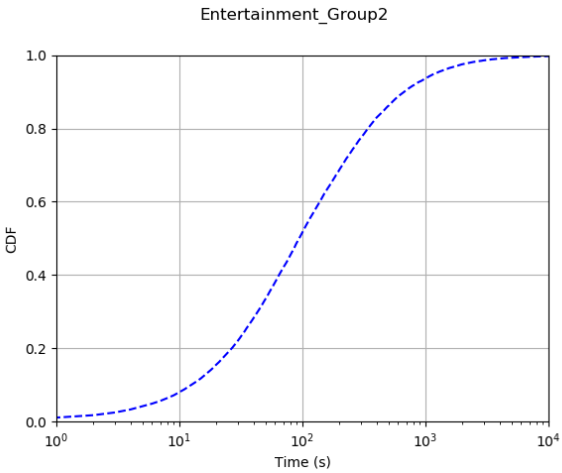
(b) Edu2



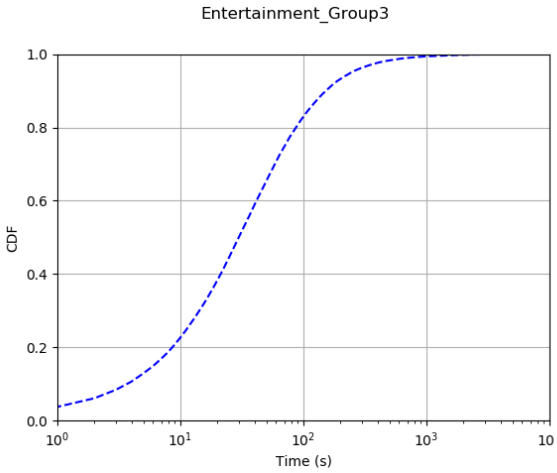
(c) Edu3



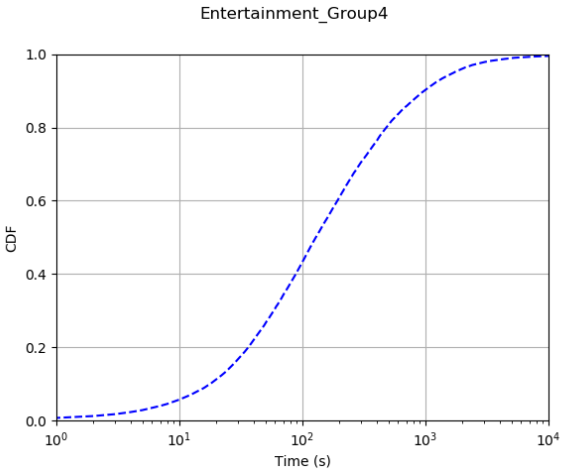
(d) Ent1



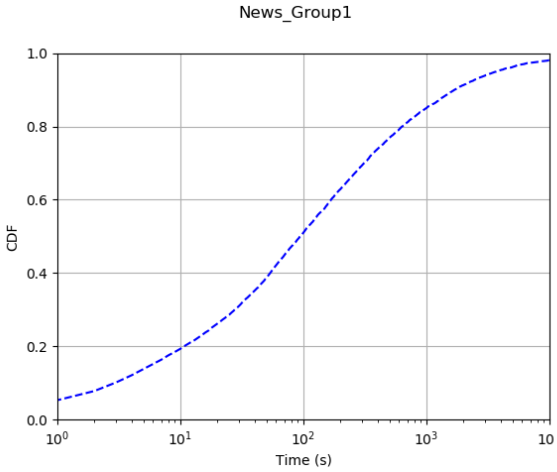
(e) Ent2



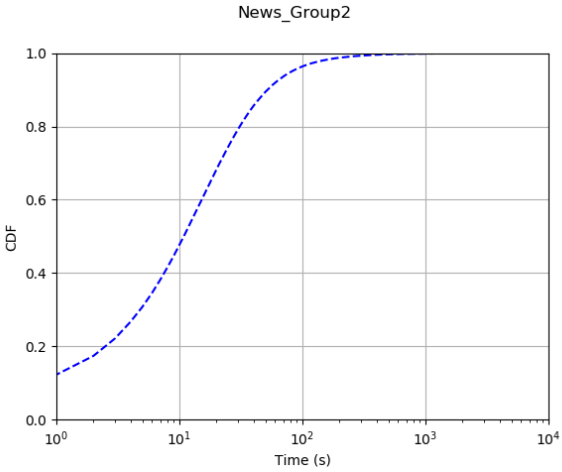
(f) Ent3



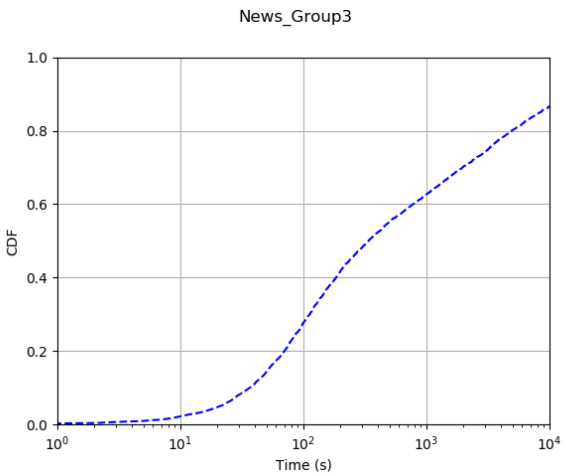
(g) Ent4



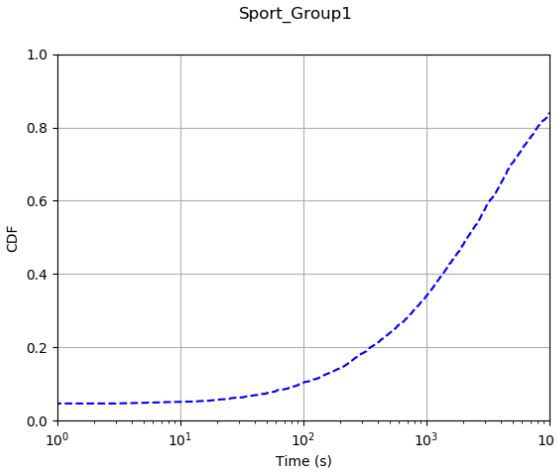
(h) News1



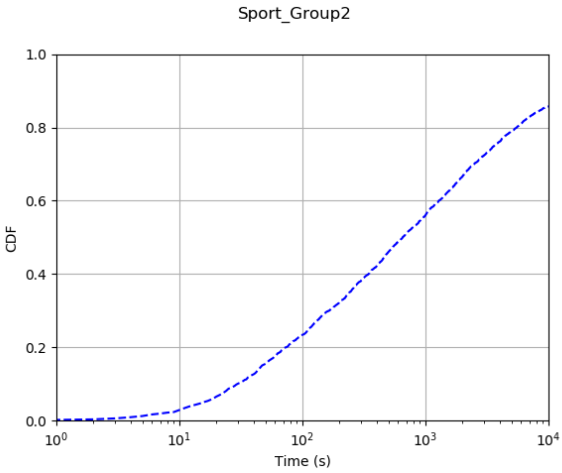
(i) News2



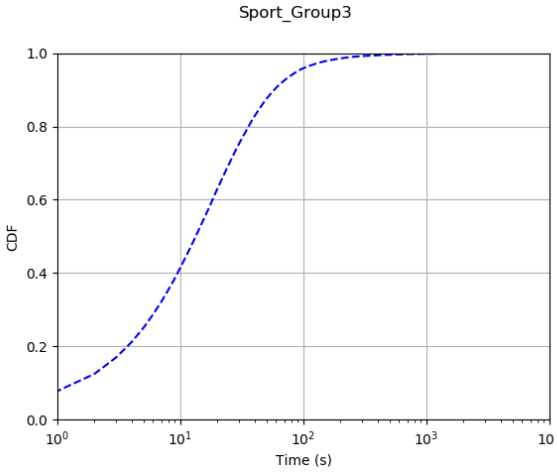
(j) News3



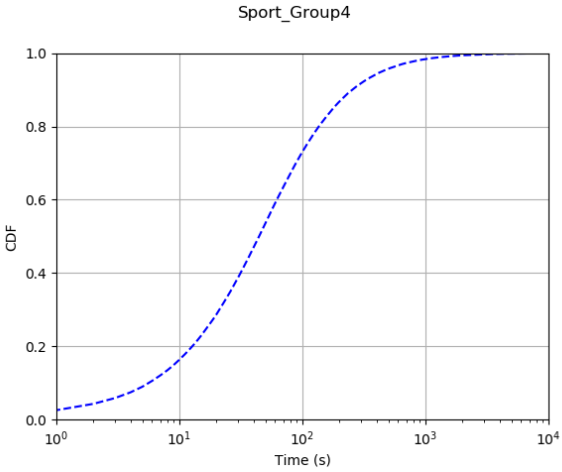
(k) Sport1



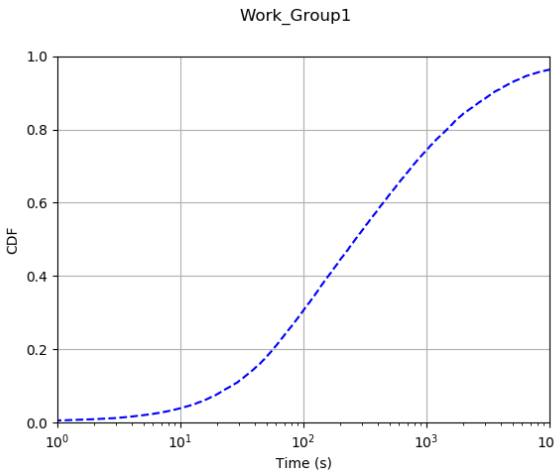
(l) Sport2



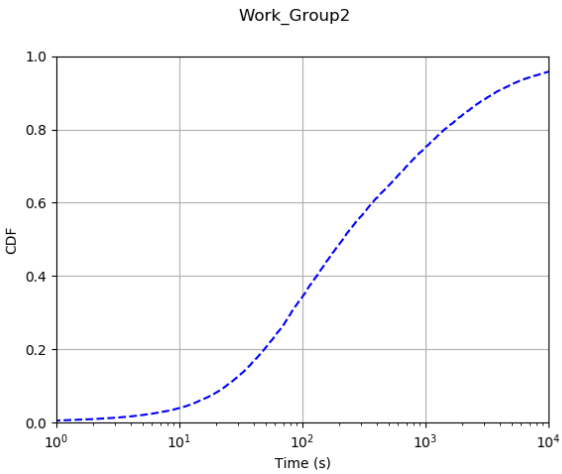
(m) Sport3



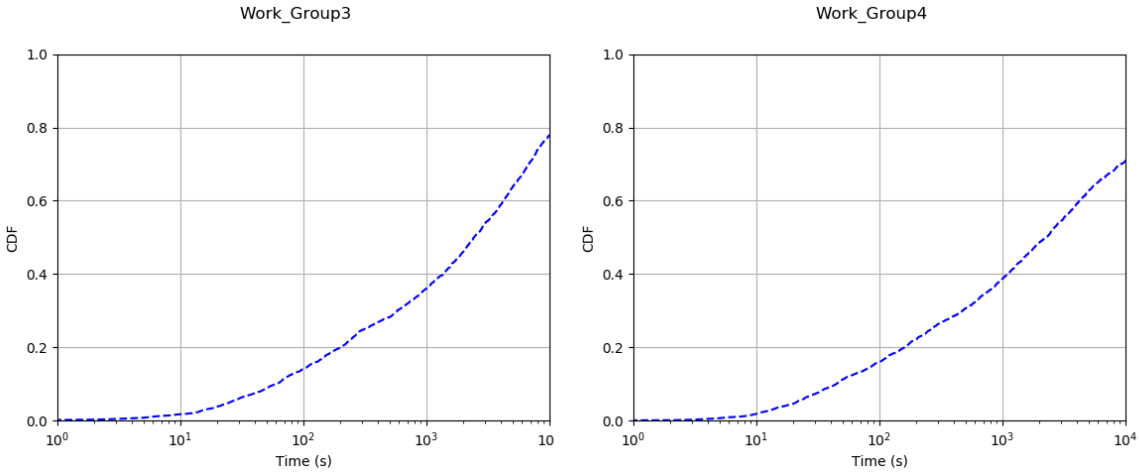
(n) Sport4



(o) Work1



(p) Work2



(q) Work3 (r) Work4

Figure 4.2: The plots of the CDF for our groups

For each temporal motif, we ran the corresponding algorithms we wrote about in **Chapter 3** on each group, passing the quartile values as parameters for the time window. Each time a motif of length k was found, the respective counter was updated, increasing the number of motifs of length from k down to its minimum size. As we previously said, this aspect is what made our approach different and innovative compared to others: since our research was parameterized by k, we have been able to find the most common lengths for the various motifs in the groups, having the chance to notice what were the maximum lengths in different time windows and groups. Moreover, we had the possibility to compare the results we obtained for groups belonging to the same category, hoping to find some similarities. We will further discuss the results in **Chapter 5**. For each group we kept two counters, one for the time window sliding approach and the other for the snapshots one. The counters collected results for both the post graph and the user graph. **Algorithm 1** shows how each counter is structured as a three-level hash map. The third level of the map contains the specific counter of a motif with a certain length, in the scope of one of the two graphs of a group. The results found by the execution of our algorithms have been stored in *.csv* files, where for each group, time window and k we reported the number of each motif with those specific parameters we obtained from the counter. In **Figure 4.3** we provide an example of the first rows of the file that contains the results for the Post Graph of each group.

	A	B	C	D	E	F	G	H	I	J
1	Group Type	Group Number	Time Window	k	Chains	Ping pongs	In-stars	Out-stars	One-Way-Couples	P-Triangles
2	Education	Group1	28	2	0	3	0	0	0	0
3	Education	Group1	28	3	292	0	5	465	0	0
4	Education	Group1	28	4	8	0	0	2	0	0
5	Education	Group1	77	2	0	140	0	0	0	37
6	Education	Group1	77	3	1619	0	72	1951	1	0
7	Education	Group1	77	4	185	2	7	98	0	0
8	Education	Group1	77	5	5	0	0	8	0	0
9	Education	Group1	243	2	0	1961	0	0	0	1264
10	Education	Group1	243	3	5140	0	517	4917	48	0
11	Education	Group1	243	4	1250	118	89	665	14	72
12	Education	Group1	243	5	121	0	25	138	5	0
13	Education	Group1	243	6	2	6	6	34	2	3
14	Education	Group1	243	7	0	0	1	12	0	0
15	Education	Group1	243	8	0	1	0	3	0	0
16	Education	Group1	243	9	0	0	0	1	0	0
17	Education	Group2	24	2	0	1	0	0	0	0
18	Education	Group2	24	3	26	0	1	780	0	0
19	Education	Group2	24	4	1	0	0	56	0	0
20	Education	Group2	24	5	0	0	0	7	0	0
21	Education	Group2	24	6	0	0	0	2	0	0
22	Education	Group2	68	2	0	68	0	0	0	5

Figure 4.3: A screenshot taken from one of the results files

The results files were the key input for our KNIME flow. In order to effectively find correlation between groups, we needed to provide bi-dimensional matrixes. We thus decided to restrict our analysis to one quartile at a time, taking into consideration significant motif lengths for each case. In particular, we chose the ones for which we counted the highest numbers of motifs in every group: for the first quartile it was 2 for the Ping-Pong and the P-Triangle, and 3 for all the other motifs; for the second and third quartile it was 4 for all motifs. Our KNIME flow was made up of three nodes. The first one was a CSV Reader. It takes as its input a .csv file, where columns are what we want to find correlation for, and rows are the attributes that are the actual base of the correlation. Since our results file had the groups as row variables, we first needed to transpose the results tables contained in them. Moreover, as we just said, we needed to extract a bi-dimensional table that just contained the number of motifs in each group, in a specific quartile and k context. For each analysis we thus removed the rows we didn't need. Our second node was the Low Variance Filter, that simply excluded columns with low variance that could interfere with the learning algorithms involved in the flow, as suggested by KNIME documentation. Finally, our last node, that is the one that generated our correlation tables, was the Linear Correlation, which computed the correlation values we were looking for with the Pearson Correlation Coefficient equation for each couple of columns (i.e. groups). Pearson Correlation Coefficient expresses the linear correlation between two variables X and Y. The correlation value goes from -1 to 1, that are respectively the

two extremes of negative and positive linear correlation. Positive correlation means that Y increases as X increases; negative correlation, on the contrary, means that Y decreases as X increases. A coefficient equivalent to 0 means that the two variables are not correlated.
The Linear Correlation node also visualized those values in a comprehensible way, that allowed us to evaluate them easily. This evaluation as well is presented in **Chapter 5**.

Chapter 5

Results

Abstract

*In **Chapter 4** we described the process that led us to analyze the dataset and elaborate our results. We now discuss them, compare them to our previous expectations and try to find some correlations between the motif-related behaviours in our Facebook groups.*

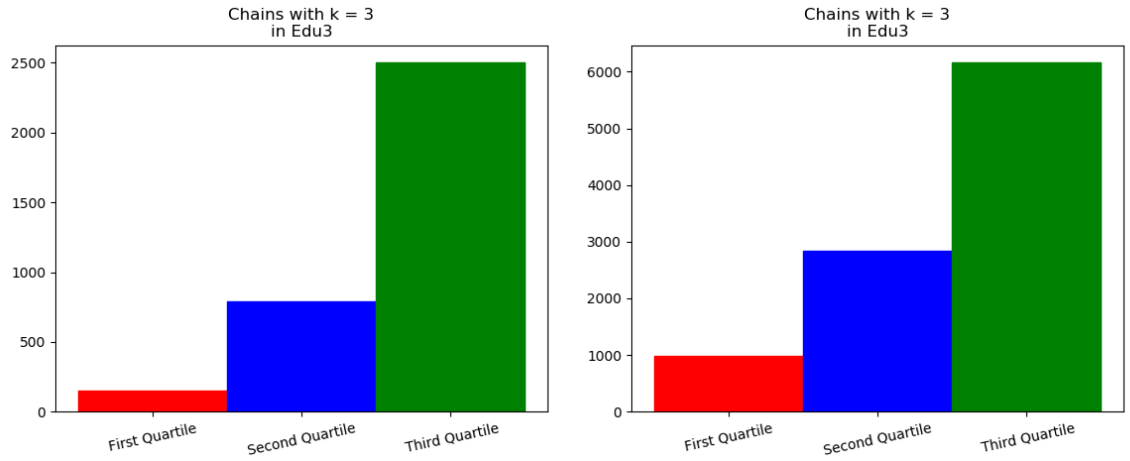
5.1 Counting of motifs

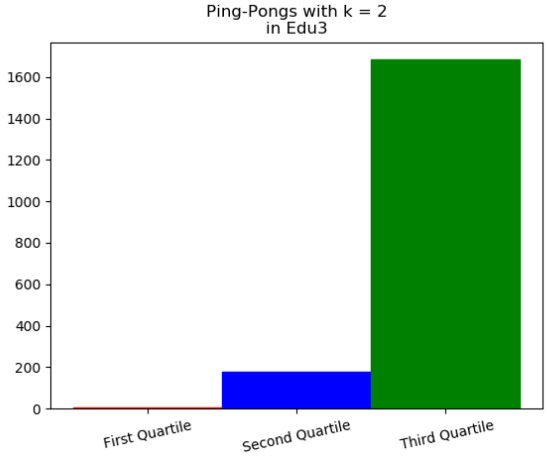
5.1.1 Group Diagrams

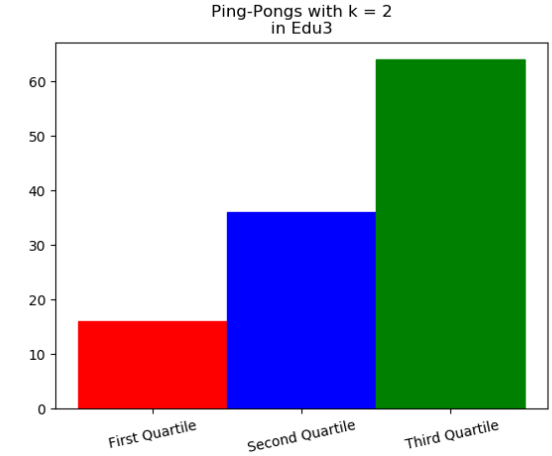
In order to obtain a better vision of the number of motifs we found, we decided to narrow down our results. In fact, we had quite a significant amount of parameters: the groups in our dataset, the chosen quartile, the motif we counted the occurrences of and the particular length reached by a motif. Our aim was to create diagrams as easy to comprehend at a glance as possible. It thus proved necessary to fix three parameters at a time, and create bi-dimensional diagrams with the remaining parameter as the x-axis variable and the number of motifs on the y-axis.
The first type of plot we decided to produce has a specific group and a k value as fixed parameters, and shows the number of motifs with that length and in that group in the three quartiles. Therefore, the x-axis is labeled by the three quartiles, while the number of motifs is on the y-axis.
The k values we chose for the motifs are representative of the shortest lengths of each one of them. In fact, we wanted to evaluate the growth of the number of motifs in each group, and thus we needed to consider lengths that were present in all groups, in order for them to share a common ground. In each bar chart we included the name

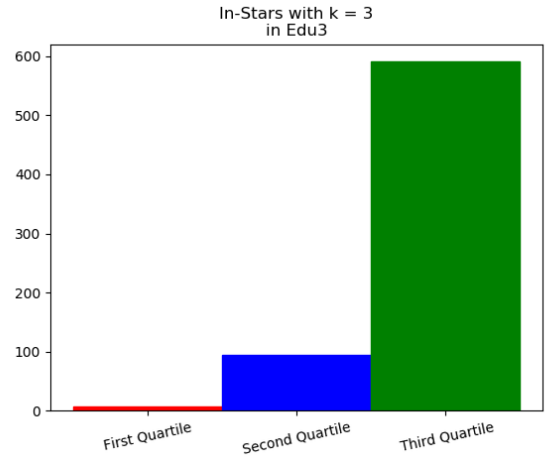
of the group and the length of the motif. The x-axis is labelled by the names of the three quartiles, while the y-axis varies based on the group and the maximum number of motifs found in the quartiles.
The charts have been created both for the Post Graph and the User Graph, splitting up the results found with the use of the time-window sliding and the snapshots approaches. Based on the plots, that are shown in **Appendix A**, we can state that the growth of the number of motifs is almost always linear or exponential with respect to the the growth of the time window. Looking at each group more in depth, there is a strong tendency of predominance by the number of Out-Stars over all the other motifs, meaning that users tend to actively interact in diverse contexts. In-Stars usually occur in a much smaller number compared to their just analyzed counterpart: single posts or comments generate less activity than individual users do all over the group (remember that the Out-Star is the only motif that fall outside the post scope). The motif that seems to be less present, instead, is the One-Way Couple; the stalker-like behaviour appears not to be so common.

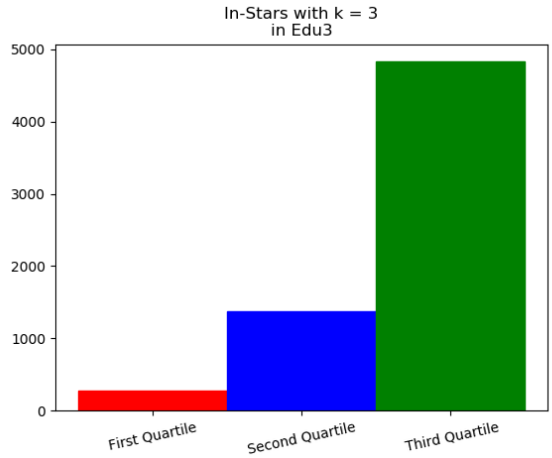
Figure 5.1 compares the Post Graph plots (in the left column) and User Graph ones (in the right column) obtained for a sample group, Edu3. We expected them to take higher values in the User Graph, with the exception of the Ping-Pong, that in the Post Graph contain replies addressed to other replies (for the reasons we explained in the previous chapters). Our expectations are fully satisfied by the results. As we can see in the example figure, motifs in the User Graph are much more numerous, reaching in the case of In-Stars almost ten times the numbers of the Post Graph. Ping-Pongs, instead, are way less in the User Graph, as anticipated.

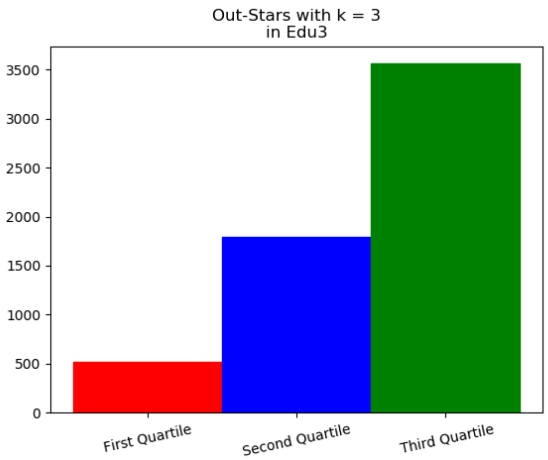


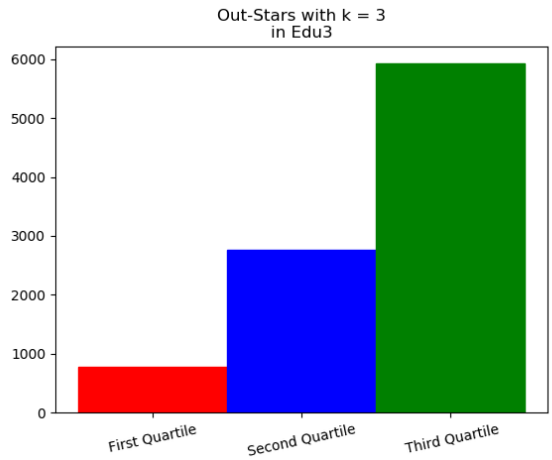

Ping-Pongs with k = 2
in Edu3
1600
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile


Ping-Pongs with k = 2
in Edu3
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile


In-Stars with k = 3
in Edu3
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile


In-Stars with k = 3
in Edu3
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile


Out-Stars with k = 3
in Edu3
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile


Out-Stars with k = 3
in Edu3
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

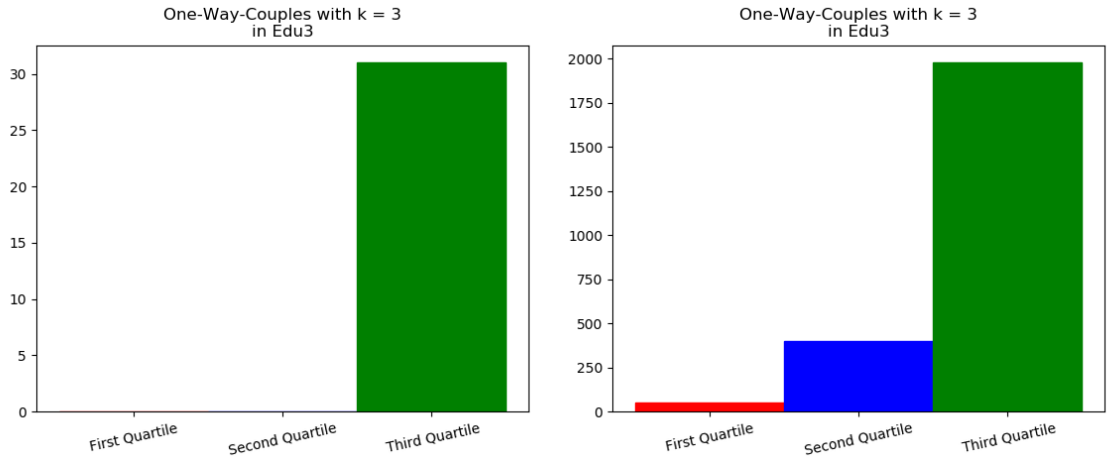
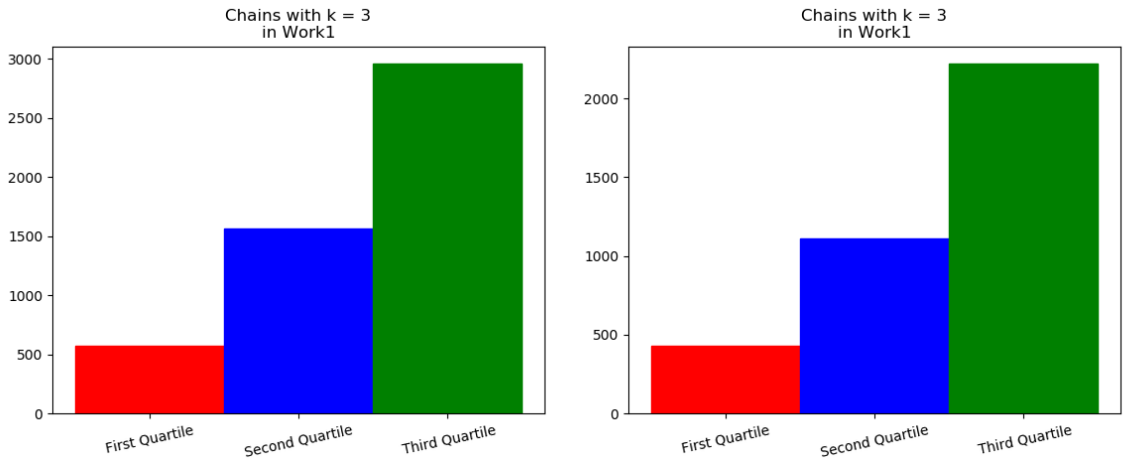
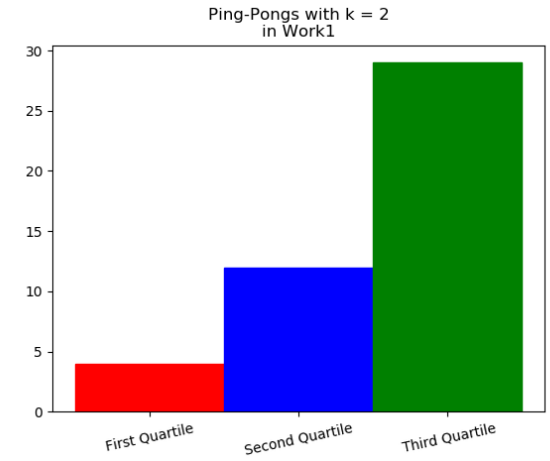
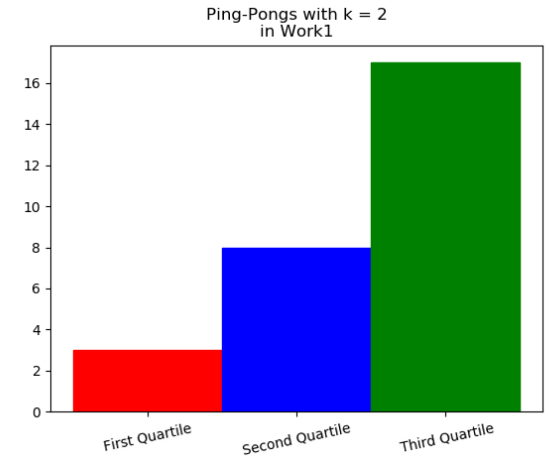


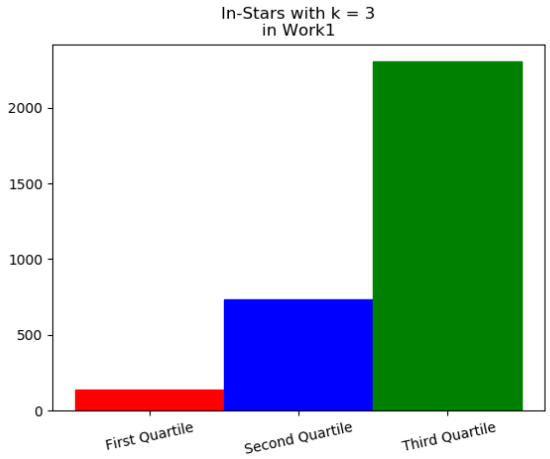
Figure 5.1: A comparison between some plots obtained from the results of the Post Graph (on the left) and the User Graph (on the right)

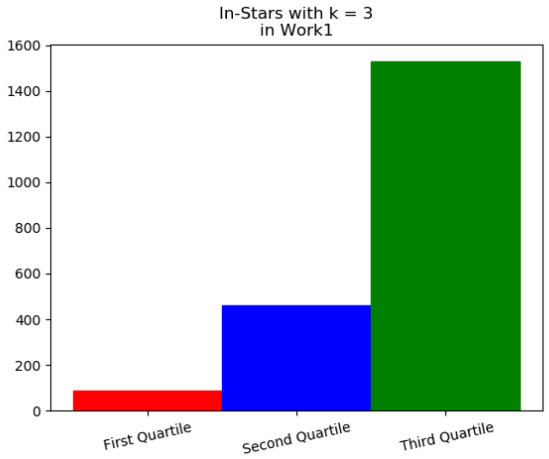
We now compare the results obtained with the two time window approaches. Since we already presented all the plots related to the Post Graph, we show the ones we got for the User Graph for this purpose. Previously in this work, we assumed that the snapshot approach would end up by cutting motifs' length. Therefore, motifs of any length should be less numerous if searched for with this kind of method as opposed to the time-window sliding one. **Figure 5.2** shows how our assumption was well-founded. The left column contains the diagrams of the motifs found with the time window sliding approach, that have values that vary from being dozens to multiple thousands more than the counterparts shown in the right column.

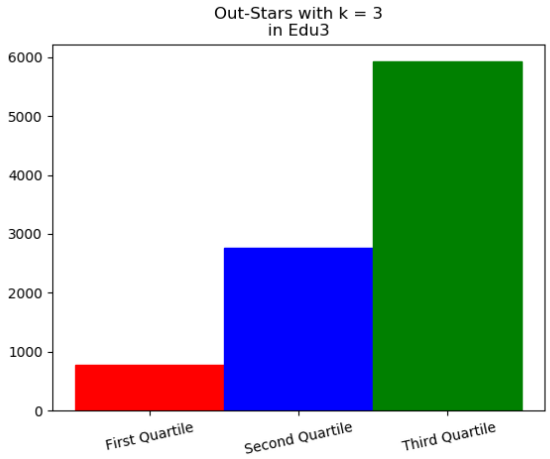


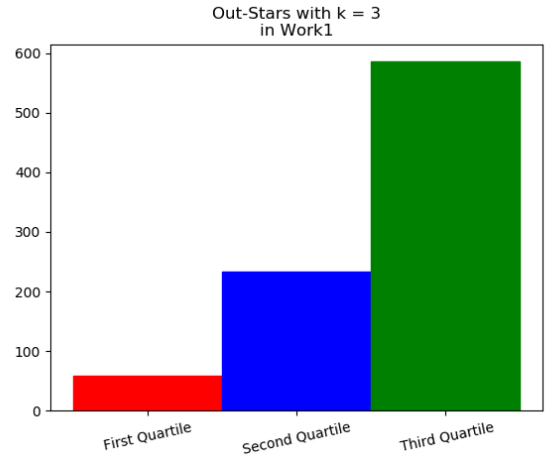

Ping-Pongs with k = 2
in Work1
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile


Ping-Pongs with k = 2
in Work1
16
14
12
10
8
6
4
2
0
First Quartile
Second Quartile
Third Quartile


In-Stars with k = 3
in Work1
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile


In-Stars with k = 3
in Work1
1600
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile


Out-Stars with k = 3
in Edu3
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile


Out-Stars with k = 3
in Work1
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

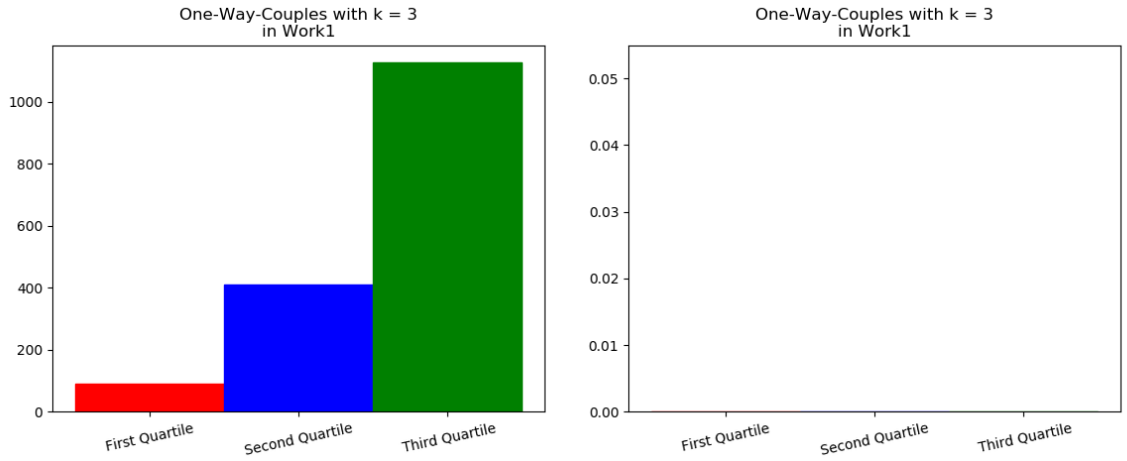
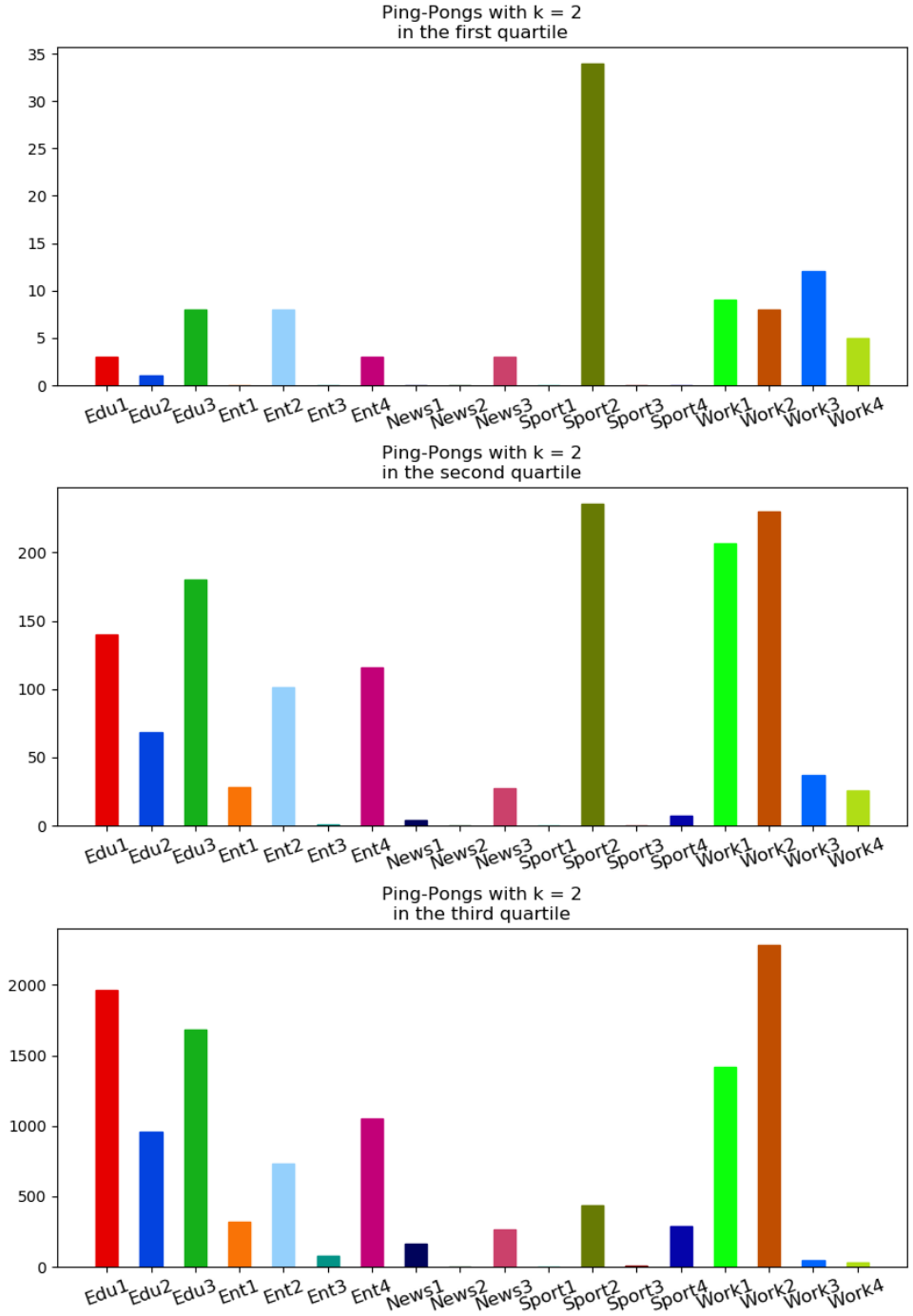


Figure 5.2: A comparison between some plots obtained from the results related to the time window sliding approach (on the left) and the snapshots approach (on the right)

5.1.2 Quartile Diagrams

Another way, maybe even more effective graphically than the one just shown, is to compare the number of motifs of each group in a diagram that represent a single quartile. This kind of plot gives a more global overview of what are the predominant groups in term of number of a particular motif. Each plot represents the counter of a particular motif in one of the three quartiles, and has all the groups on the x-axis and the number of motifs on the y-axis. It is important to keep in mind that, as the quartile number grows, so does the scale of the ordinates: the labels as well trivially change.

In **Figure 5.3** we decided to show the diagrams of P-Triangles and Ping-Pongs counted in the Post Graph, since the former are only found in it, and the latter, given their structure, are mainly present in it. The diagrams of the other motifs in **Figure 5.4, 5.5** and **5.6**, instead, are taken from the counters of the User Graph.


Ping-Pongs with k = 2
in the first quartile
0
5
10
15
20
25
30
35
Edu1
Edu2
Edu3
Ent1
Ent2
Ent3
Ent4
News1
News2
News3
Sport1
Sport2
Sport3
Sport4
Work1
Work2
Work3
Work4
Ping-Pongs with k = 2
in the second quartile
0
50
100
150
200
Edu1
Edu2
Edu3
Ent1
Ent2
Ent3
Ent4
News1
News2
News3
Sport1
Sport2
Sport3
Sport4
Work1
Work2
Work3
Work4
Ping-Pongs with k = 2
in the third quartile
0
500
1000
1500
2000
Edu1
Edu2
Edu3
Ent1
Ent2
Ent3
Ent4
News1
News2
News3
Sport1
Sport2
Sport3
Sport4
Work1
Work2
Work3
Work4

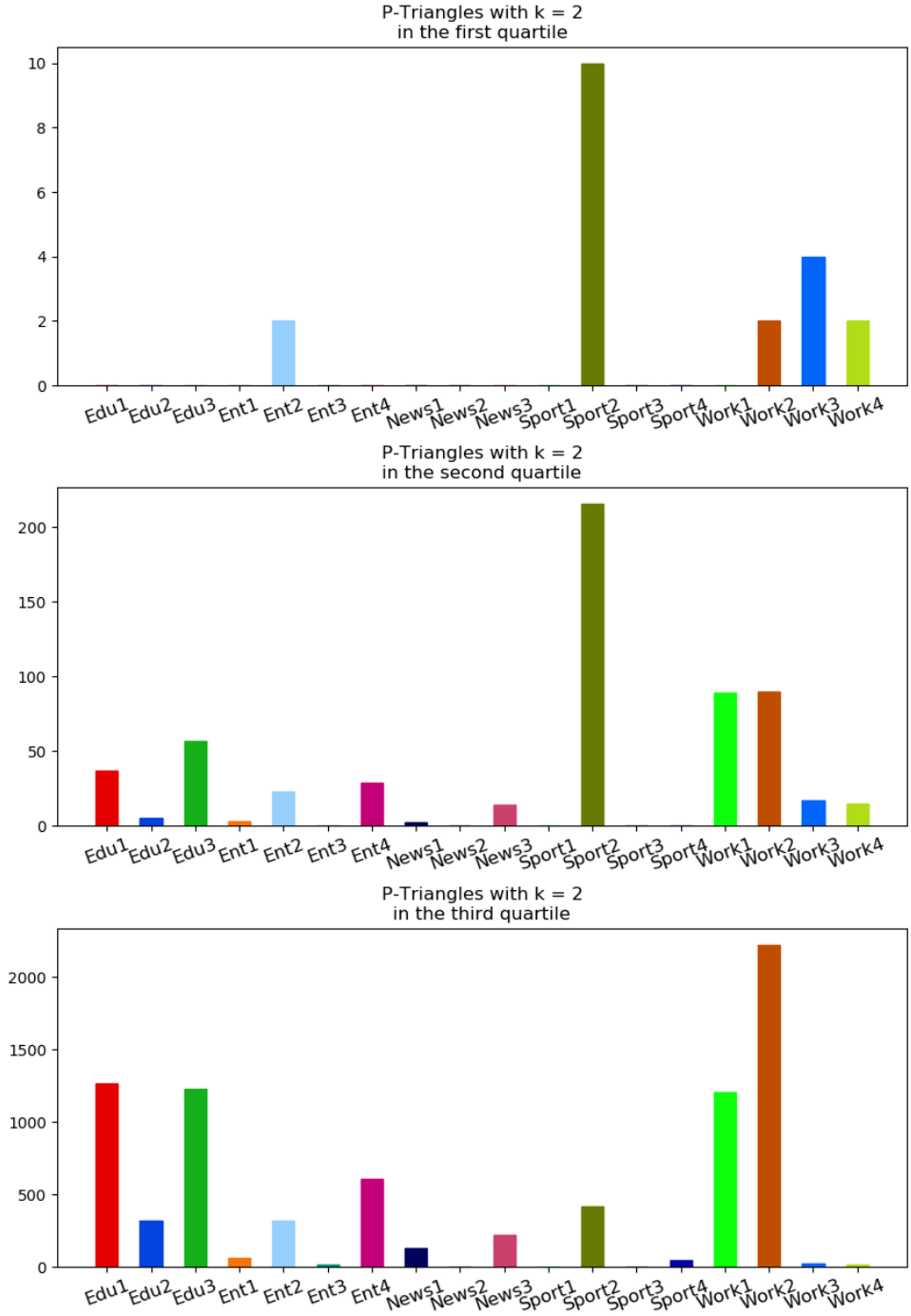


Figure 5.3: The three quartile-based plots related to the results obtained for the ping pongs and the p-triangles in the groups

As we can see in the figures above, both P-Triangles and Ping-Pongs reach particularly high numbers in groups that belong to the Education and Work categories. However, for both motifs it is immediate to notice in the first and second quartile the presence of a peak in Sport2. Since each P-Triangle is made up by a Ping-Pong, the

proportions between groups of the number of these two motifs are respected in each quartile.

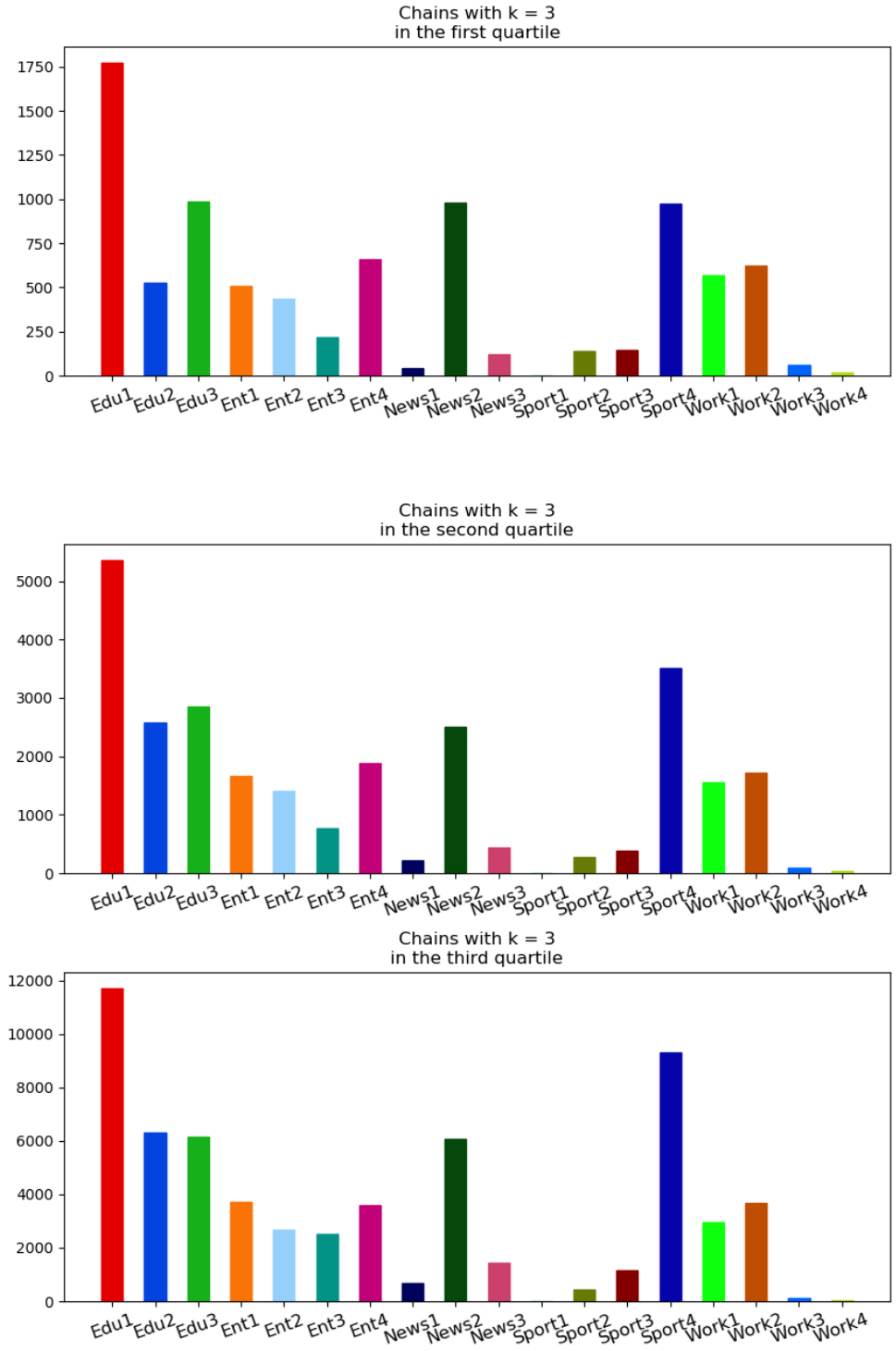
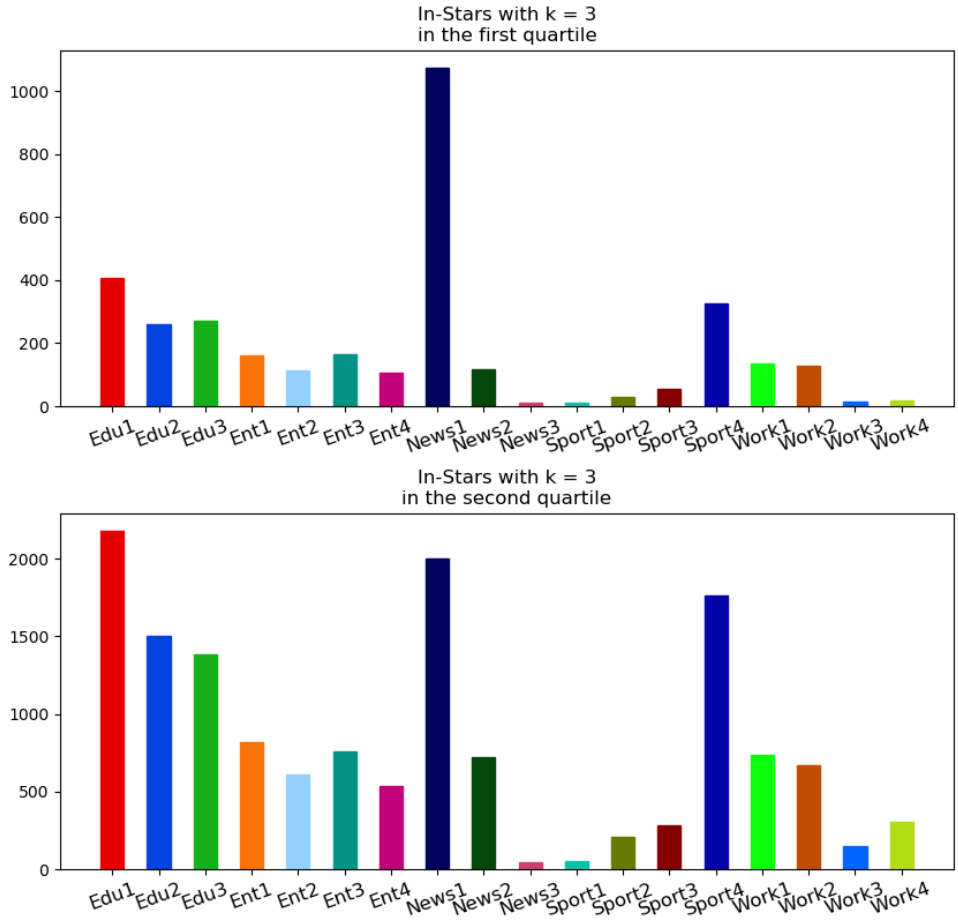
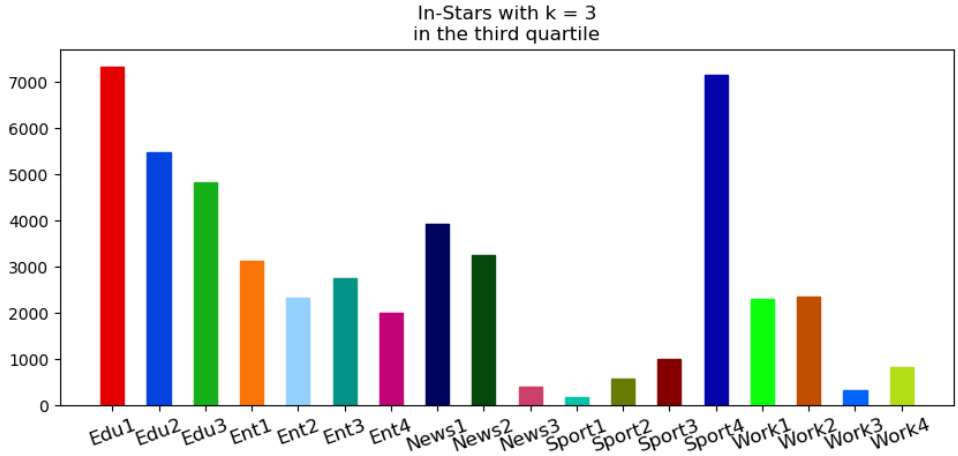
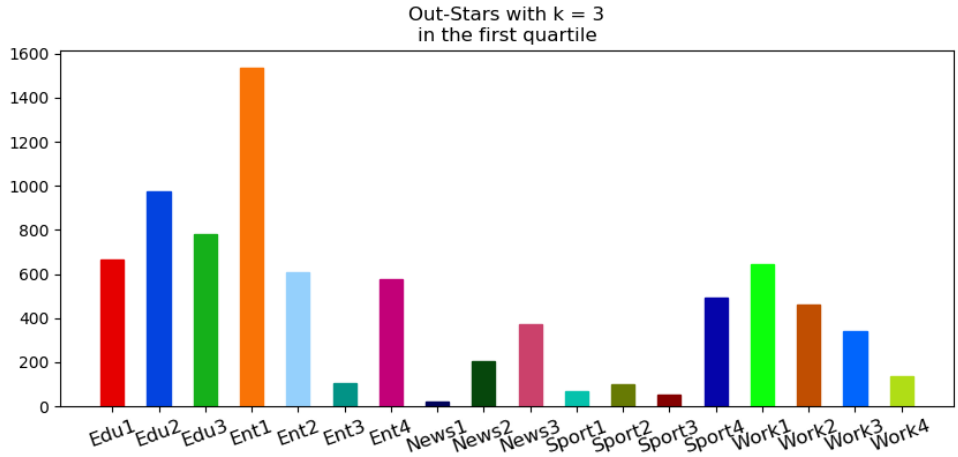


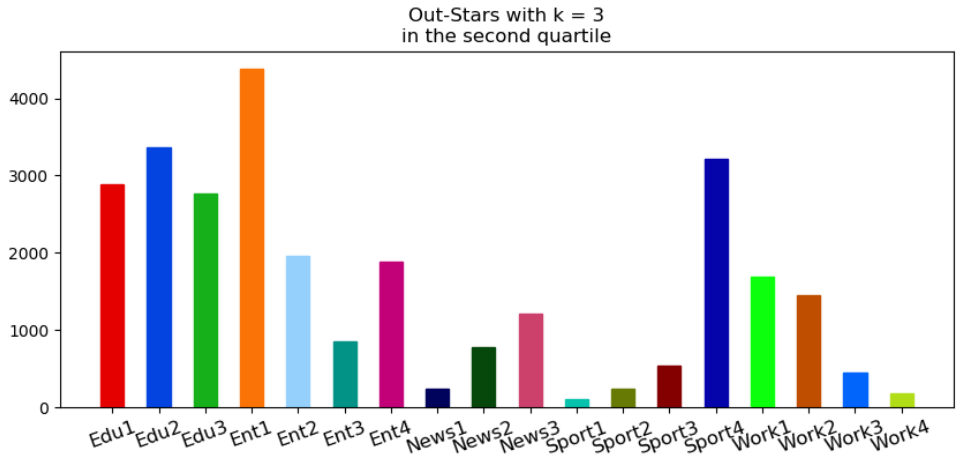
Figure 5.4: The three quartile-based plots related to the results obtained for the chains in the groups

In **Figure 5.4** we can see that 3-interactions long Chains can be found pretty much in all groups, with prevalence in Education and Entertainment ones. This result is reasonable for what concerns Entertainment topics, but at the same time surprising for the other category we mentioned, that seems to generate a lot of fast-paced discussion as well (considering that Education's quartiles do not exceed 10 minutes). A peak is reached in Edu1, that comes to have a bit less than 12000 motifs of this kind in the third quartile.




In-Stars with k = 3
in the third quartile
7000
6000
5000
4000
3000
2000
1000
0
Edu1
Edu2
Edu3
Ent1
Ent2
Ent3
Ent4
News1
News2
News3
Sport1
Sport2
Sport3
Sport4
Work1
Work2
Work3
Work4


Out-Stars with k = 3
in the first quartile
1600
1400
1200
1000
800
600
400
200
0
Edu1
Edu2
Edu3
Ent1
Ent2
Ent3
Ent4
News1
News2
News3
Sport1
Sport2
Sport3
Sport4
Work1
Work2
Work3
Work4


Out-Stars with k = 3
in the second quartile
4000
3000
2000
1000
0
Edu1
Edu2
Edu3
Ent1
Ent2
Ent3
Ent4
News1
News2
News3
Sport1
Sport2
Sport3
Sport4
Work1
Work2
Work3
Work4

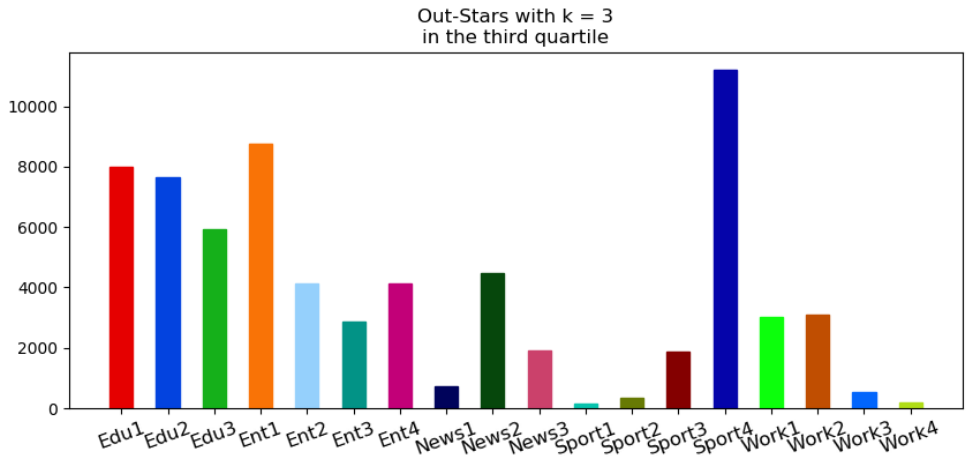


Figure 5.5: The three quartile-based plots related to the results obtained for the two kinds of stars in the groups

We decided to group together (in **Figure 5.5**) In-Stars and Out-Stars' plots in order to detect similarities and differences among them. Both motifs already reach very high numbers from the first quartile, and grow in similar ways through the quartiles. The peaks aren't reached by the same groups, but there seems to be an overall predominance in the Education field. The fact that Education sometimes prevails over the other categories is again very interesting.
Finally, **Figure 5.6** contains the bar diagrams related to One Way Couples. It is immediately noticeable that Edu1, one more time, is the most populous group in terms of this motif; its number grows very rapidly through the three quartiles, keeping itself at distance from the numbers of the other groups. Apart from this peak, the presence of this motif looks quite homogenuous in the groups, with the exception of the Sport ones that seem to be less prone to give birth to One-Way Couples.

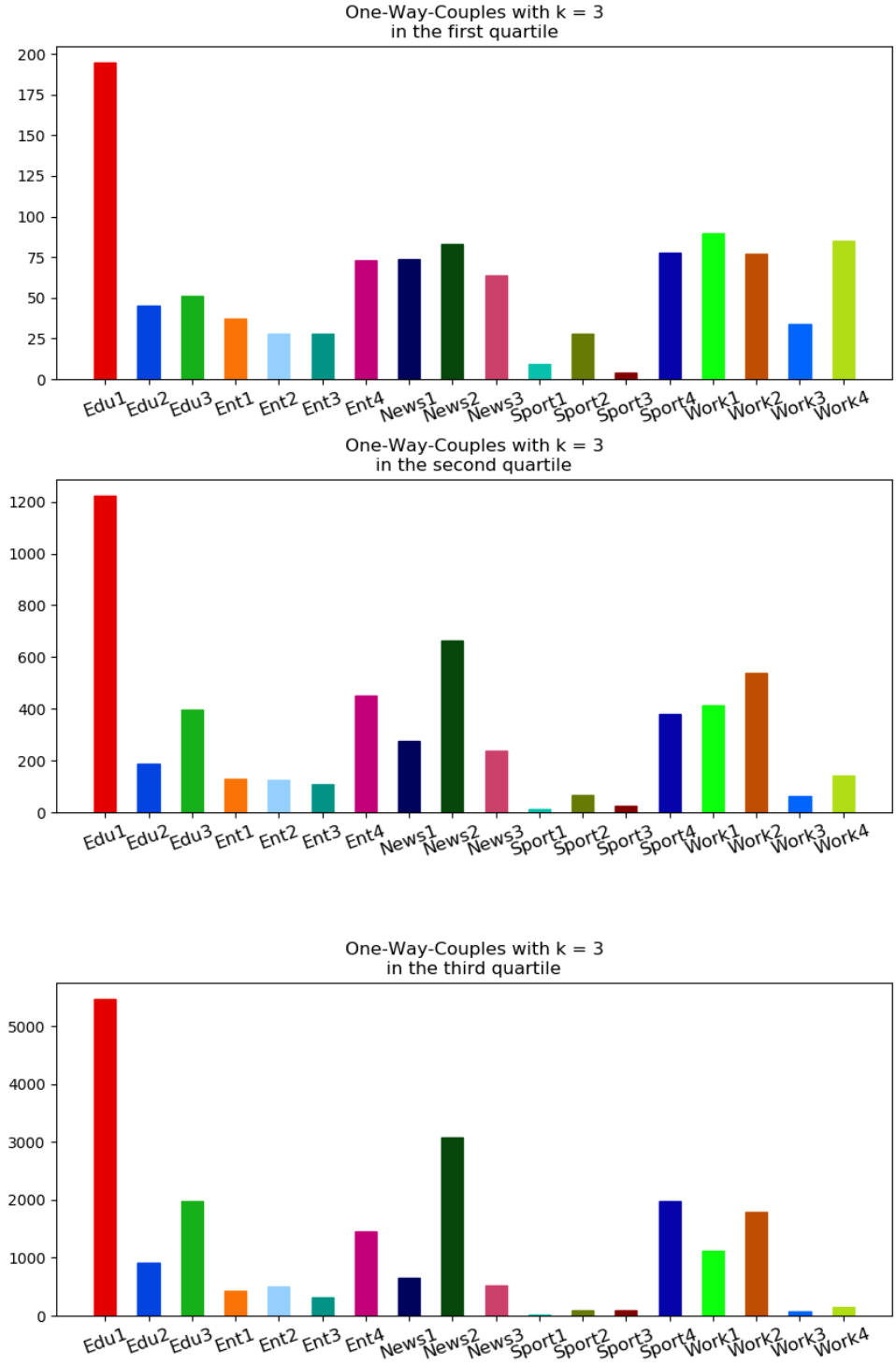


Figure 5.6: The three quartile-based plots related to the results obtained for the one-way couples in the groups

All the quartile diagrams are available in **Appendix A**.

5.2 Highest k

We extracted the maximum value reached by k for all motifs in the groups. **Table 5.1** and **Table 5.2** group together these values (related to the time window sliding approach) for the Post Graph and User Graph. It is immediately noticeable how the maximum k grows over the three quartiles, and how in the User Graph it is generally higher. The reason of it is that the Post Graph motifs are captured and provide a lower bound (User Graph's events are the same ones as Post Graph's without the type label), and in additions there are all the inter-posts events that aren't taken into consideration in the Post Graph. This doesn't always apply to motifs that contain a succession of replies (particularly chains and ping-pongs), because of the loss of intra-post information we wrote about in **Chapter 3** when we described the User Graph. The case of ping-pongs makes it even more evident: in the User Graph they rarely reach the length of 4, whereas in the Post Graph they get to be even 8 or 10 interactions long. This means that this kind of motif is much more present in the scope of individual posts, and this makes sense since we're talking about a situation of back-and-forth. In Work4, for example, in the Post Graph we found ping-pongs up to 6 interactions long, whereas in the User Graph there were no ping-pongs at all, meaning that they must be predominantly composed by replies. On the other hand, a motif that reaches high lengths in the User Graph, and not so much in the Post Graph, is the One Way Couple. This means that the "stalker" behaviour tends to occur more in different posts: users tend to interact with the same person in a one sided way under various discussions. While the User Graph is full of one-way couples with considerable length, in the Post Graph many are the groups for which we weren't able to find any motifs of that kind, not even in the third quartile.
Another aspect that may be taken into consideration is the lack of substantial correlation between the daily activity density of groups and the maximum k reached by motifs in them. Let's take News1, Work3 and Sport3 as our examples. For the first two just mentioned groups we have been able to detect quite long motifs: in the first case we even have a peak of a 73 interactions long in-star in the Post Graph (and 170 interactions long in the User Graph). However, if we take a look at **Table 4.3**, both groups are not that active compared to other ones like Sport3. Although the latter, according to our analysis, is the one that generates the biggest amount of daily activity, its motifs don't reach significant lengths at all. The same fact applies for Ent3, another pretty busy group. Discussions in very active groups tend to exhaust their liveliness in the short term, and thus their motifs don't grow much across the

quartiles.

Highest k in the Post Graph																		
Group	Chains			PPongs			IStars			OStars			OWC			PTri		
Quartiles	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III
Edu1	4	5	6	2	4	8	3	4	7	4	5	9	-	3	6	-	2	6
Edu2	4	4	6	2	2	6	3	4	9	6	8	18	-	-	-	-	2	4
Edu3	4	5	6	2	4	10	3	5	14	4	5	8	-	-	6	-	2	10
Ent1	3	4	6	-	2	4	-	4	7	7	10	14	-	-	3	-	2	4
Ent2	5	5	6	2	4	4	3	7	9	4	6	10	-	3	3	2	2	4
Ent3	4	4	6	-	2	2	-	3	4	4	4	6	-	-	-	-	-	2
Ent4	4	4	5	2	4	8	3	4	8	6	8	8	-	4	6	-	2	6
News1	-	3	3	-	2	8	-	14	73	3	4	6	-	-	4	-	2	6
News2	3	3	4	-	-	-	-	-	-	5	5	9	-	-	-	-	-	-
News3	3	3	4	2	4	6	3	4	6	4	6	12	3	4	13	-	4	6
Sport1	-	-	-	-	-	-	3	3	3	4	7	9	-	-	3	-	-	-
Sport2	3	4	5	2	4	6	3	6	22	4	5	7	-	11	13	2	4	6
Sport3	3	3	4	-	-	2	-	3	3	4	4	6	-	-	-	-	-	2
Sport4	3	4	6	-	2	6	-	3	7	5	5	8	-	-	-	-	-	4
Work1	5	5	6	2	4	8	3	5	8	8	15	15	-	3	13	-	4	8
Work2	4	4	5	2	4	8	3	6	13	4	9	14	-	3	10	2	2	8
Work3	3	4	7	2	4	4	4	6	12	10	15	24	-	6	6	2	2	4
Work4	-	3	3	4	6	6	3	7	16	11	20	20	-	-	-	-	4	4

Table 5.1: The maximum length found for all our temporal motifs in the various groups modeled as Post Graphs

Highest k in the User Graph															
Group	Chains			PingPongs			IStars			OStars			OWCouples		
Quartiles	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III
Edu1	4	5	6	2	2	4	5	6	8	4	5	9	6	9	16
Edu2	4	4	5	2	2	2	6	8	14	6	8	18	6	11	17
Edu3	5	5	6	2	4	4	5	8	15	4	5	8	3	6	10
Ent1	4	4	5	2	2	2	5	7	11	7	10	14	4	6	7

Ent2	4	4	6	2	2	2	5	7	13	4	6	10	5	5	8
Ent3	3	3	4	2	2	2	5	7	10	4	4	6	4	8	15
Ent4	4	4	5	2	2	2	4	6	8	6	8	8	4	6	13
News1	3	4	4	-	-	2	20	62	170	3	4	6	5	11	14
News2	4	4	4	-	-	2	4	5	6	5	5	9	5	8	12
News3	4	4	5	2	2	4	3	3	5	4	6	13	5	10	15
Sport1	-	-	-	-	-	-	4	6	7	4	7	9	4	4	4
Sport2	4	4	6	2	2	2	4	5	21	4	5	7	5	12	14
Sport3	3	3	4	2	2	2	4	5	8	4	4	6	3	4	5
Sport4	4	4	5	2	2	2	5	6	11	5	5	8	5	8	16
Work1	5	6	7	2	2	2	5	7	14	8	15	15	5	10	28
Work2	4	5	6	2	2	4	5	6	18	4	9	14	4	8	14
Work3	4	4	4	-	-	-	3	6	13	10	13	20	5	9	9
Work4	3	4	4	-	-	-	4	9	18	6	6	7	7	16	16

Table 5.2: The maximum length found for all our temporal motifs in the various groups modeled as User Graphs

Table 5.3 compares, for all the groups, the maximum values of k in the Post Graph and in the User Graph for both the time window approaches used to find the motifs. The results confirm our initial hypothesis that the length of motifs found with the snapshot approach would be lower or equal to the one found with the time window sliding, because of the clean cut of the snapshot that may have interrupted a motif that would otherwise have been longer.

Highest k with the two time window approaches				
Group	**Time window sliding approach**		**Snapshots approach**	
	Post Graph	User Graph	Post Graph	User Graph
Edu1	9	16	9	8
Edu2	18	18	17	16
Edu3	14	15	10	13
Ent1	14	14	14	13
Ent2	10	13	10	13

Ent3	6	15	5	8
Ent4	8	13	8	8
News1	73	170	73	168
News2	9	12	9	8
News3	13	15	13	12
Sport1	9	9	9	7
Sport2	22	21	22	21
Sport3	6	8	6	8
Sport4	8	16	8	10
Work1	15	28	15	14
Work2	14	18	13	15
Work3	24	20	24	19
Work4	20	18	20	15

Table 5.3: Comparison of the general maximum lengths of our temporal motifs found by using the two different time window approaches

5.3 Correlation between groups

In this section we analyze if the results we found constitute a connection between some groups and, if so, what are the ones that have the highest correlation (and if significant differences exist regarding the correlation between different pairs of groups).
The tool we used in order to do so is KNIME Analytics, with the procedure we described more in depth in **Chapter 4**. Specifically, we exploited the output of the Linear Correlation node, which is the Correlation Table. It is a $n \times n$ matrix C (where n is the number of groups, or the cardinality of a subset of groups); every item in position C_{ij} is the correlation coefficient between the group i and the group j, and varies between -1 (strongly negative linear correlation) to 1 (strongly positive linear correlation), with 0 meaning no correlation whatsoever. Trivially, $i = j \implies C_{ij} = 1$. Coefficients are represented by colors: the stronger blue-colored is a cell, the higher positive correlation exists between two groups; on the other hand, when two groups have a less than zero coefficient, its colors varies on the shades of red, where bright red stands for a coefficient that equals -1. The more intense are the colors, the stronger is correlation between them. A white table cell means that there is no correlation

between the two groups that are on its row and column.
For each quartile we elaborated a table based on the highest k found in the groups for each motif. Particularly, we used the User Graph results, with the exception of the ping-pong and the p-triangle, the two temporal motifs that are more meaningful for what concerns the Post Graph. **Figure 5.7**, **Figure 5.8** and **Figure 5.9** represent the Correlation Tables for the first, the second and the third quartile respectively. The first thing to notice is that the higher is the quartile, the generally lower becomes the correlation between the groups. This observation is fairly trivial, since in the first quartile, being it quite temporally short, motifs tend to stabilize their length on small values, while in longer time windows they have the chance to differ more. Furthermore, in all quartiles groups are generally more positively correlated if they belong to the same category. This is visible the most in the figure representing the third quartile, where the differences become more evident. Some groups have high positive correlation despite not belonging to the same categories. While it is certainly not easy to trace what could be the key factors at the base of this factor, we were able to identify some possible elements. Work3 and Sport1, for example, have very similar numbers of posts, comments and average daily posts and comments, as we can see in **Table 4.3**. This could mean that similar activity density in different groups can lead to the development of temporal motifs having close lengths.

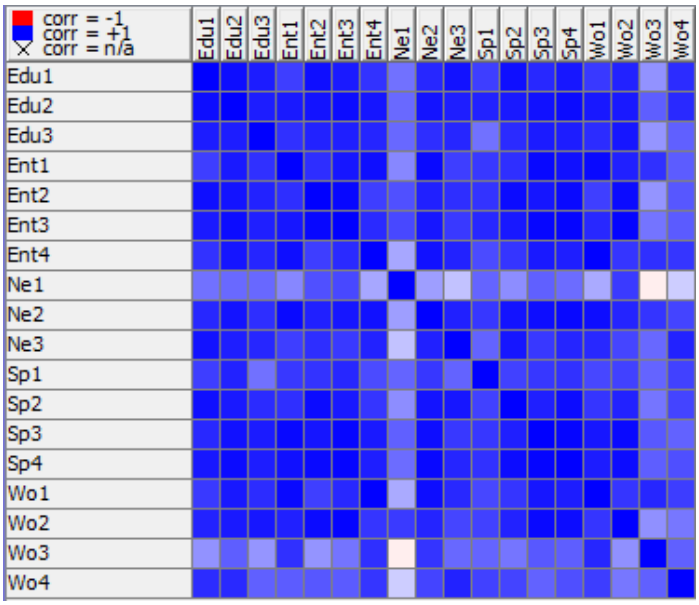


Figure 5.7: Correlation Table for the first quartile

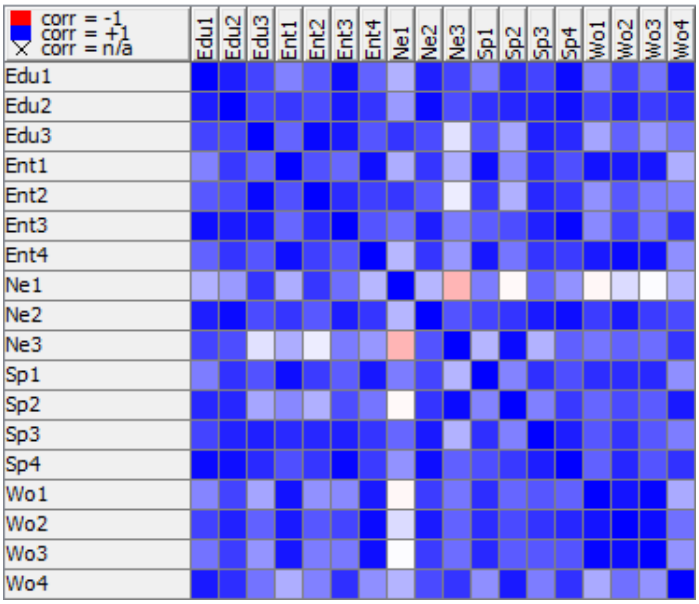


Figure 5.8: Correlation Table for the second quartile

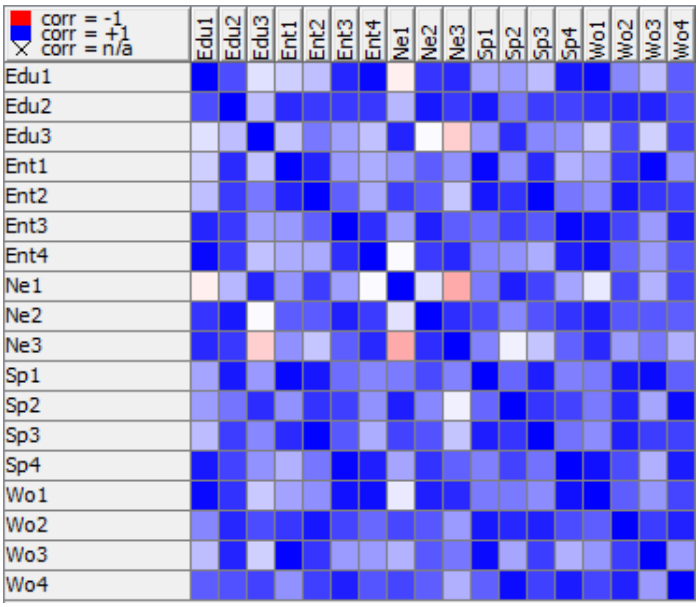


Figure 5.9: Correlation Table for the third quartile

Chapter 6

Conclusions

In this work we studied a set of Facebook groups belonging to different categories to study what communication patterns develop the most into them. We particularly focused on two aspects of temporal motifs: their number of occurrences in the various groups and their lengths. Moreover, we also performed an analysis from a broader point of view, looking for common aspects and differences in the groups, trying to figure out if having the same underlying general topic would result in the development of similar temporal motifs in terms of size and frequency. We made a further comparison between the results we found with the two different formulations of our temporal graphs, which are the Post Graph and the User Graph. Since the Post Graph only contains separate subgraphs that are relative to a single post, while the User Graph includes all users (as nodes) and all the interactions between them, we trivially expected to have longer and more frequent temporal motifs in the User Graph. Finally, we compared what we obtained by the two different ways we treated the time window throughout the temporal life span of the groups. We either used the sliding approach or snapshots approach, expecting the first one to capture longer motifs, since the snapshots approach can cut motifs that may be longer if looked for with the other approach.

The general tendency, regardless of the use of a specific graph formulation or a time window approach, is that the growth of the time window almost always turns out in a decisive increase of the number of temporal motifs. This probably depends on the fact that the values we used as our time windows were the quartiles of the interactions inter-arrival time; in the longest time window, that corresponds to the third quartile, our algorithms managed to catch the 75% of the interactions, thus trivially more motifs compared to the other two shorter time windows.

Our expectations regarding the Post Graph and the User Graph were fully satisfied.

In fact, using the second kind of temporal graph formulation, the number of temporal motifs is much higher. The same happened for what we anticipated in relation to the utilisation of the two different time window approaches: with the snapshots approach the maximum length reached by temporal motifs was generally lower, and the sliding approach resulted in finding way more motifs of a specific length.
The category that seems to prevail on the others is surprisingly the Education one. Considering temporal motifs with standard lengths (the ones for which they are generally more frequent), Education groups tend to have peaks in all the quartiles, i.e. users tend to generate communication patterns more frequently in them. With the exception of these peaks and some smaller ones in other groups, generally groups of all categories tend to be quite correlated, as we can see from the Correlation Tables we showed in **Chapter 5**.

6.1 Future Works

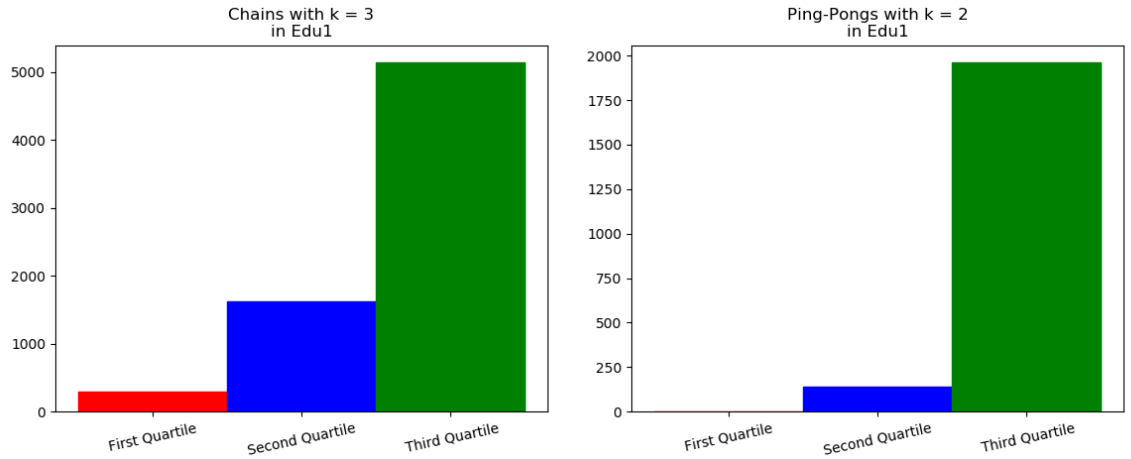
In future works we would surely like to investigate more in depth differences and similarities between temporal motifs in groups that belong to different categories. This thesis didn't consider the contents of the posts, comments and replies that form the motifs; it would be interesting to analyze what particular figures, like influencers, write, and see if there are any recurring words or expressions using text mining techniques. Moreover, we concentrated on a core group of just five temporal motifs, that were the most common ones in literature. We would like to find more peculiar ones that represent other possible common users' behaviors. Finally, we plan to find a new, improved model for the time window handling. In this work we explained how both the time window sliding approach and the snapshots approach have disadvantages: the first one tends to count each motif many times, while the second one, on the other hand, is likely to cut some motifs' lengths. The ideal model would be an adaptive one, that adjusts on the basis of the activity density.

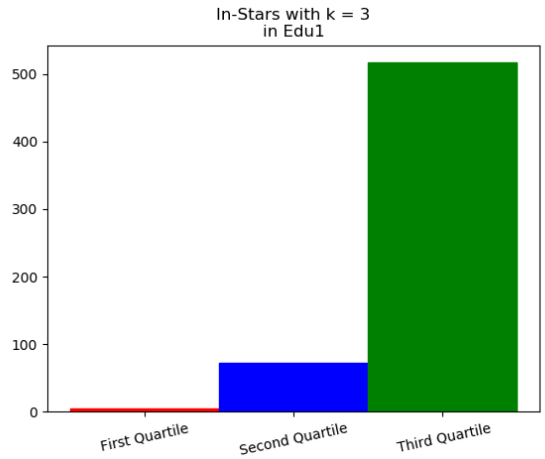
Appendix A

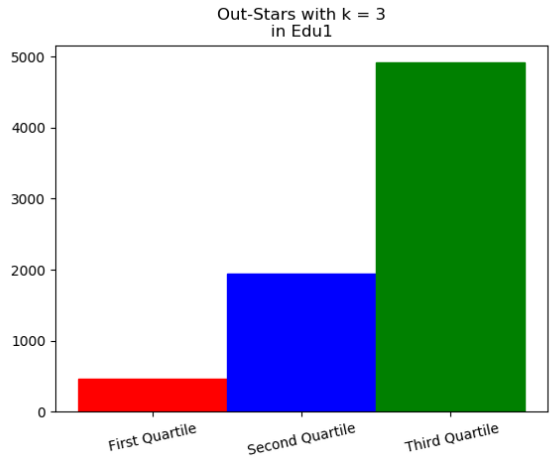
This appendix contains all the plots we elaborated on the basis of our results.

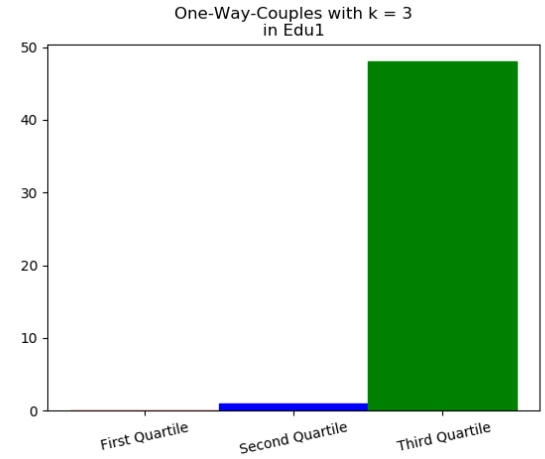
A.1 Group-based Diagrams

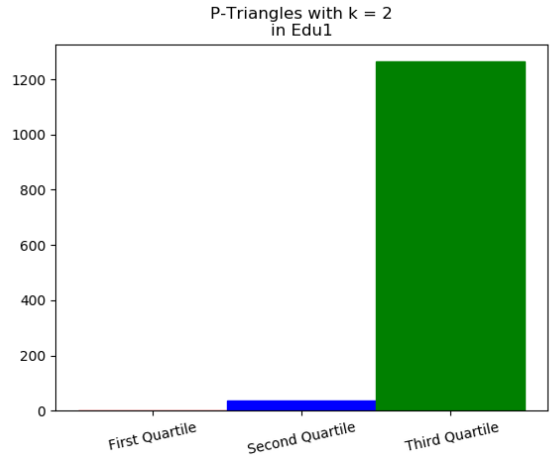
This section contains all the group-based plots, obtained by using the two graphs formulations and the two time window approaches. Each group has its own diagram that show the results obtained by counting a particular temporal motif using a specific graph formulation and with one of the two time-window approaches. Every diagram contains the results for the three quartiles.
The first set of diagrams, in **Figure A.1**, contains the group-based plots relating to the number of motifs we found in the Post Graph using the time window sliding approach.

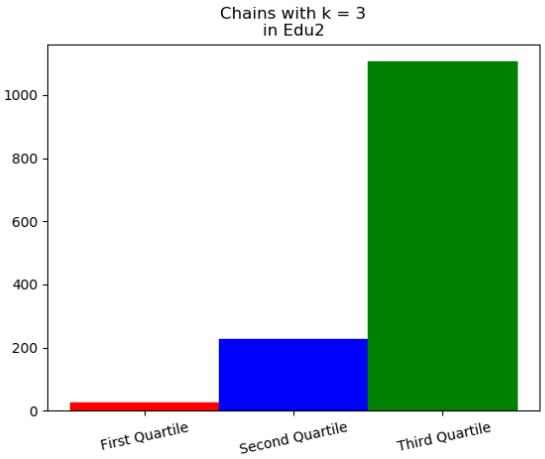


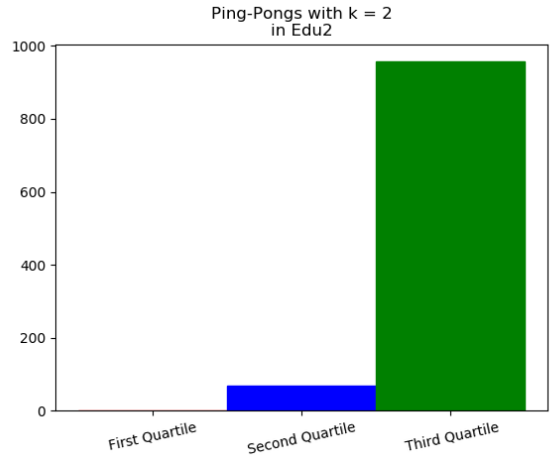

In-Stars with k = 3
in Edu1
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

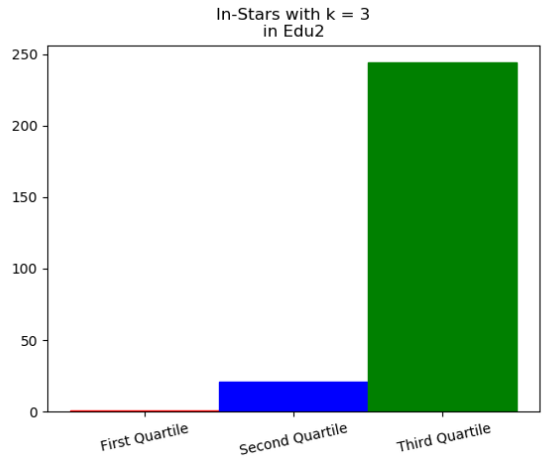

Out-Stars with k = 3
in Edu1
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

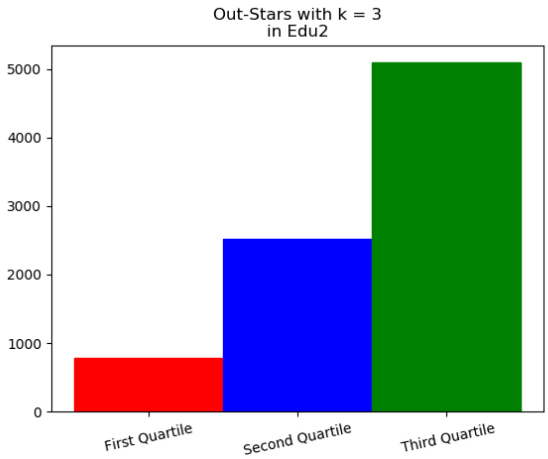

One-Way-Couples with k = 3
in Edu1
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

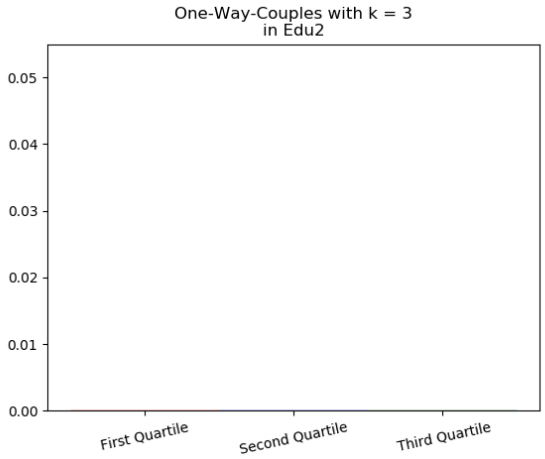

P-Triangles with k = 2
in Edu1
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

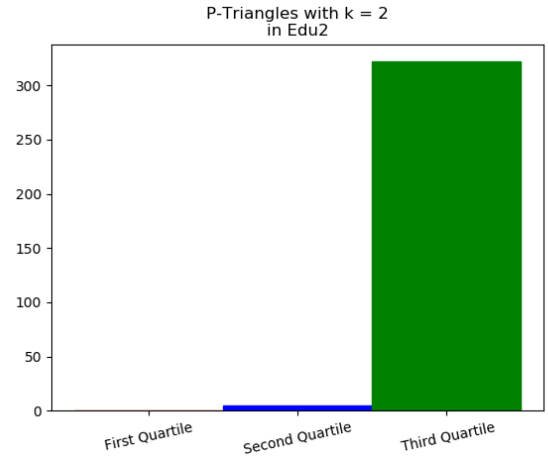

Chains with k = 3
in Edu2
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

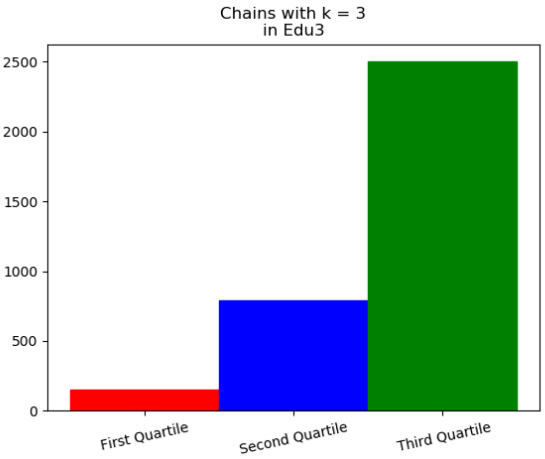

Ping-Pongs with k = 2
in Edu2
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

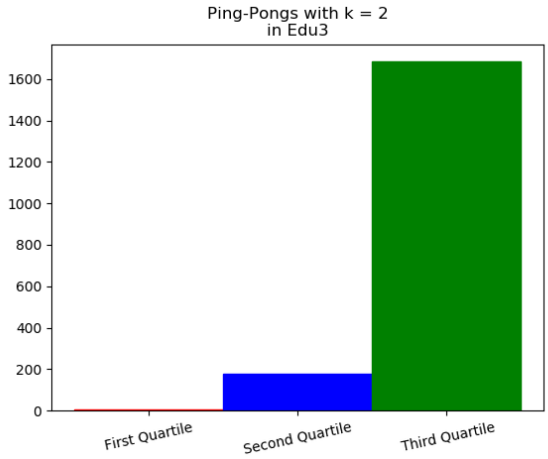

In-Stars with k = 3
in Edu2
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

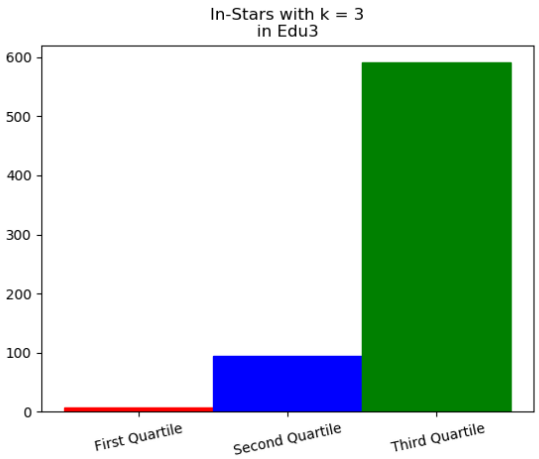

Out-Stars with k = 3
in Edu2
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

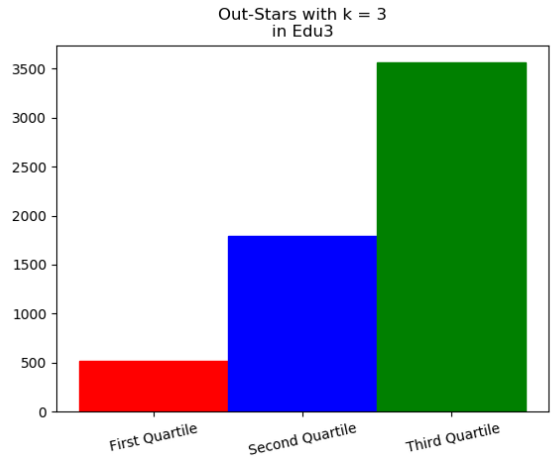

One-Way-Couples with k = 3
in Edu2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

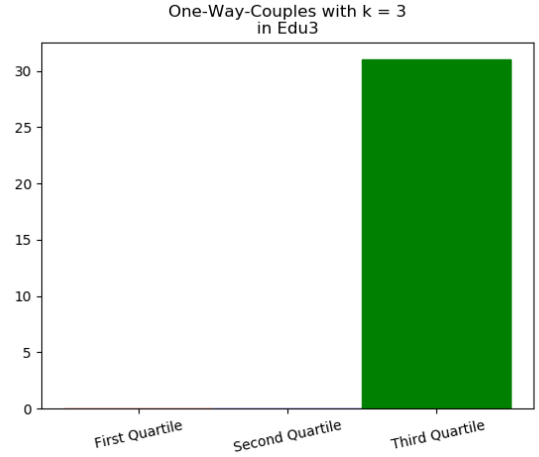

P-Triangles with k = 2
in Edu2
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

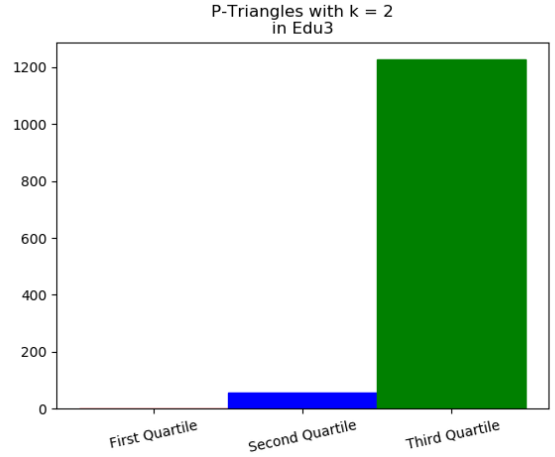

Chains with k = 3
in Edu3
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

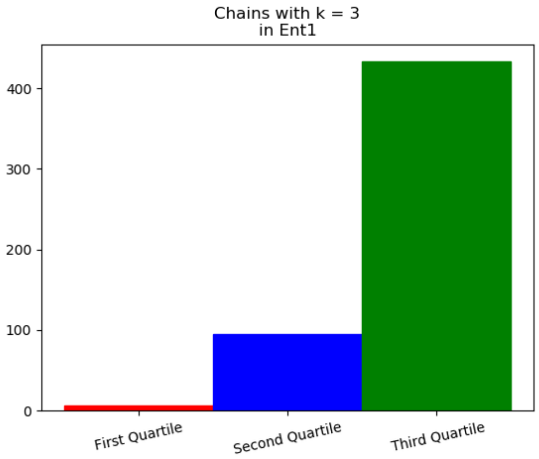

Ping-Pongs with k = 2
in Edu3
1600
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

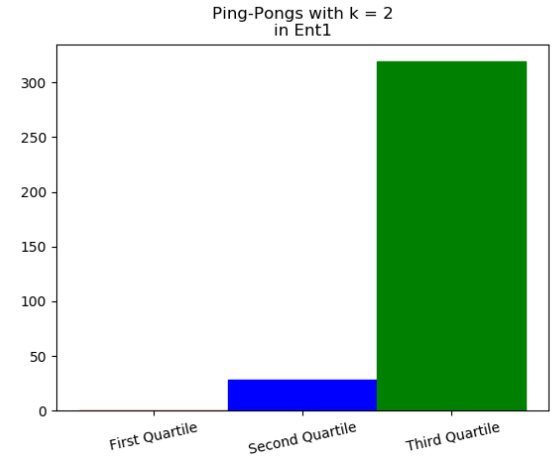

In-Stars with k = 3
in Edu3
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

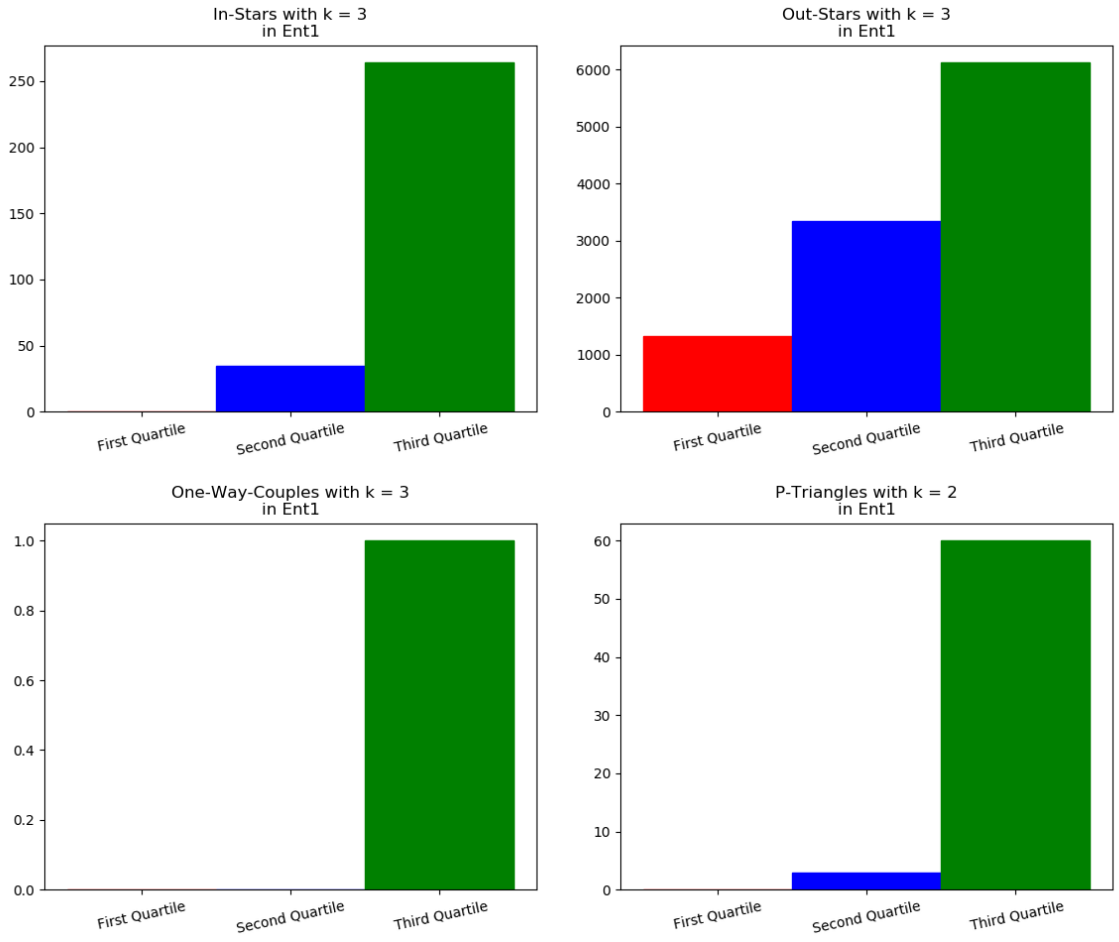

Out-Stars with k = 3
in Edu3
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

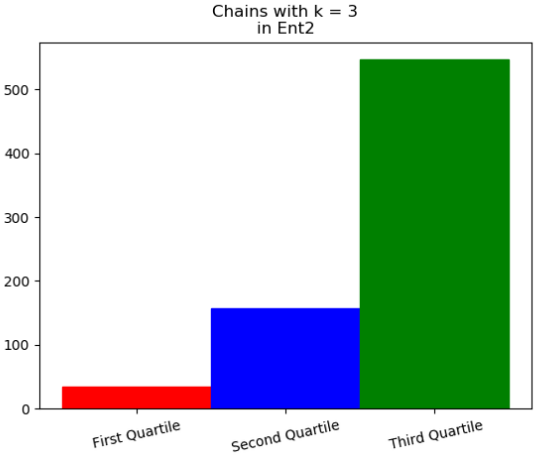

One-Way-Couples with k = 3
in Edu3
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

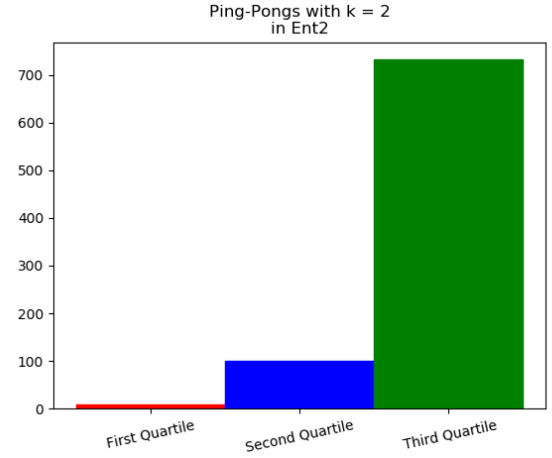

P-Triangles with k = 2
in Edu3
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

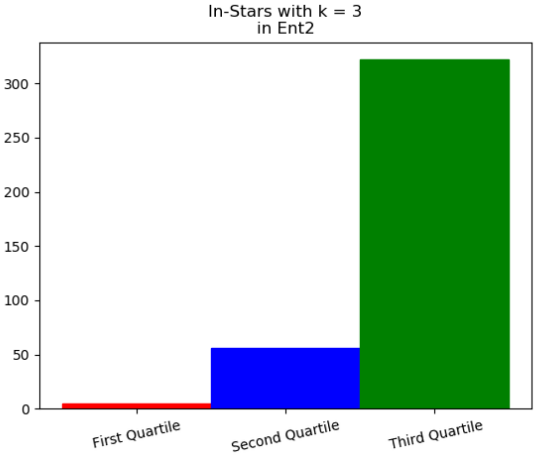

Chains with k = 3
in Ent1
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

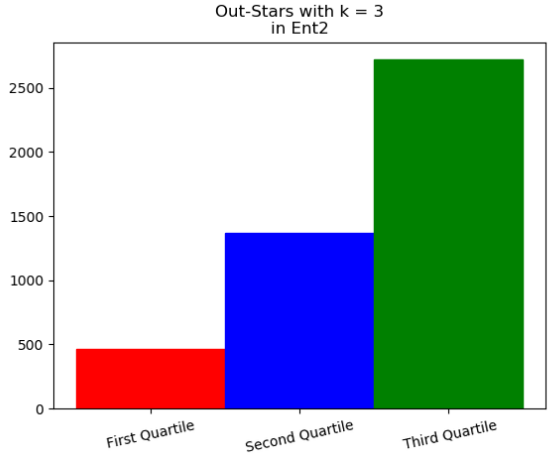

Ping-Pongs with k = 2
in Ent1
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

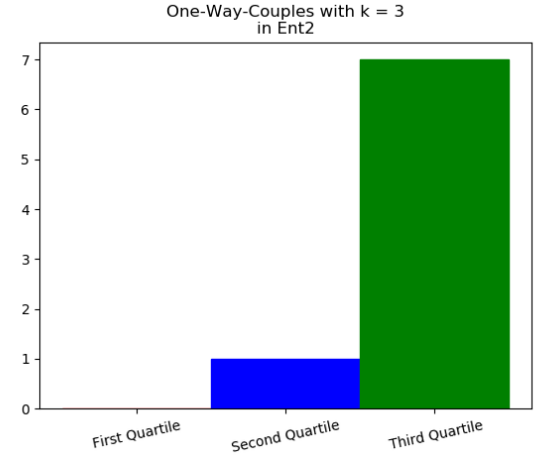

In-Stars with k = 3
in Ent1
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile
Out-Stars with k = 3
in Ent1
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile
One-Way-Couples with k = 3
in Ent1
1.0
0.8
0.6
0.4
0.2
0.0
First Quartile
Second Quartile
Third Quartile
P-Triangles with k = 2
in Ent1
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

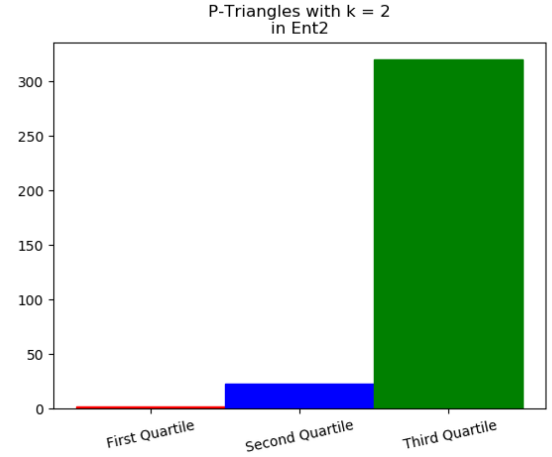

Chains with k = 3
in Ent2
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

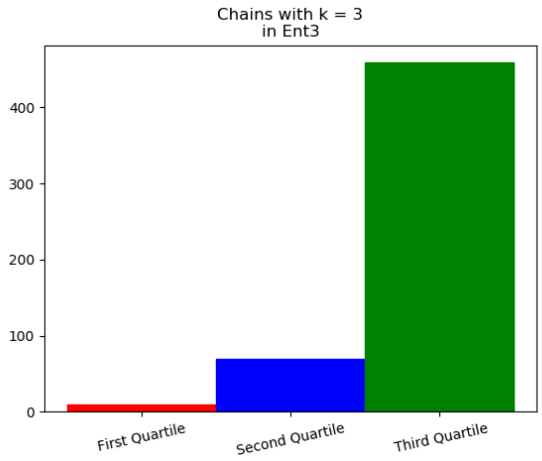

Ping-Pongs with k = 2
in Ent2
700
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

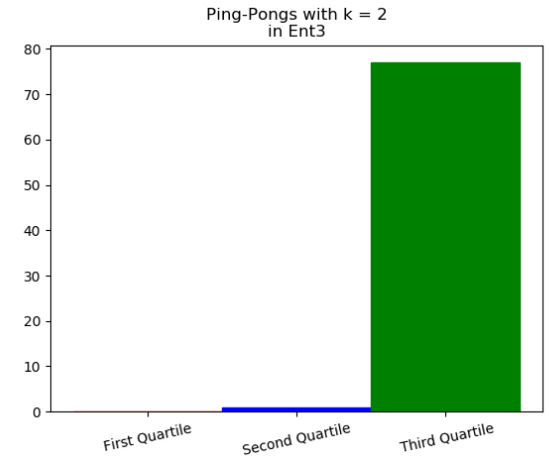

In-Stars with k = 3
in Ent2
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

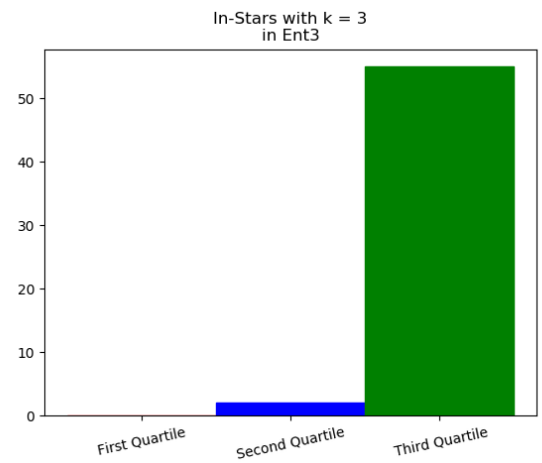

Out-Stars with k = 3
in Ent2
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

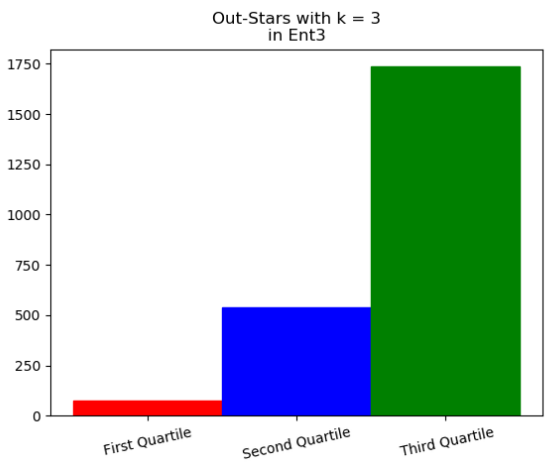

One-Way-Couples with k = 3
in Ent2
7
6
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

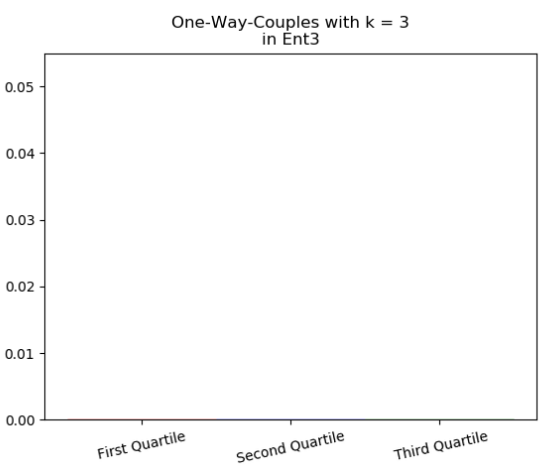

P-Triangles with k = 2
in Ent2
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

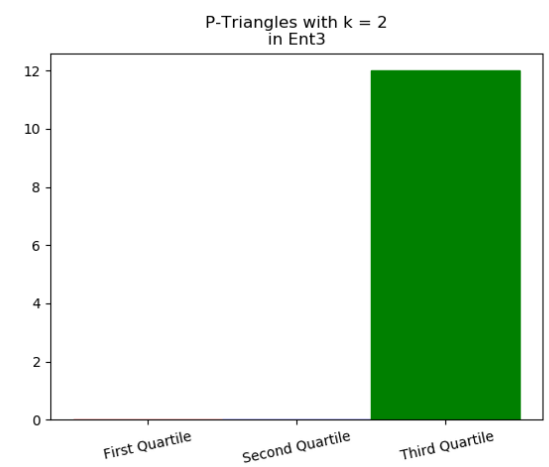

Chains with k = 3
in Ent3
0
100
200
300
400
First Quartile
Second Quartile
Third Quartile

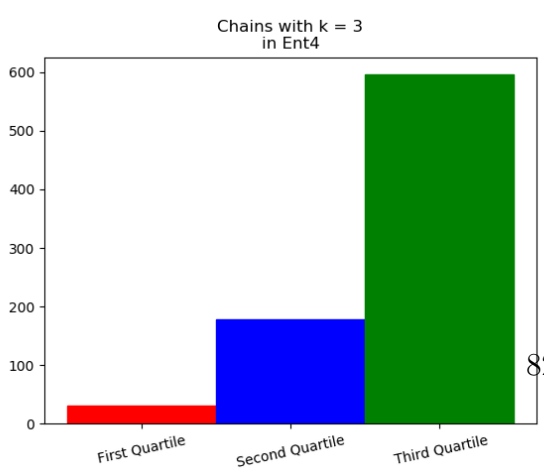

Ping-Pongs with k = 2
in Ent3
0
10
20
30
40
50
60
70
80
First Quartile
Second Quartile
Third Quartile

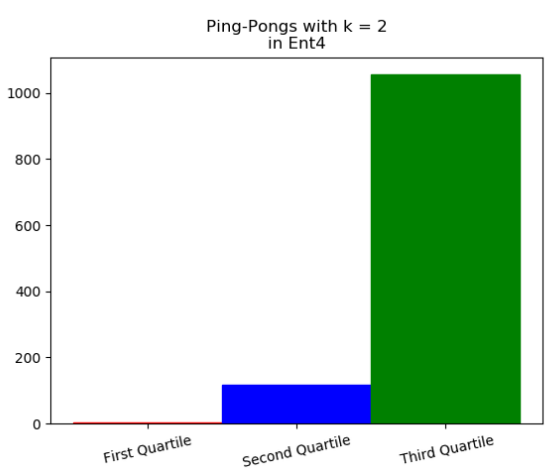

In-Stars with k = 3
in Ent3
0
10
20
30
40
50
First Quartile
Second Quartile
Third Quartile

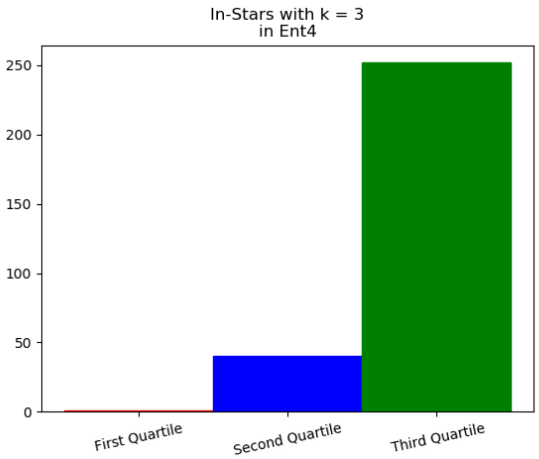

Out-Stars with k = 3
in Ent3
0
250
500
750
1000
1250
1500
1750
First Quartile
Second Quartile
Third Quartile

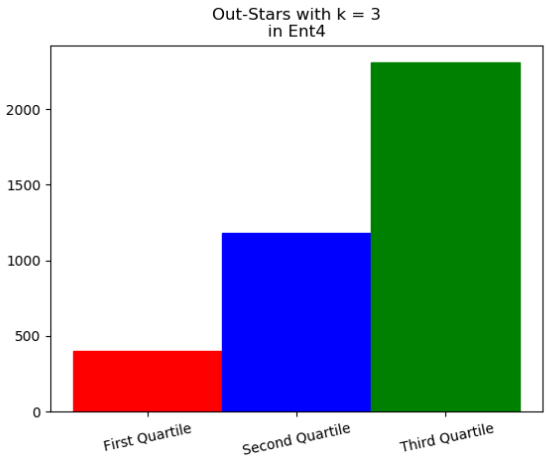

One-Way-Couples with k = 3
in Ent3
0.00
0.01
0.02
0.03
0.04
0.05
First Quartile
Second Quartile
Third Quartile

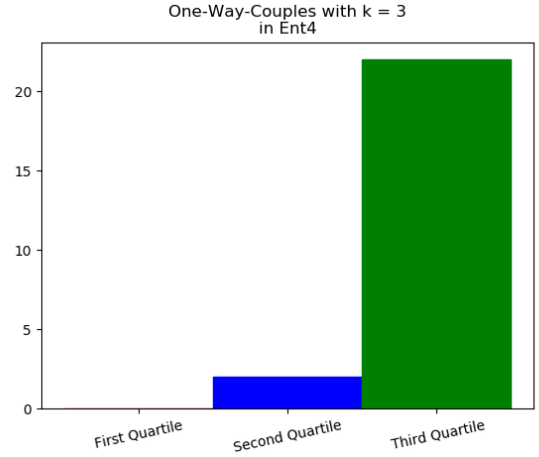

P-Triangles with k = 2
in Ent3
0
2
4
6
8
10
12
First Quartile
Second Quartile
Third Quartile

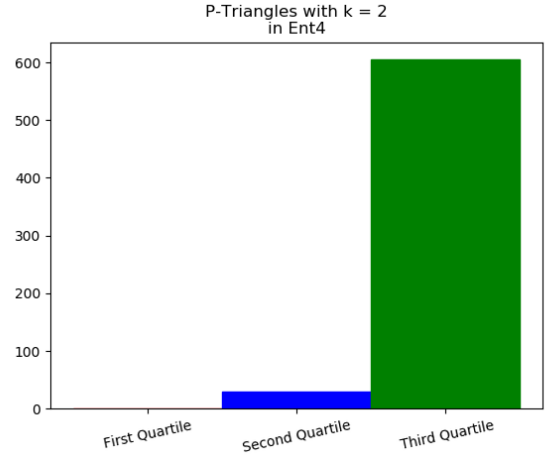

Chains with k = 3
in Ent4
0
100
200
300
400
500
600
First Quartile
Second Quartile
Third Quartile

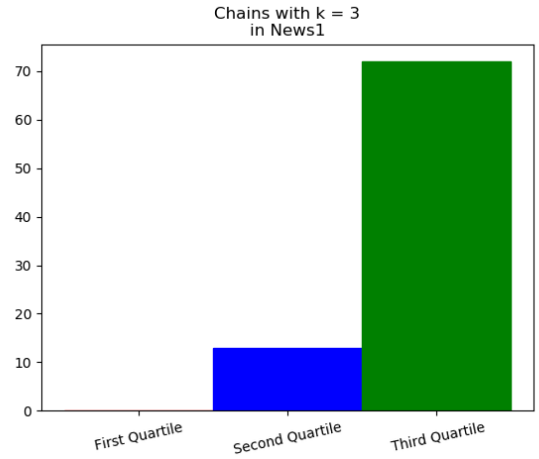

Ping-Pongs with k = 2
in Ent4
0
200
400
600
800
1000
First Quartile
Second Quartile
Third Quartile

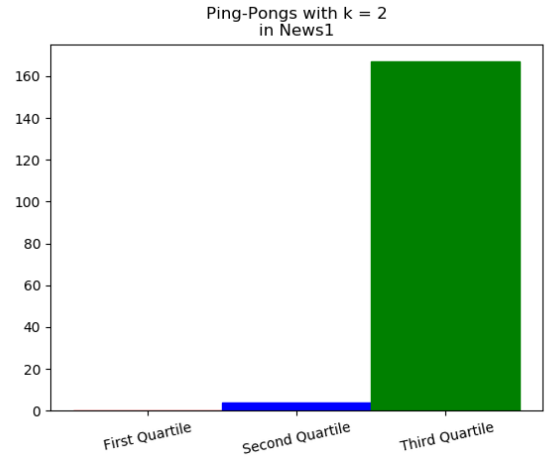

In-Stars with k = 3
in Ent4
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

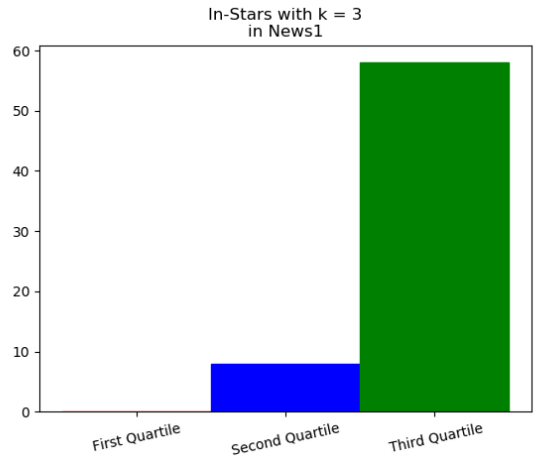

Out-Stars with k = 3
in Ent4
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

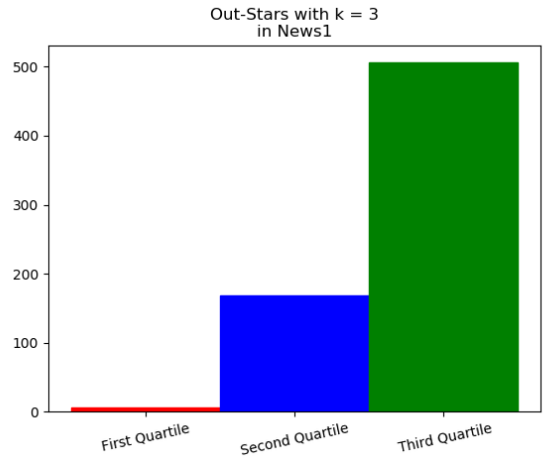

One-Way-Couples with k = 3
in Ent4
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

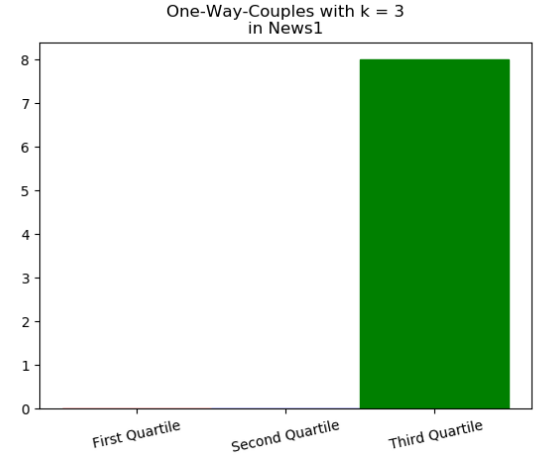

P-Triangles with k = 2
in Ent4
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

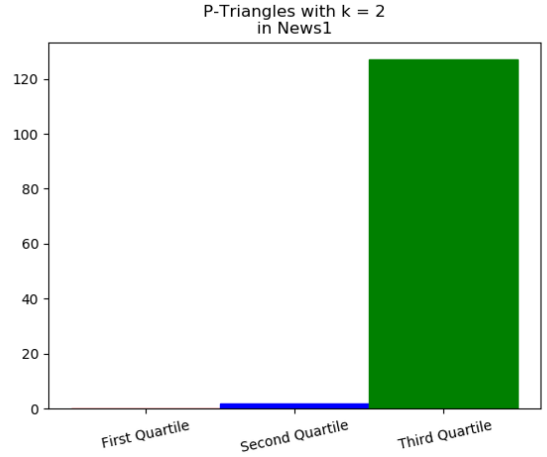

Chains with k = 3
in News1
70
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

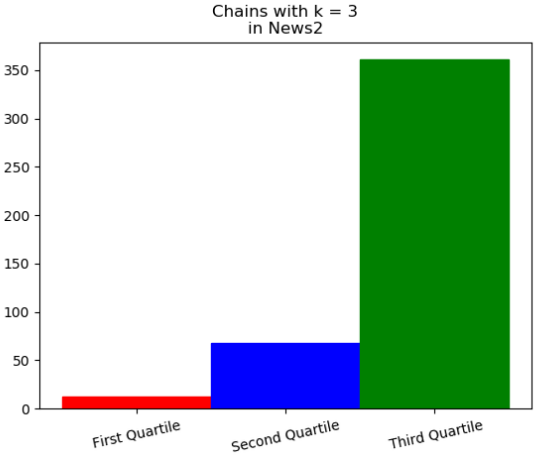

Ping-Pongs with k = 2
in News1
160
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

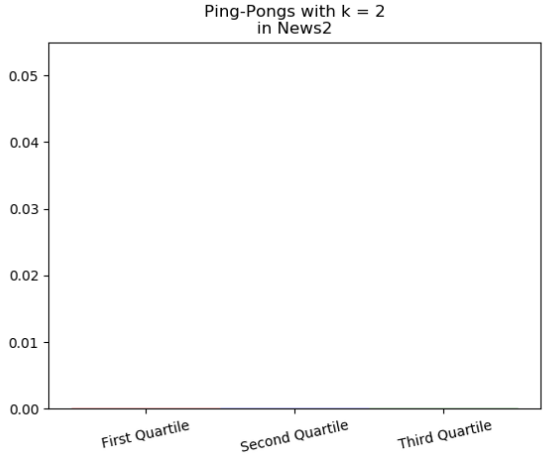

In-Stars with k = 3
in News1
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

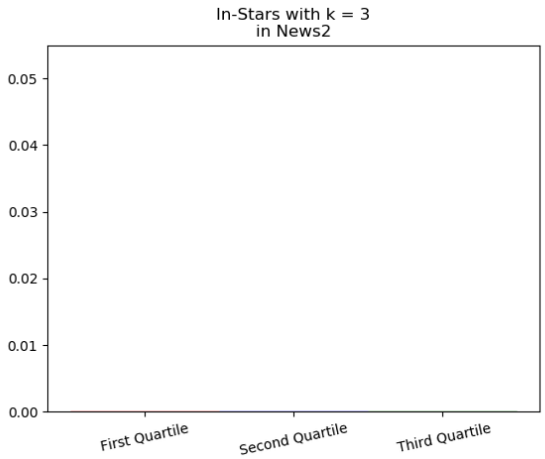

Out-Stars with k = 3
in News1
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

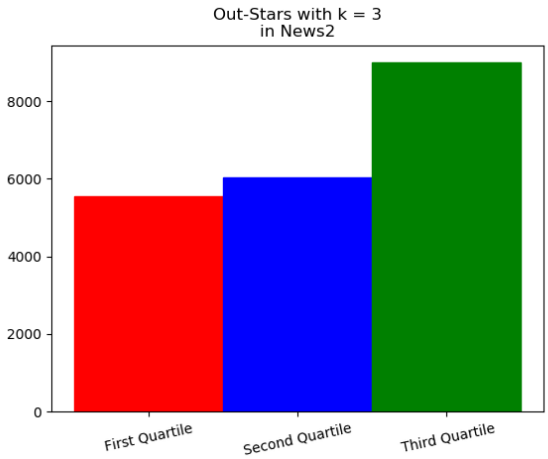

One-Way-Couples with k = 3
in News1
8
7
6
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

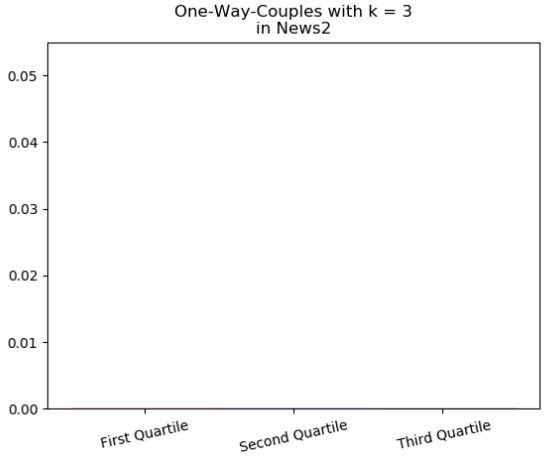

P-Triangles with k = 2
in News1
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

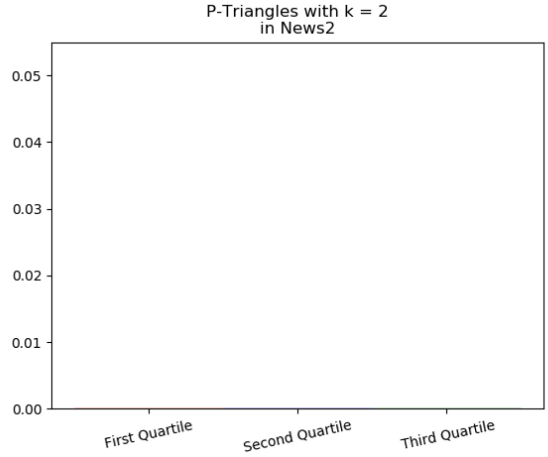

Chains with k = 3
in News2
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

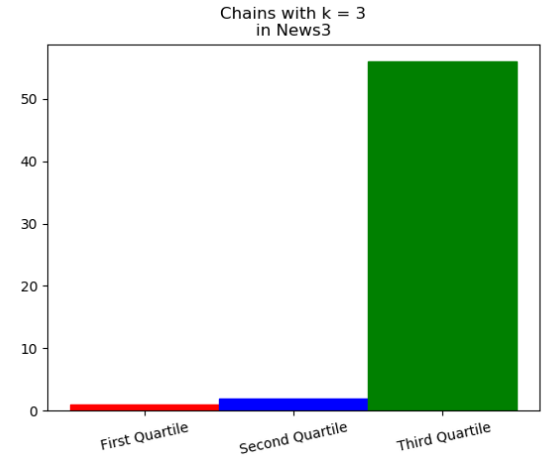

Ping-Pongs with k = 2
in News2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

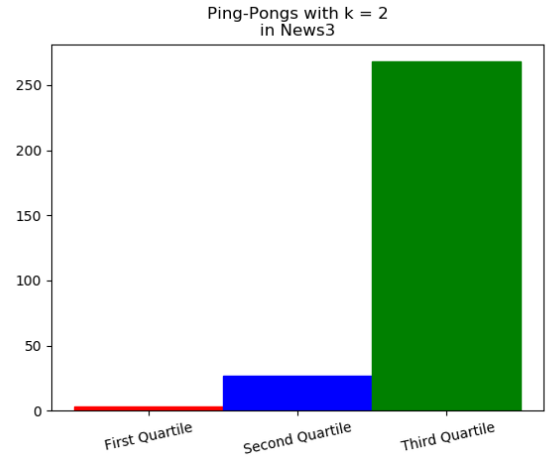

In-Stars with k = 3
in News2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

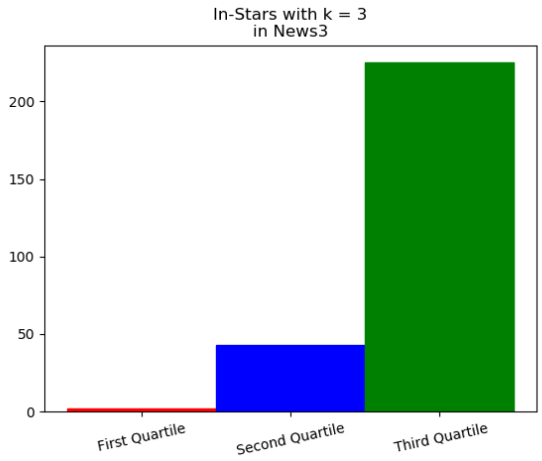

Out-Stars with k = 3
in News2
8000
6000
4000
2000
0
First Quartile
Second Quartile
Third Quartile

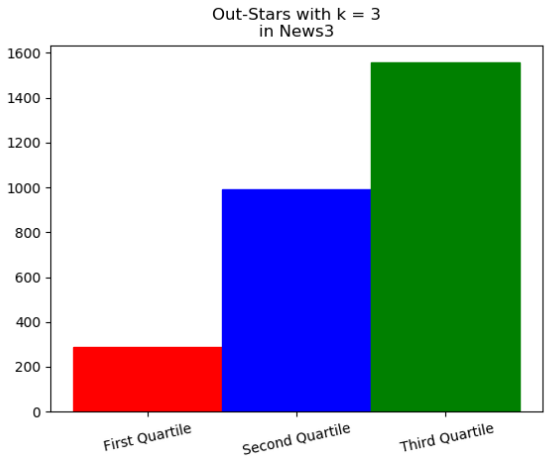

One-Way-Couples with k = 3
in News2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

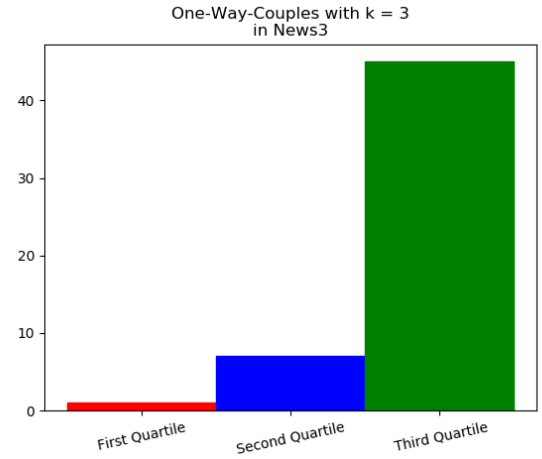

P-Triangles with k = 2
in News2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

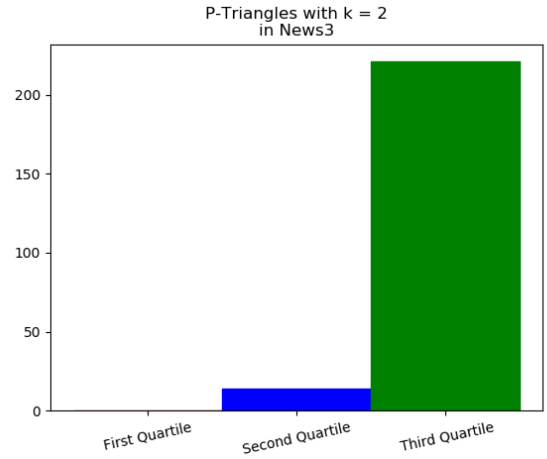

Chains with k = 3
in News3
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

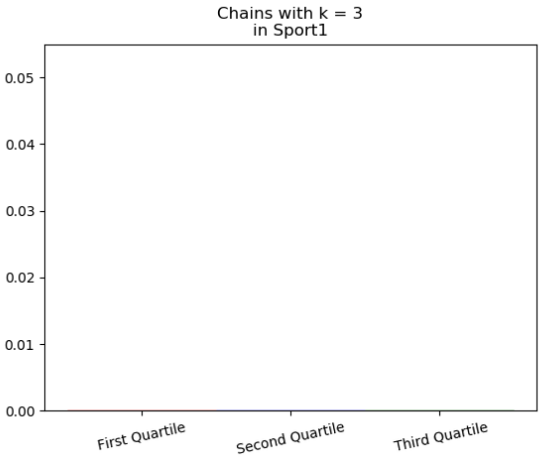

Ping-Pongs with k = 2
in News3
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

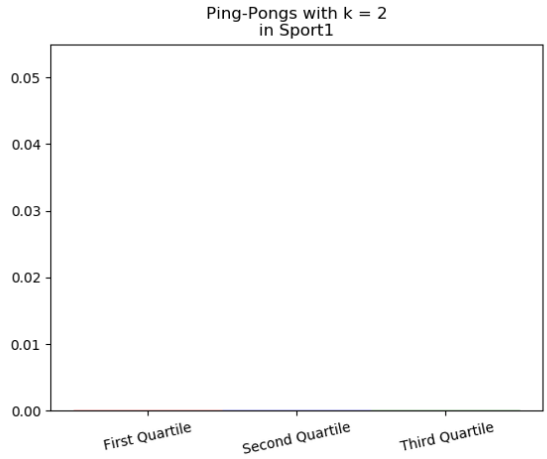

In-Stars with k = 3
in News3
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

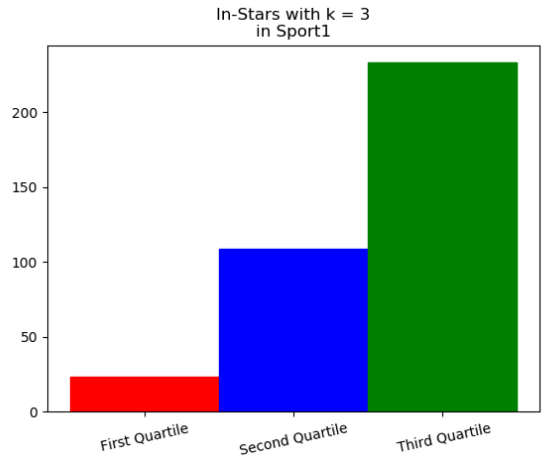

Out-Stars with k = 3
in News3
1600
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

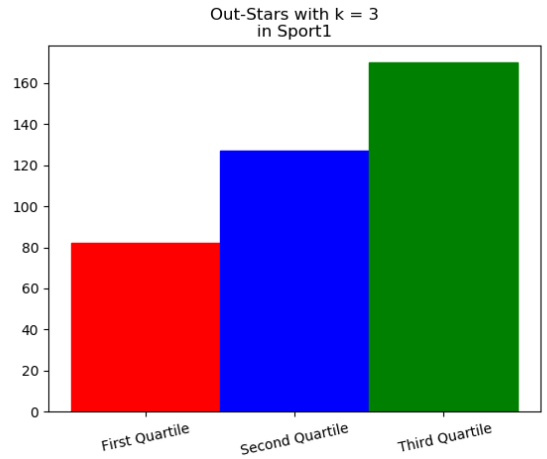

One-Way-Couples with k = 3
in News3
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

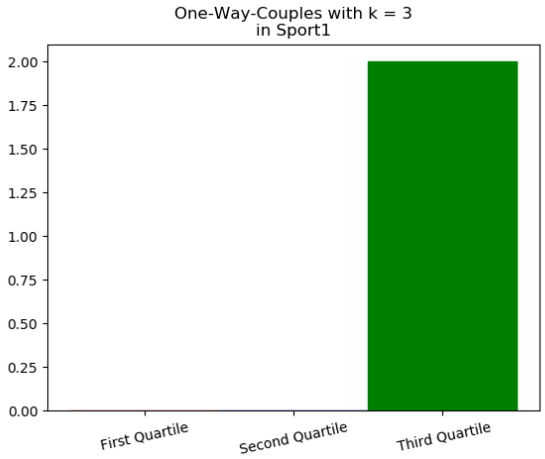

P-Triangles with k = 2
in News3
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

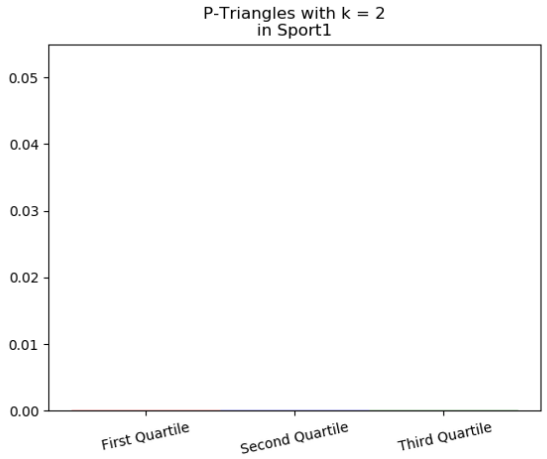

Chains with k = 3
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

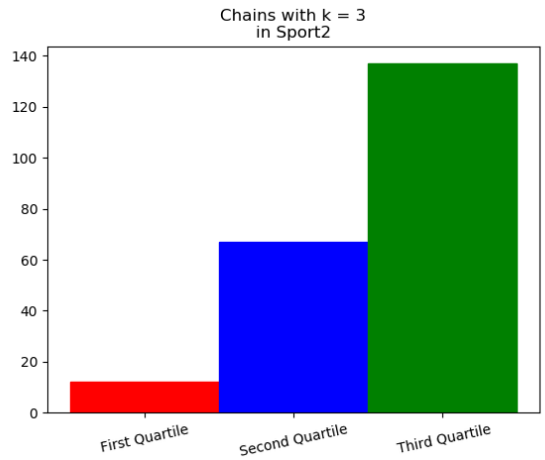

Ping-Pongs with k = 2
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

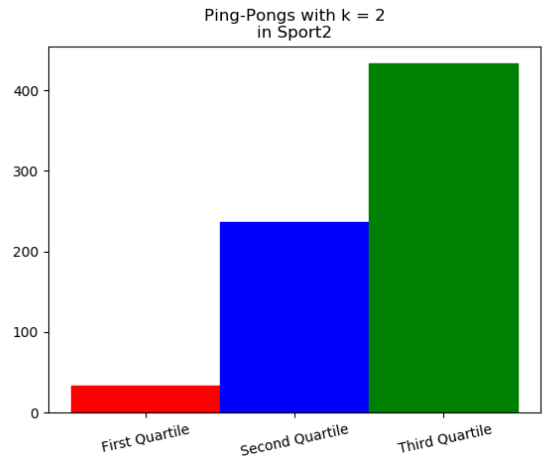

In-Stars with k = 3
in Sport1
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

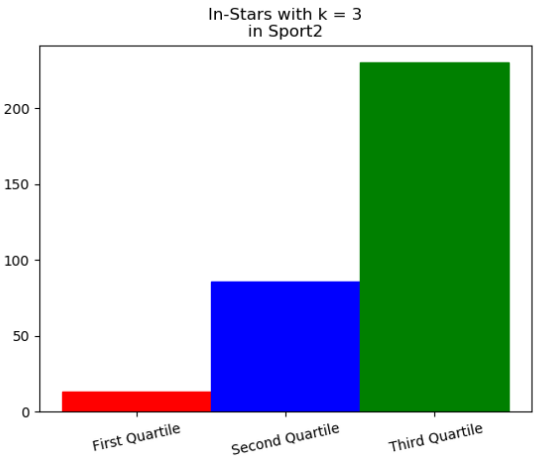

Out-Stars with k = 3
in Sport1
160
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

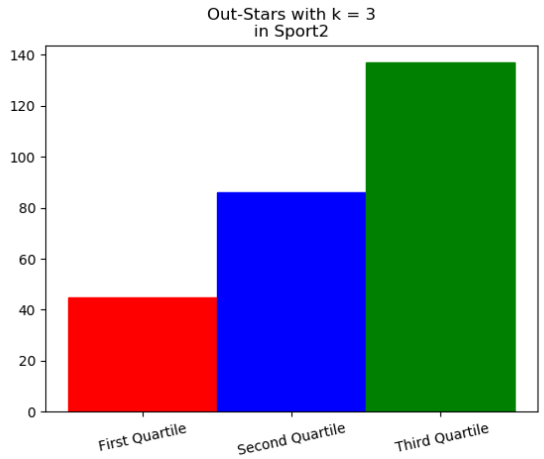

One-Way-Couples with k = 3
in Sport1
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
First Quartile
Second Quartile
Third Quartile

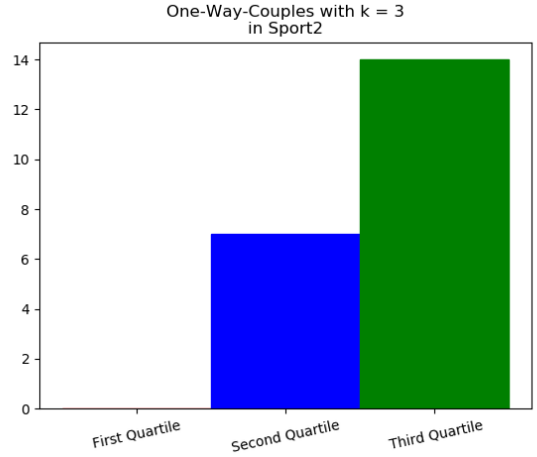

P-Triangles with k = 2
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

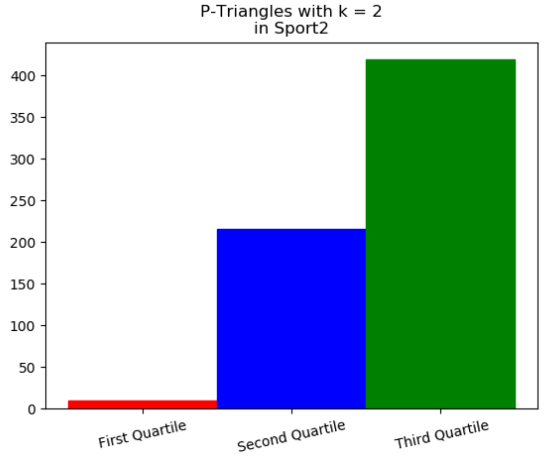

Chains with k = 3
in Sport2
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

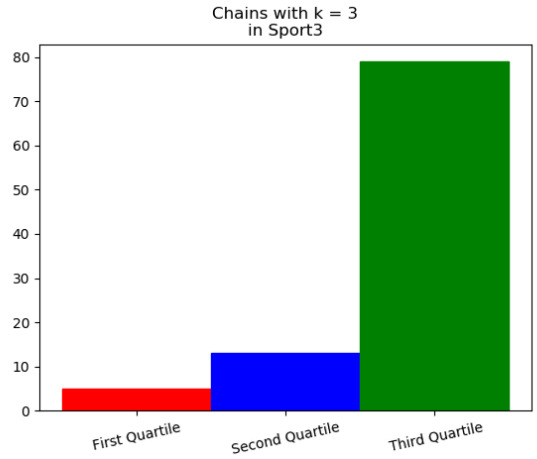

Ping-Pongs with k = 2
in Sport2
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

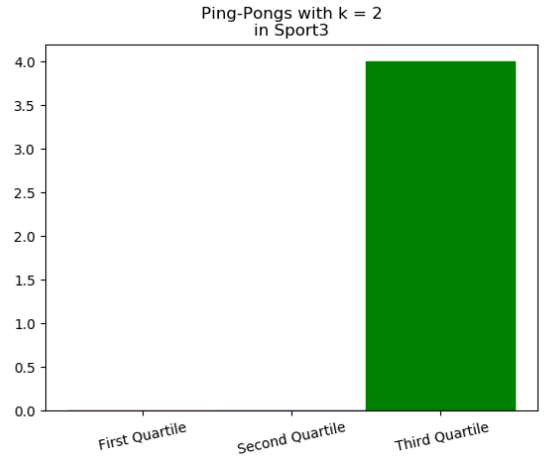

In-Stars with k = 3
in Sport2
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

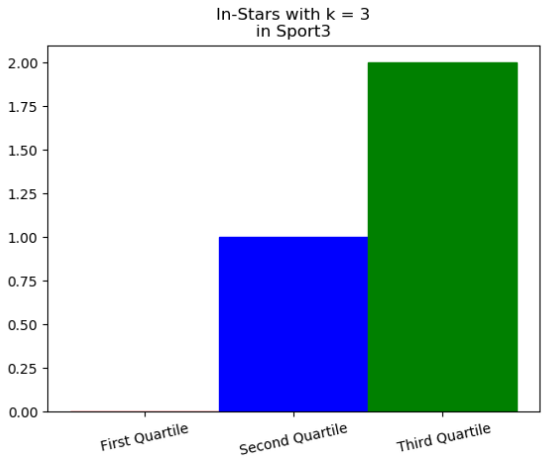

Out-Stars with k = 3
in Sport2
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

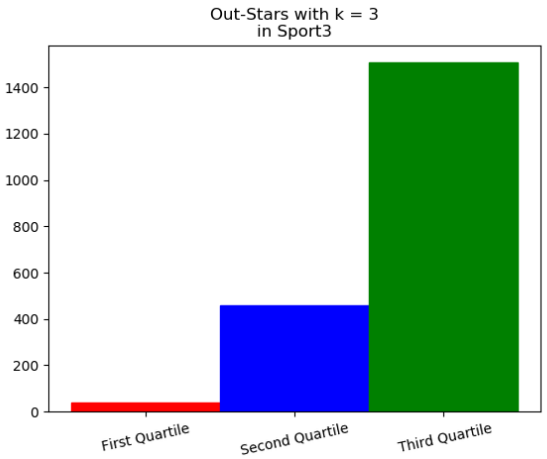

One-Way-Couples with k = 3
in Sport2
14
12
10
8
6
4
2
0
First Quartile
Second Quartile
Third Quartile

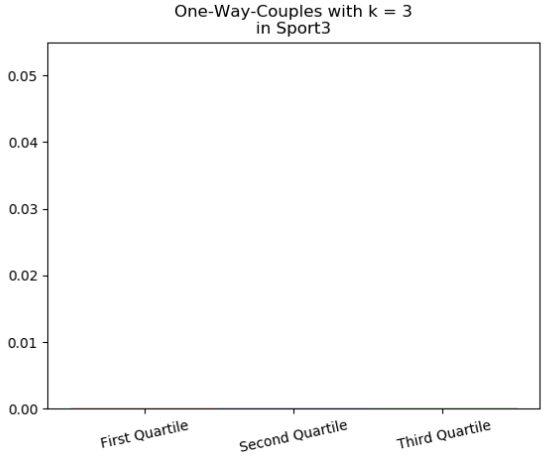

P-Triangles with k = 2
in Sport2
400
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

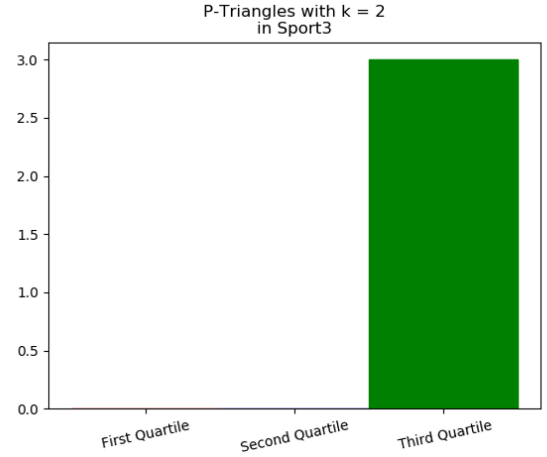

Chains with k = 3
in Sport3
80
70
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

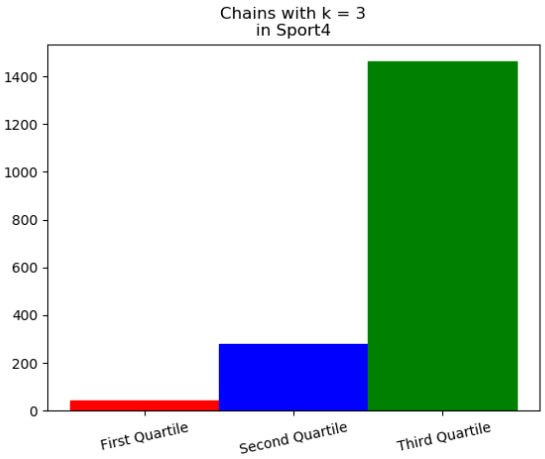

Ping-Pongs with k = 2
in Sport3
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
First Quartile
Second Quartile
Third Quartile

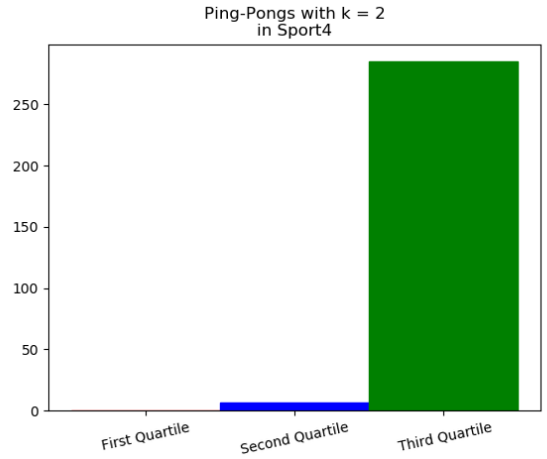

In-Stars with k = 3
in Sport3
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
First Quartile
Second Quartile
Third Quartile

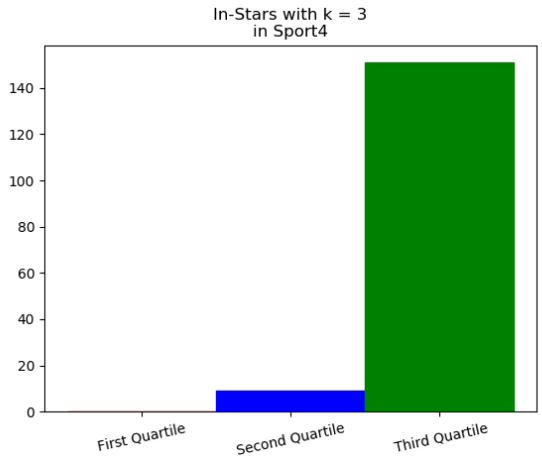

Out-Stars with k = 3
in Sport3
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

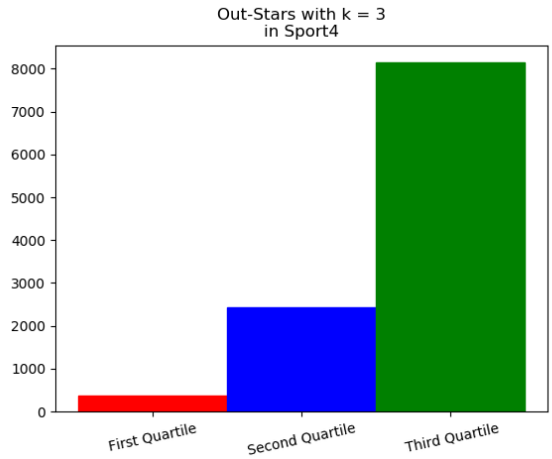

One-Way-Couples with k = 3
in Sport3
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

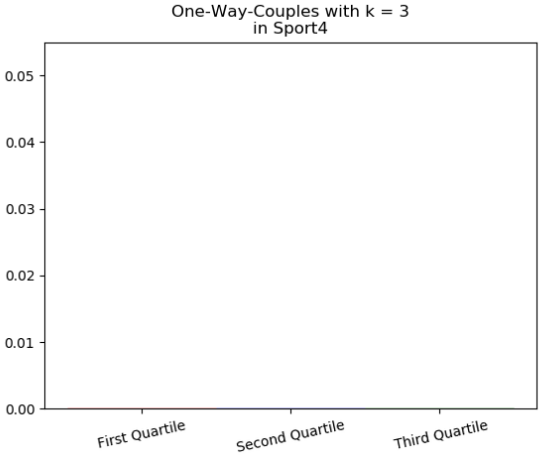

P-Triangles with k = 2
in Sport3
3.0
2.5
2.0
1.5
1.0
0.5
0.0
First Quartile
Second Quartile
Third Quartile

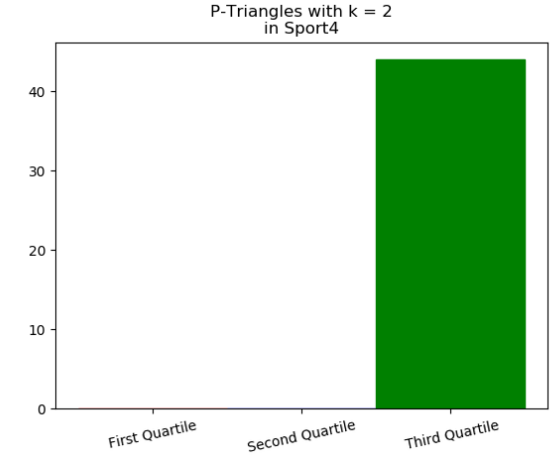

Chains with k = 3
in Sport4
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

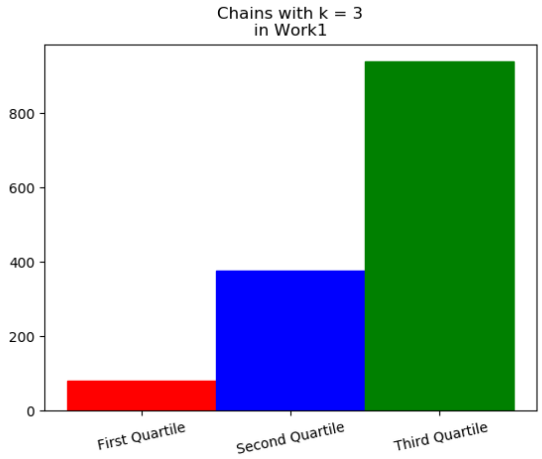

Ping-Pongs with k = 2
in Sport4
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

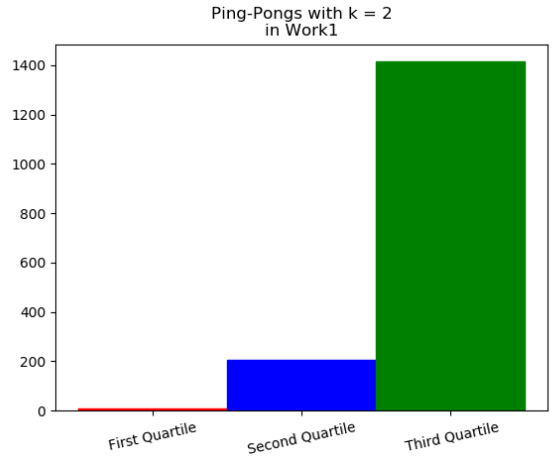

In-Stars with k = 3
in Sport4
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

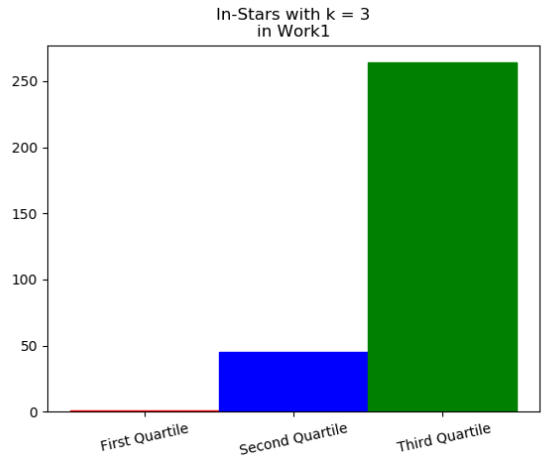

Out-Stars with k = 3
in Sport4
8000
7000
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

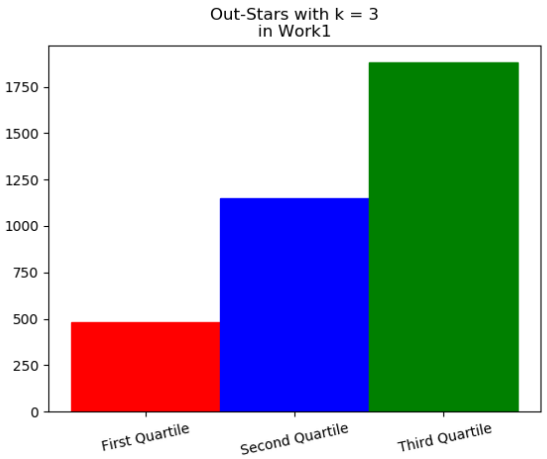

One-Way-Couples with k = 3
in Sport4
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

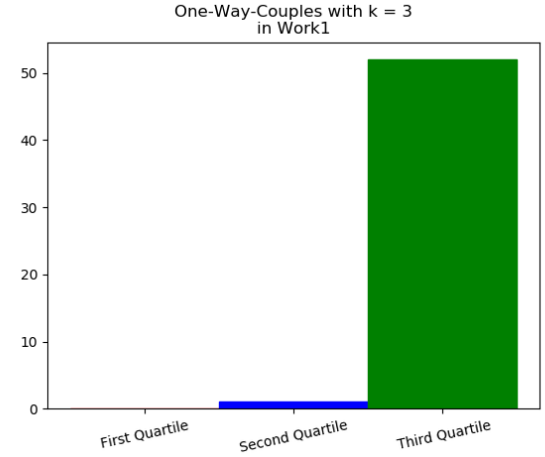

P-Triangles with k = 2
in Sport4
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

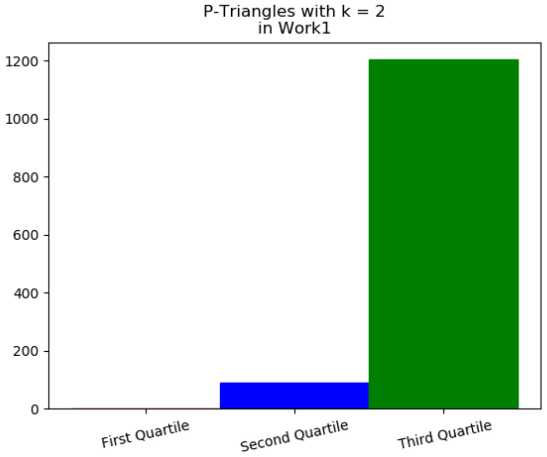

Chains with k = 3
in Work1
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

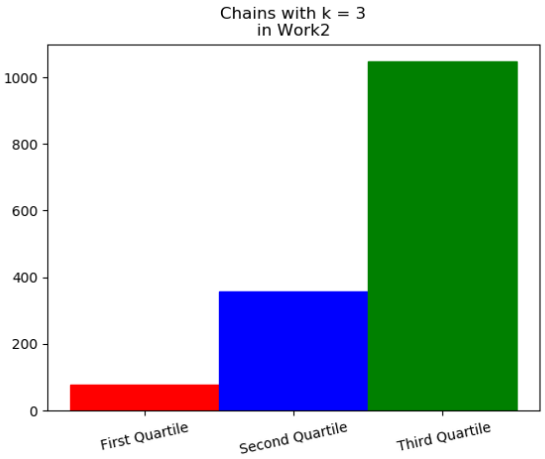

Ping-Pongs with k = 2
in Work1
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

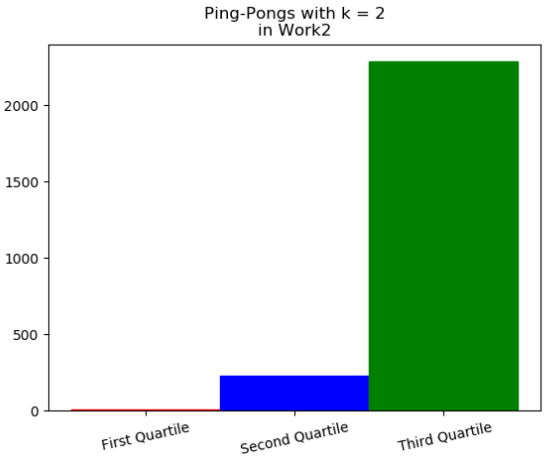

In-Stars with k = 3
in Work1
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

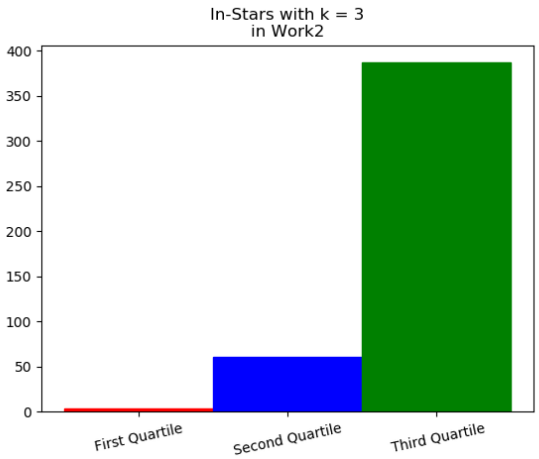

Out-Stars with k = 3
in Work1
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

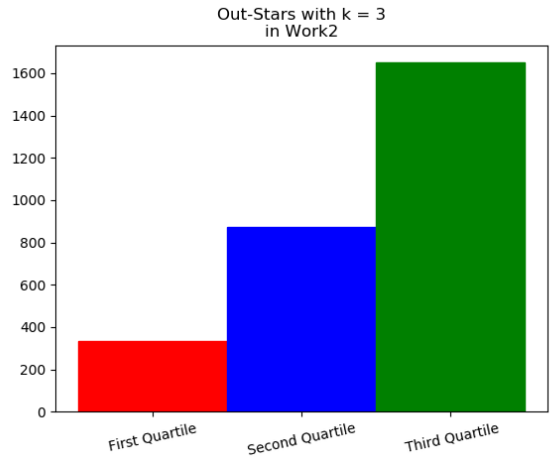

One-Way-Couples with k = 3
in Work1
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

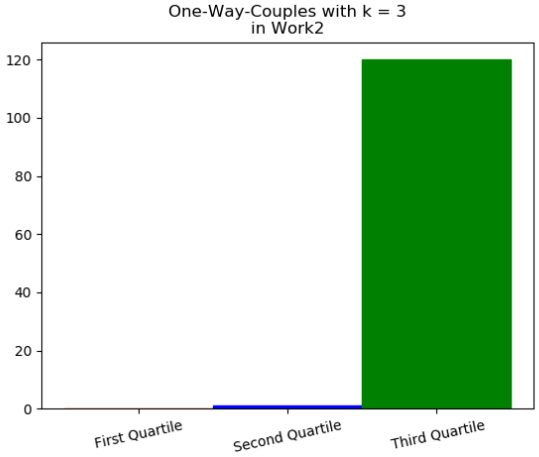

P-Triangles with k = 2
in Work1
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

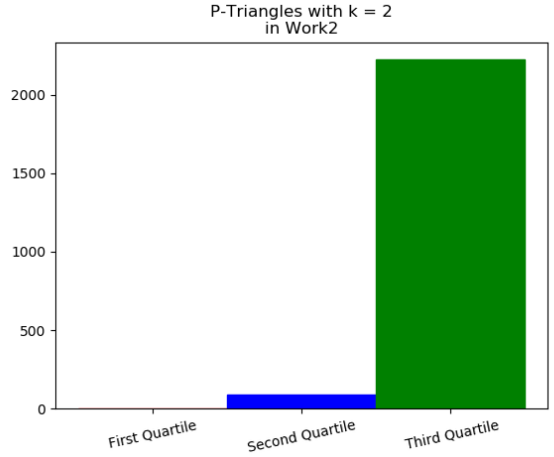

Chains with k = 3
in Work2
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

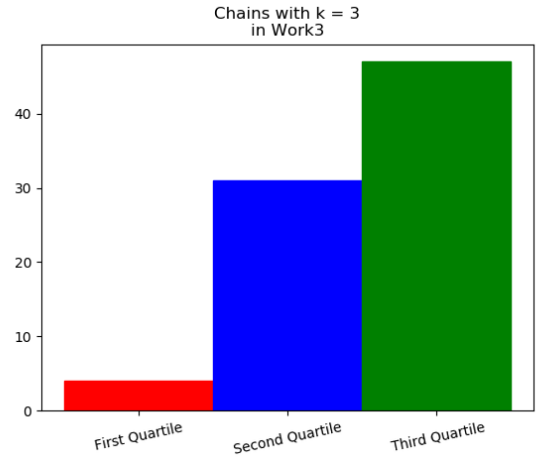

Ping-Pongs with k = 2
in Work2
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

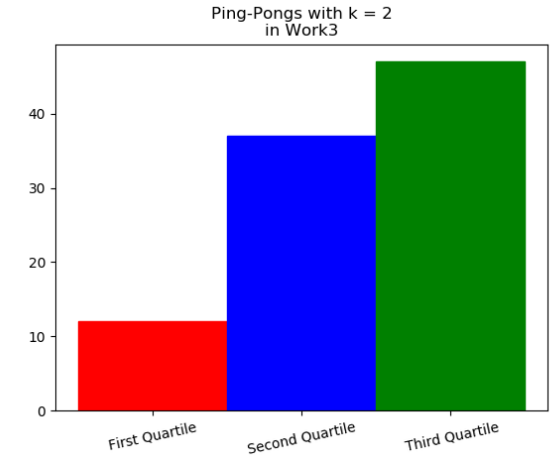

In-Stars with k = 3
in Work2
400
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

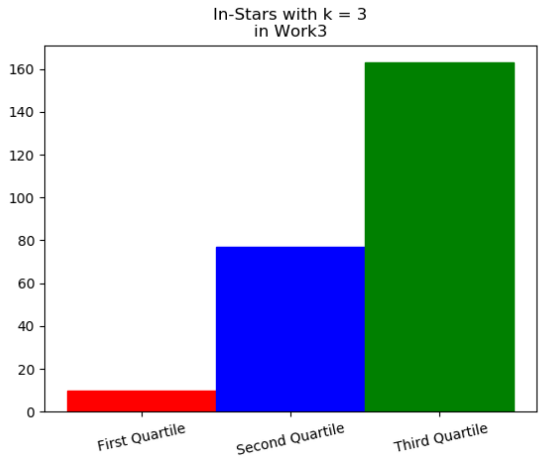

Out-Stars with k = 3
in Work2
1600
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

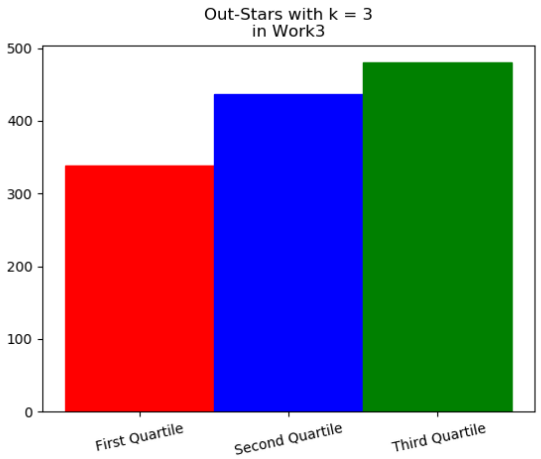

One-Way-Couples with k = 3
in Work2
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

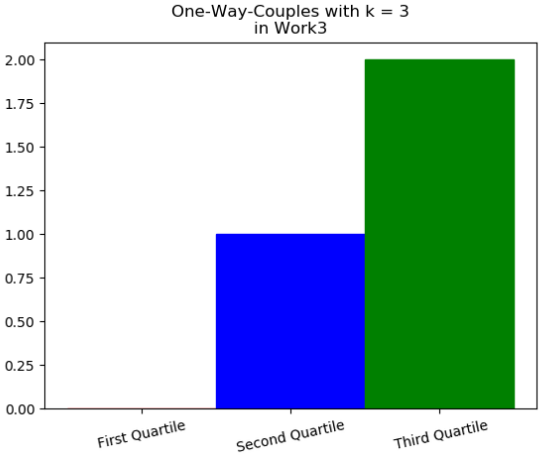

P-Triangles with k = 2
in Work2
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

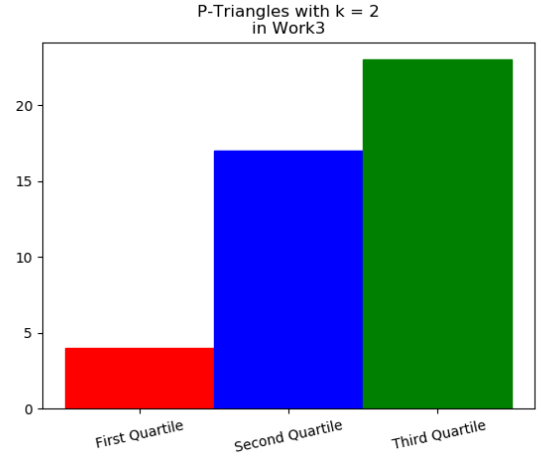

Chains with k = 3
in Work3
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

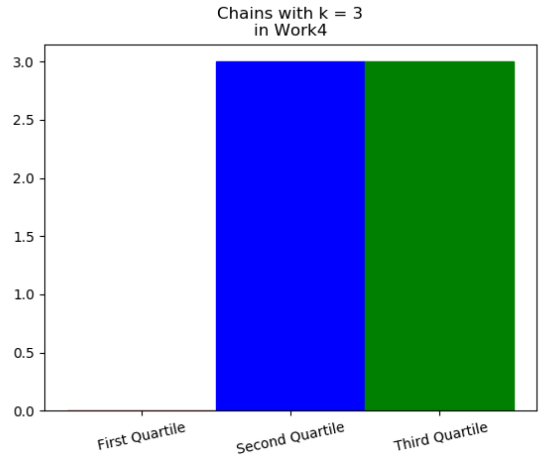

Ping-Pongs with k = 2
in Work3
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

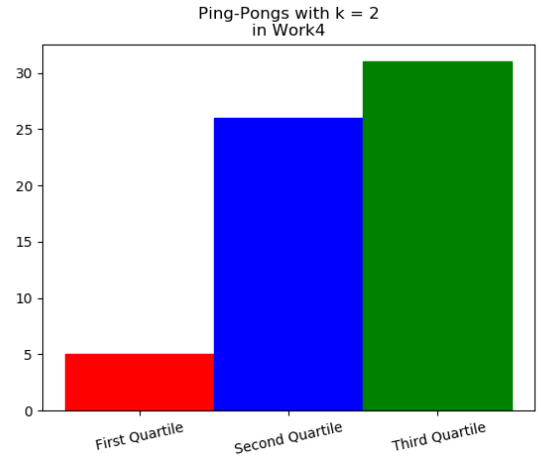

In-Stars with k = 3
in Work3
160
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile


Out-Stars with k = 3
in Work3
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile


One-Way-Couples with k = 3
in Work3
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
First Quartile
Second Quartile
Third Quartile


P-Triangles with k = 2
in Work3
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile


Chains with k = 3
in Work4
3.0
2.5
2.0
1.5
1.0
0.5
0.0
First Quartile
Second Quartile
Third Quartile


Ping-Pongs with k = 2
in Work4
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

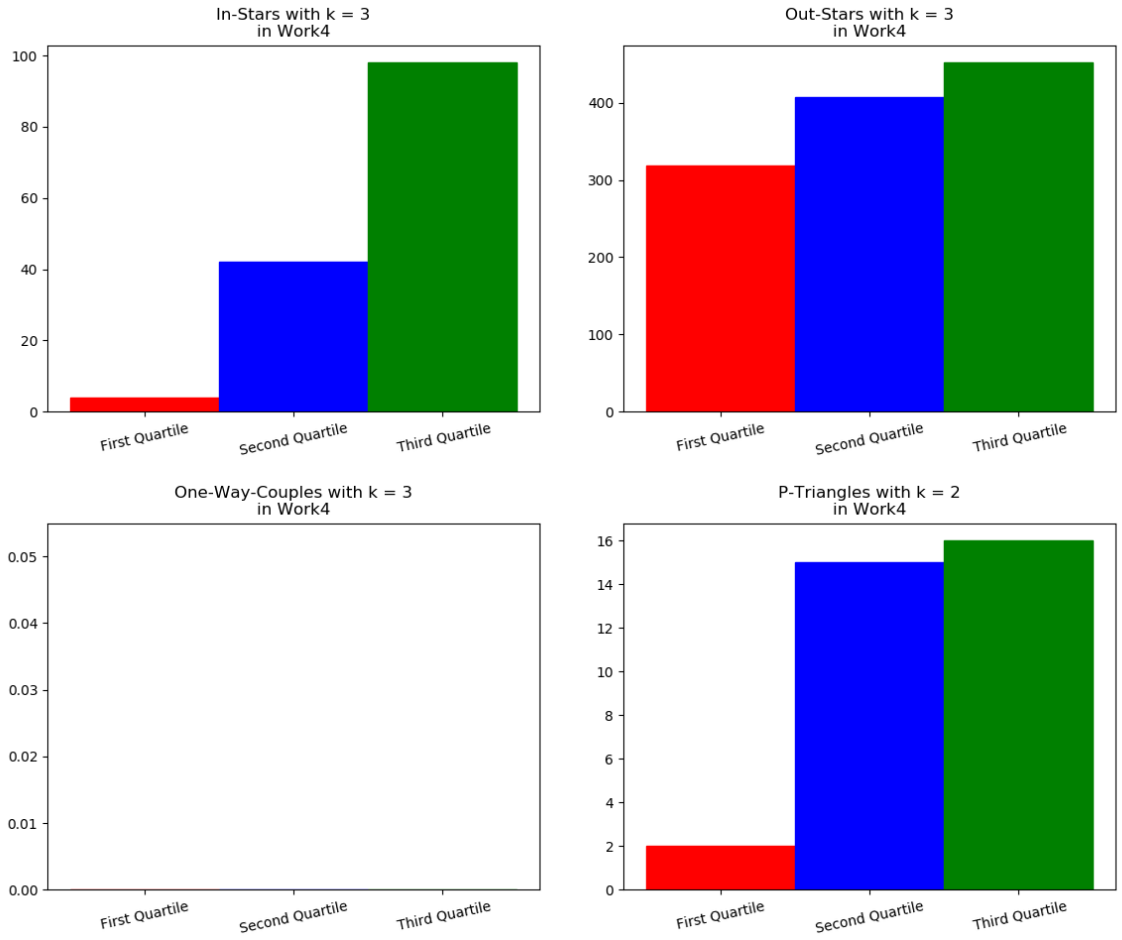
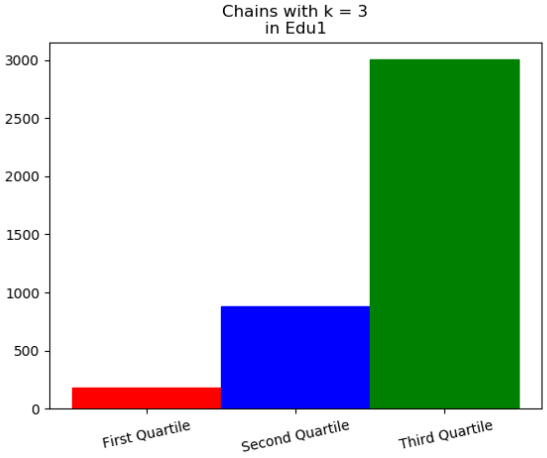
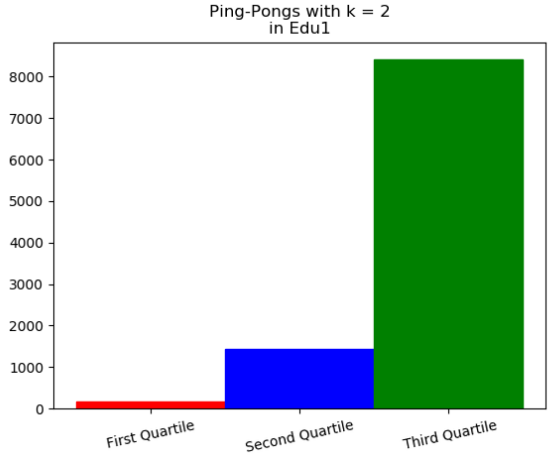
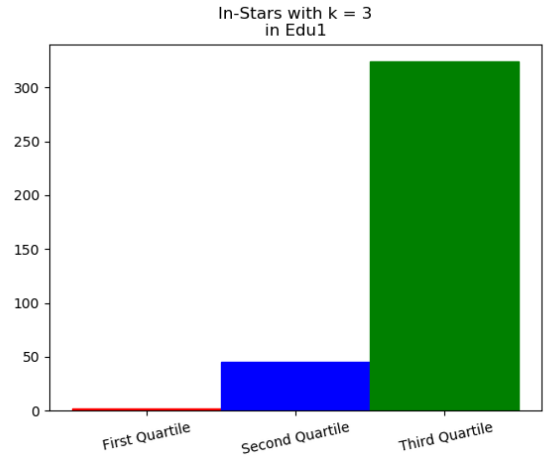


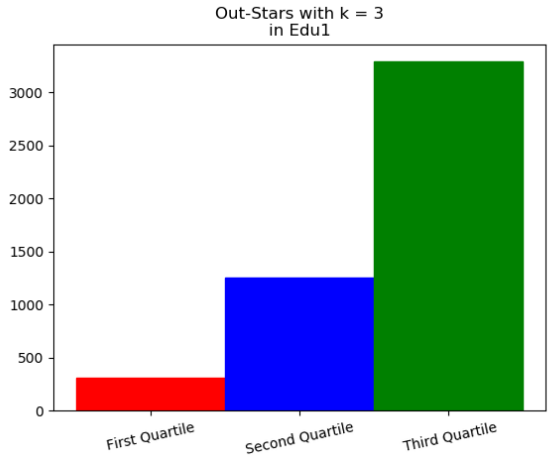
Figure A.1: The plots of the results obtained for all groups modeled with the Post Graph with the time window sliding approach

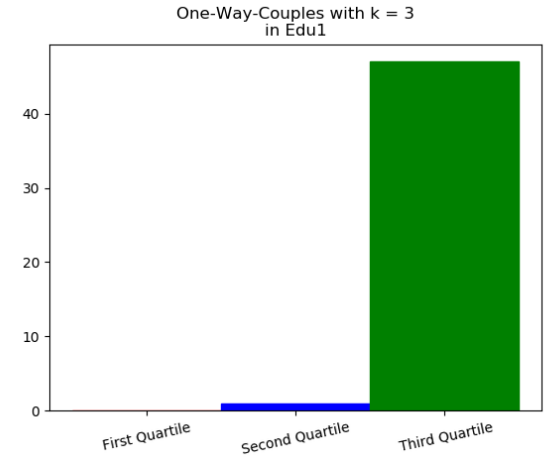
In **Figure A.2** we show the group-based diagrams elaborated on the results of the number of motifs found in the Post Graph with the snapshots approach.

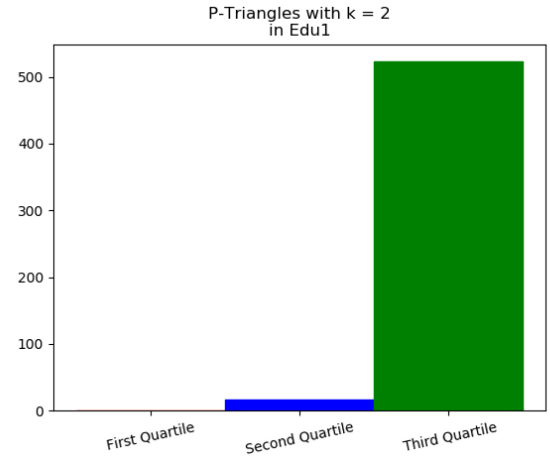

Chains with k = 3
in Edu1
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

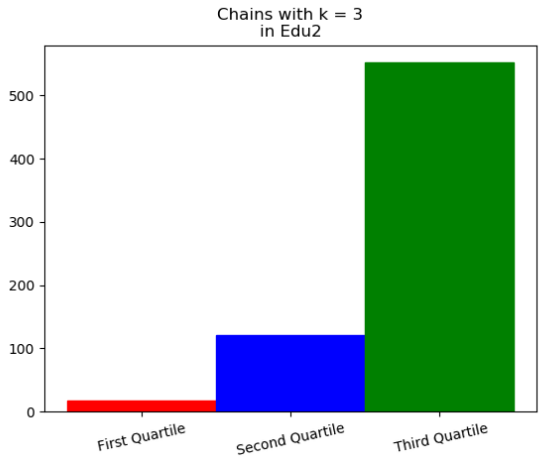

Ping-Pongs with k = 2
in Edu1
8000
7000
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

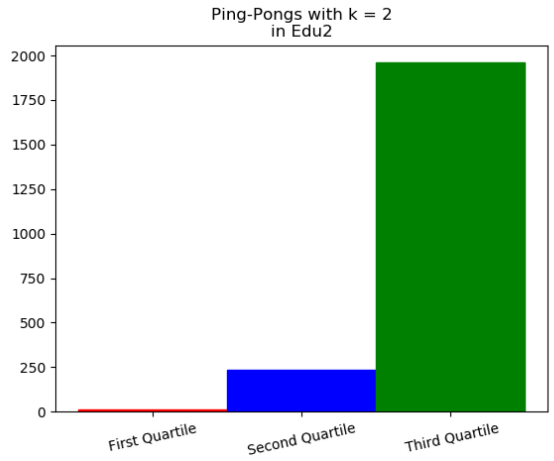

In-Stars with k = 3
in Edu1
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

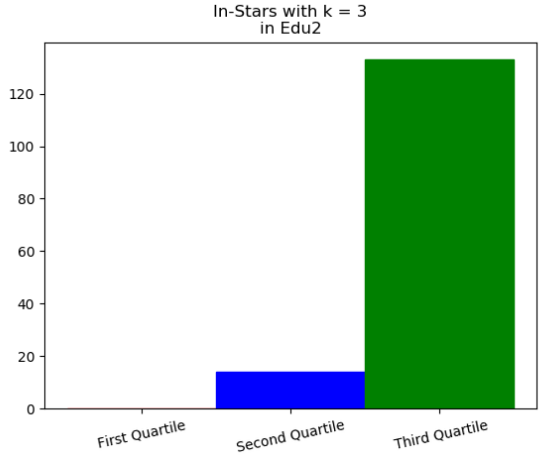

Out-Stars with k = 3
in Edu1
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

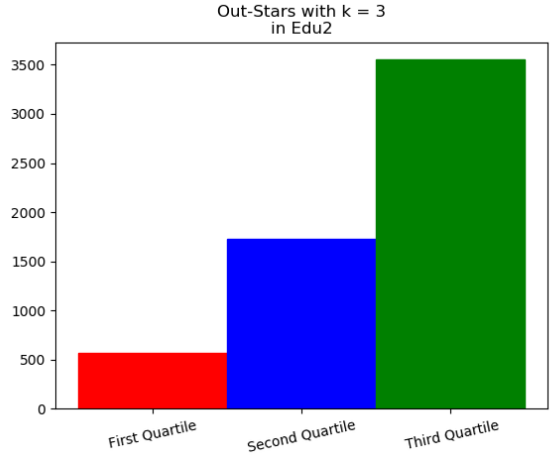

One-Way-Couples with k = 3
in Edu1
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

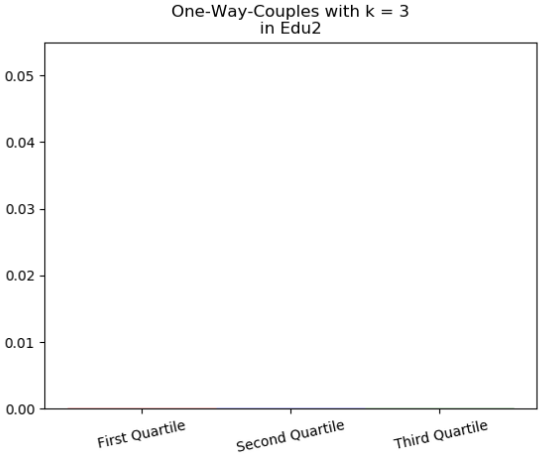

P-Triangles with k = 2
in Edu1
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

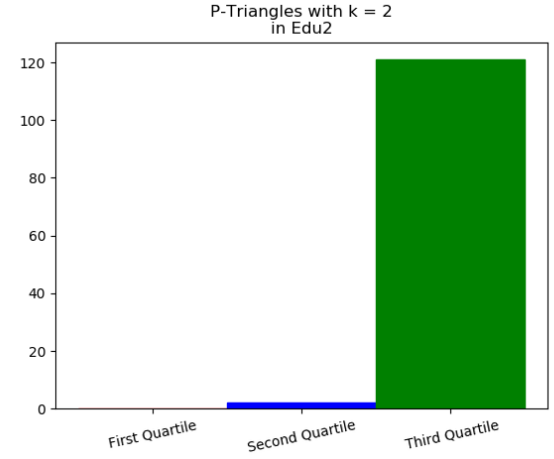

Chains with k = 3
in Edu2
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

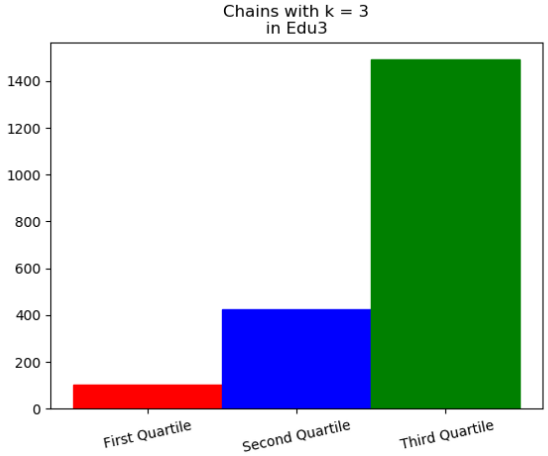

Ping-Pongs with k = 2
in Edu2
2000
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

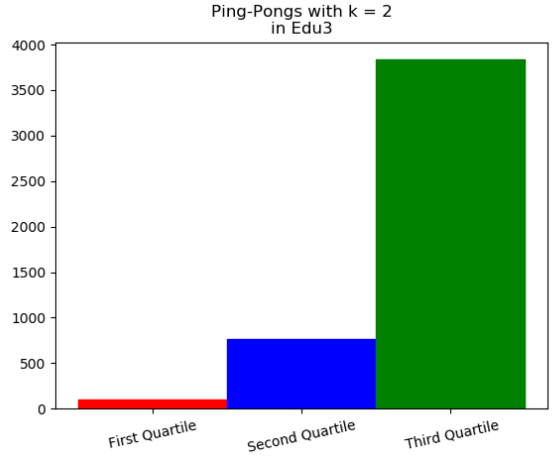

In-Stars with k = 3
in Edu2
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

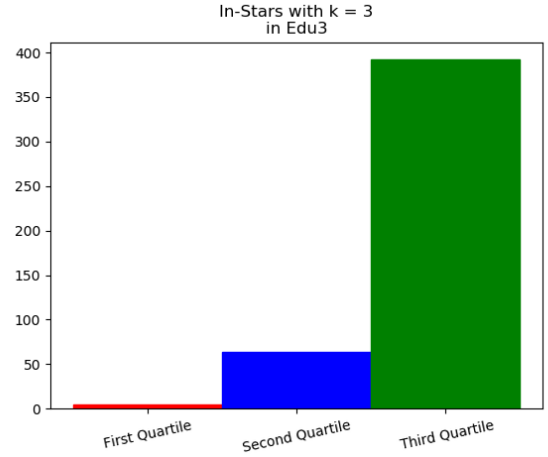

Out-Stars with k = 3
in Edu2
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

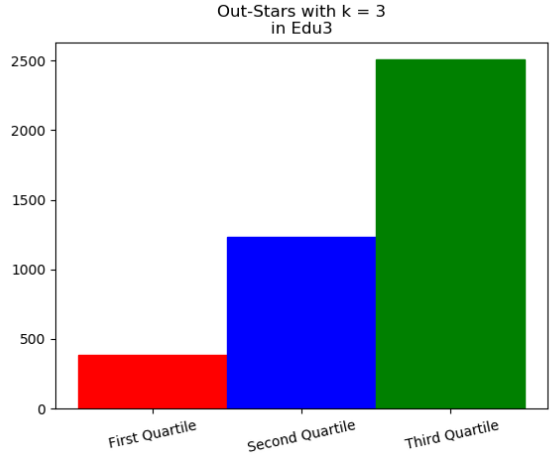

One-Way-Couples with k = 3
in Edu2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

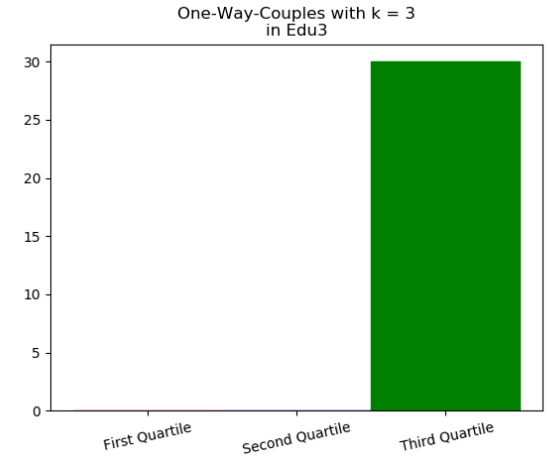

P-Triangles with k = 2
in Edu2
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

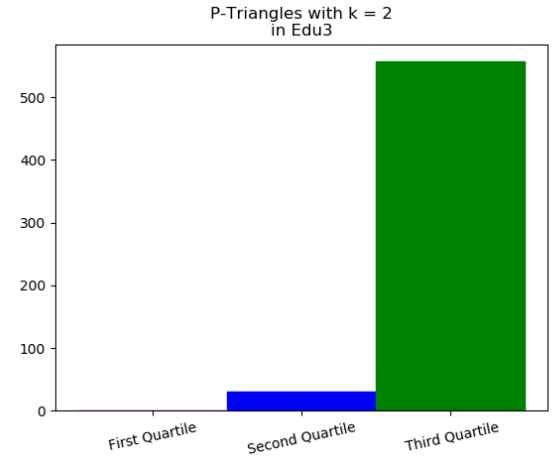

Chains with k = 3
in Edu3
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

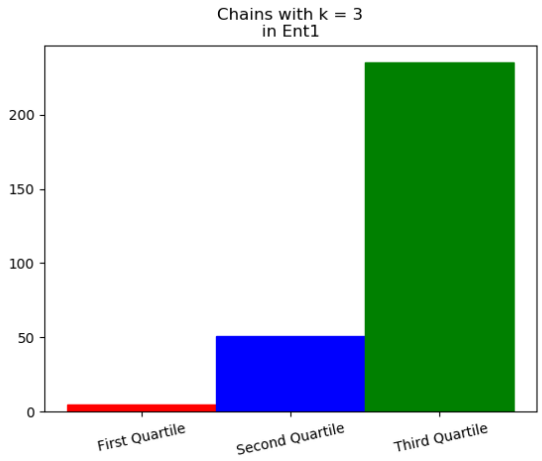

Ping-Pongs with k = 2
in Edu3
4000
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

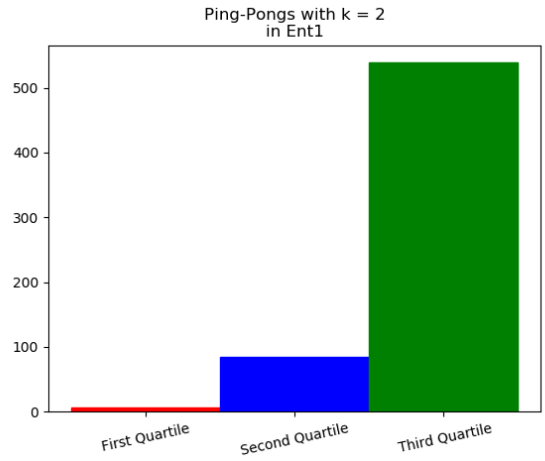

In-Stars with k = 3
in Edu3
400
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

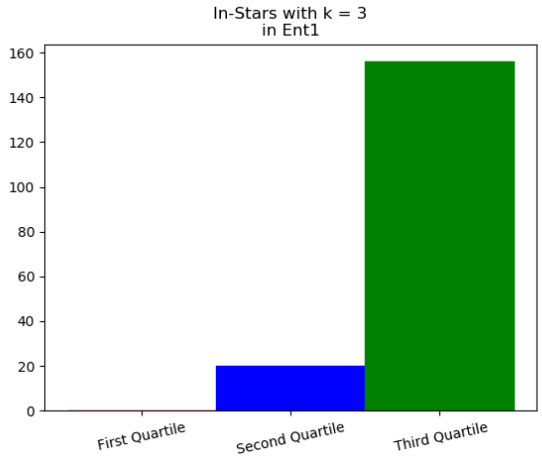

Out-Stars with k = 3
in Edu3
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

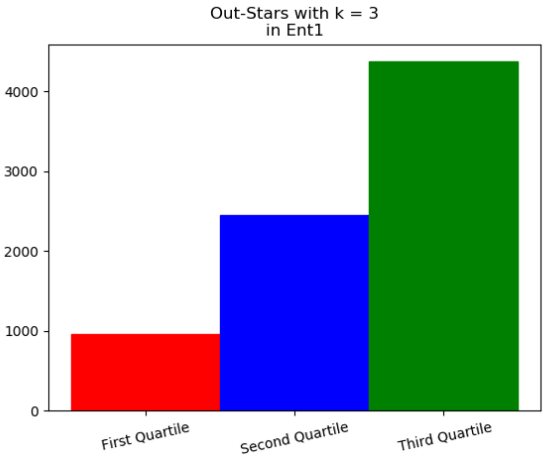

One-Way-Couples with k = 3
in Edu3
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

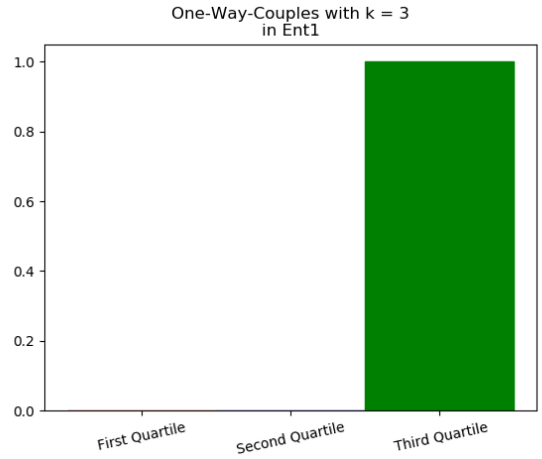

P-Triangles with k = 2
in Edu3
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

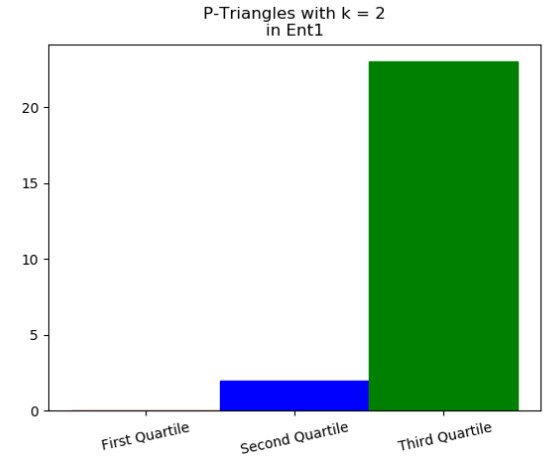

Chains with k = 3
in Ent1
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

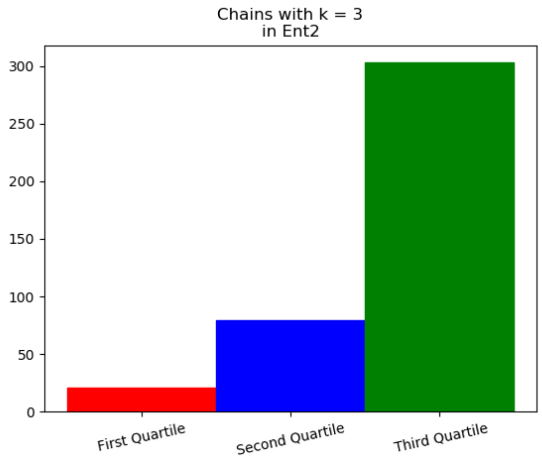

Ping-Pongs with k = 2
in Ent1
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

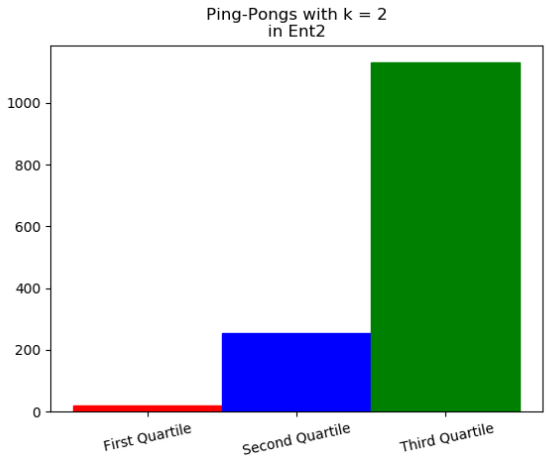

In-Stars with k = 3
in Ent1
160
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

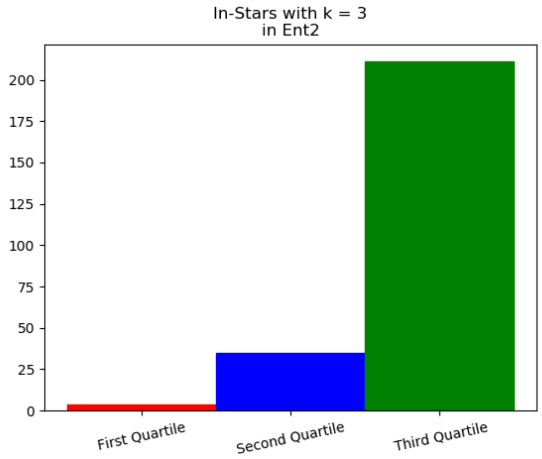

Out-Stars with k = 3
in Ent1
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

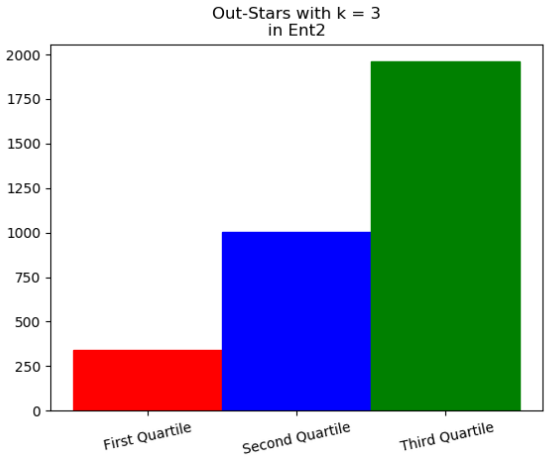

One-Way-Couples with k = 3
in Ent1
1.0
0.8
0.6
0.4
0.2
0.0
First Quartile
Second Quartile
Third Quartile

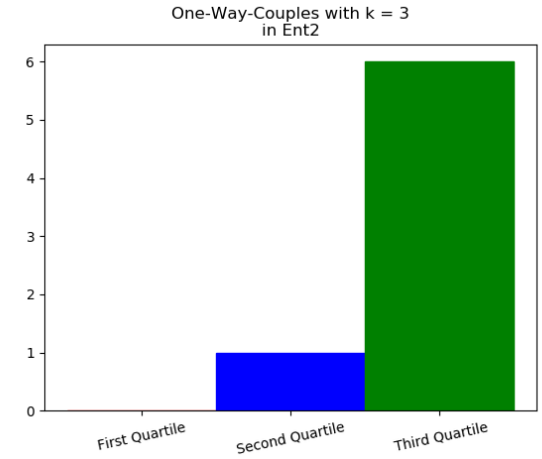

P-Triangles with k = 2
in Ent1
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile


Chains with k = 3
in Ent2
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

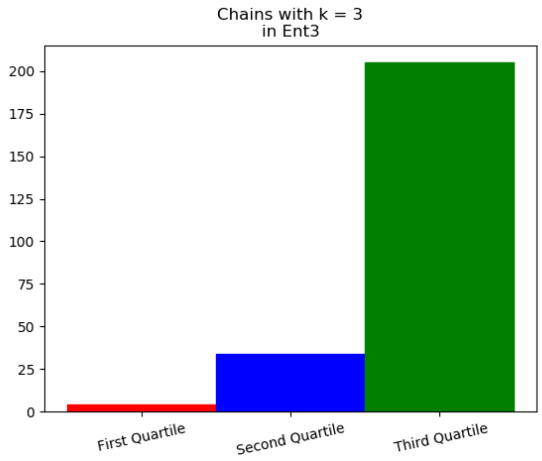

Ping-Pongs with k = 2
in Ent2
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

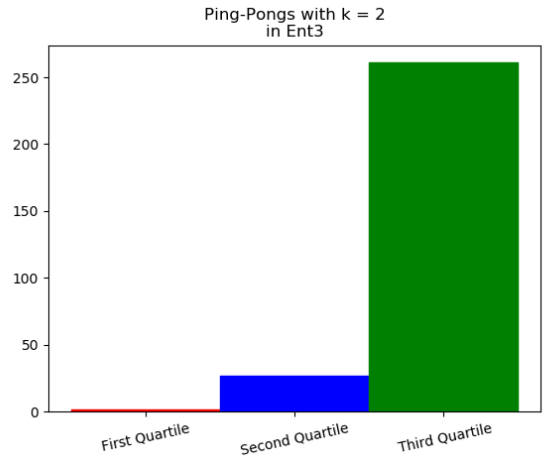

In-Stars with k = 3
in Ent2
200
175
150
125
100
75
50
25
0
First Quartile
Second Quartile
Third Quartile

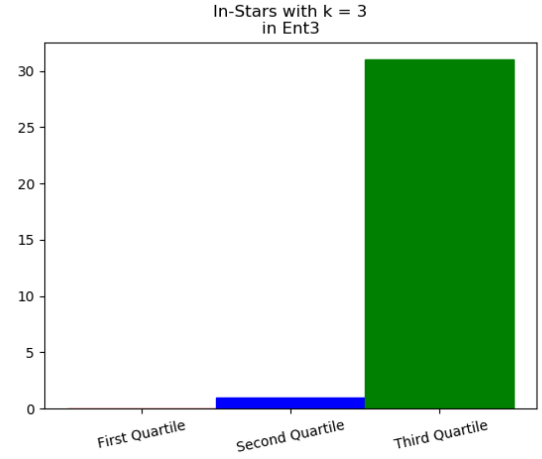

Out-Stars with k = 3
in Ent2
2000
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

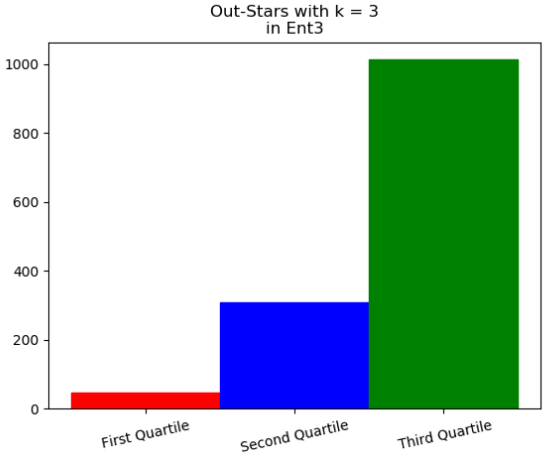

One-Way-Couples with k = 3
in Ent2
6
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

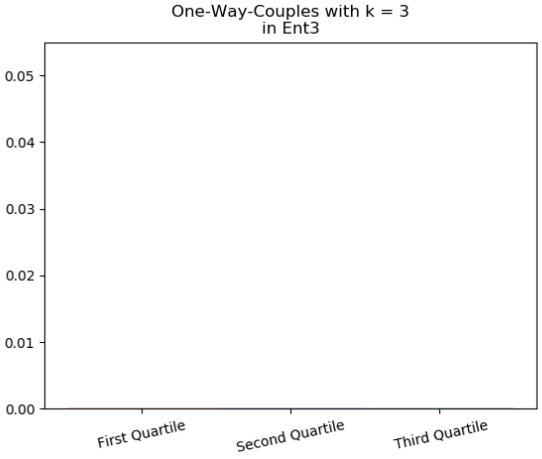

P-Triangles with k = 2
in Ent2
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

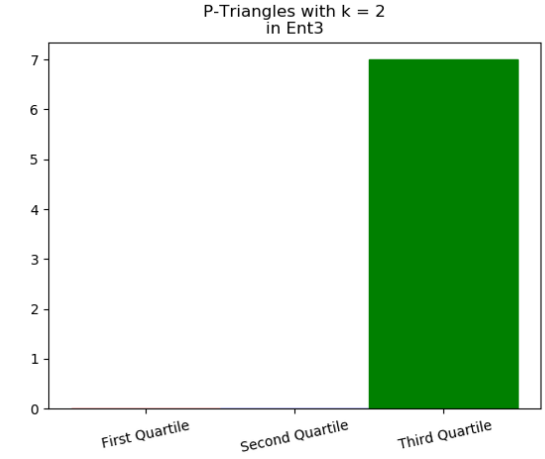

Chains with k = 3
in Ent3
200
175
150
125
100
75
50
25
0
First Quartile
Second Quartile
Third Quartile

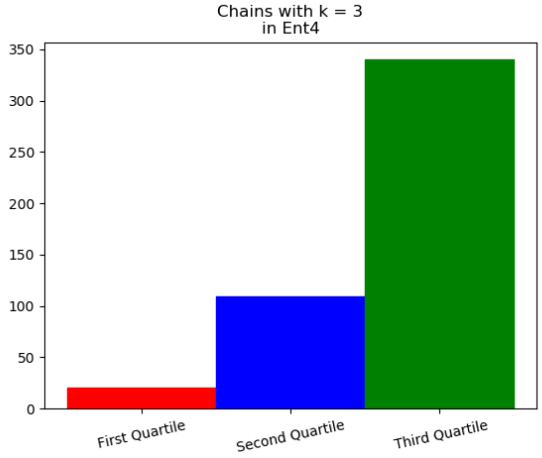

Ping-Pongs with k = 2
in Ent3
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

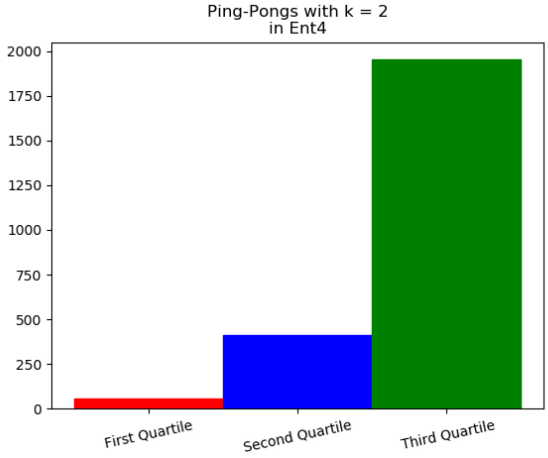

In-Stars with k = 3
in Ent3
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

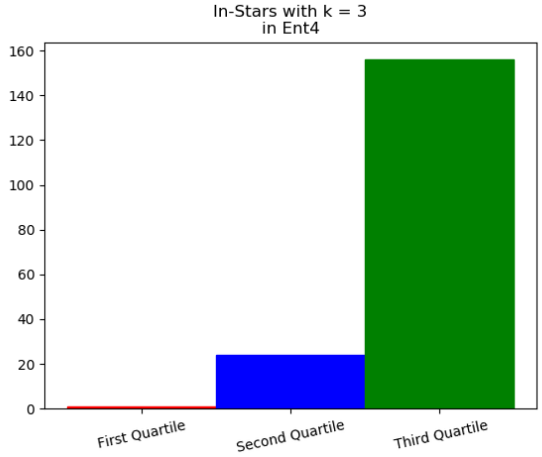

Out-Stars with k = 3
in Ent3
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

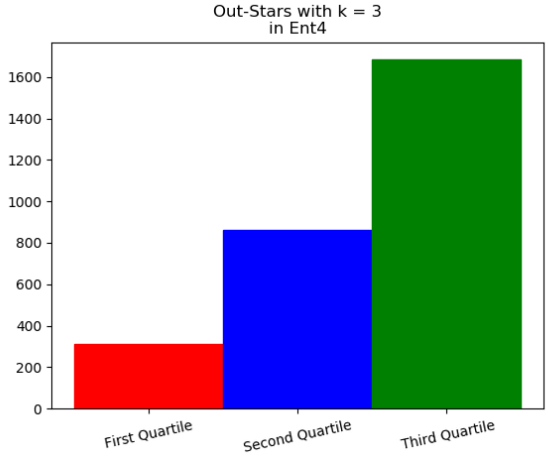

One-Way-Couples with k = 3
in Ent3
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

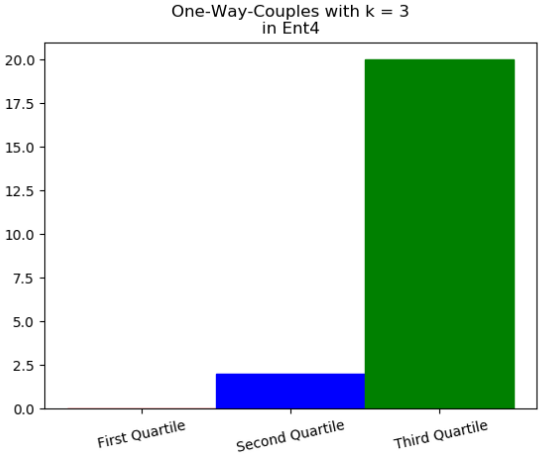

P-Triangles with k = 2
in Ent3
7
6
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

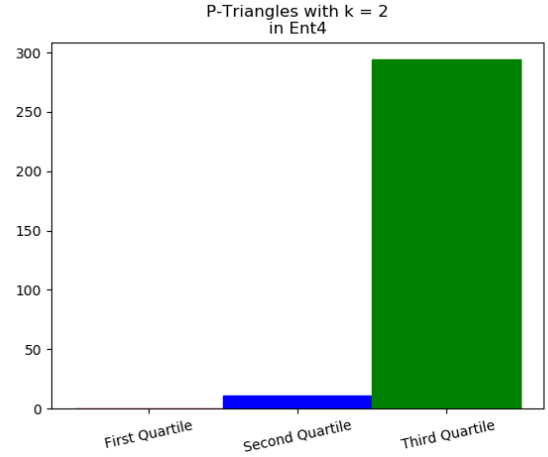

Chains with k = 3
in Ent4
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

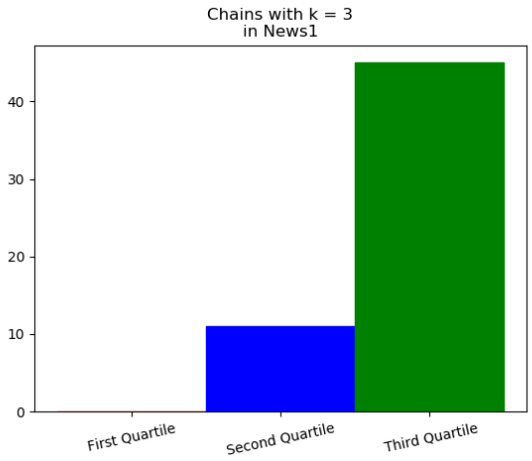

Ping-Pongs with k = 2
in Ent4
2000
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

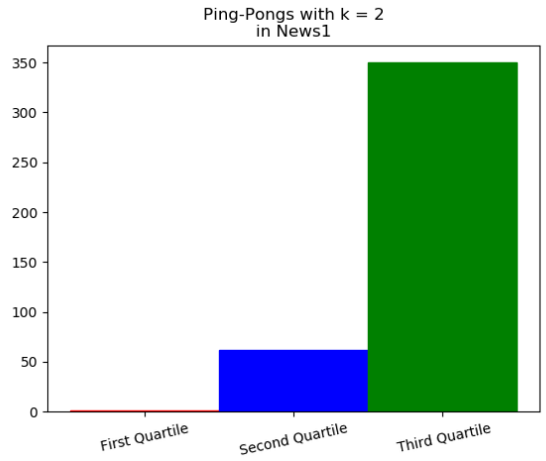

In-Stars with k = 3
in Ent4
160
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

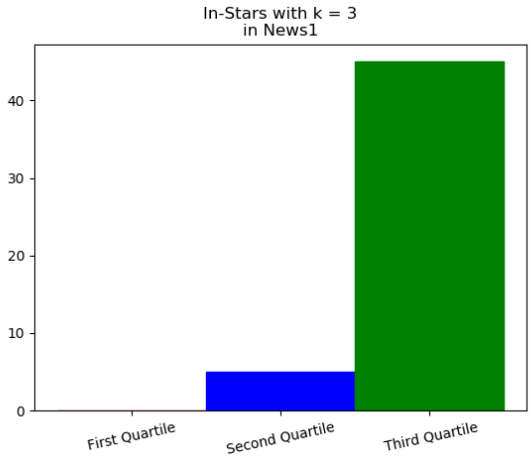

Out-Stars with k = 3
in Ent4
1600
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

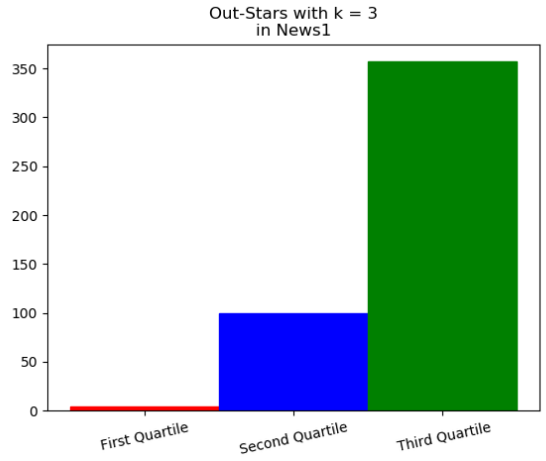

One-Way-Couples with k = 3
in Ent4
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
First Quartile
Second Quartile
Third Quartile

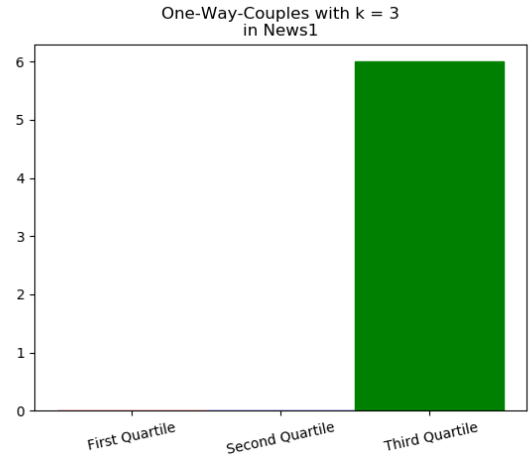

P-Triangles with k = 2
in Ent4
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

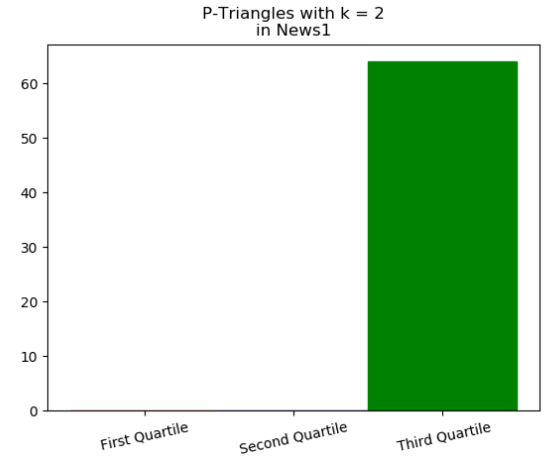

Chains with k = 3
in News1
0
10
20
30
40
First Quartile
Second Quartile
Third Quartile

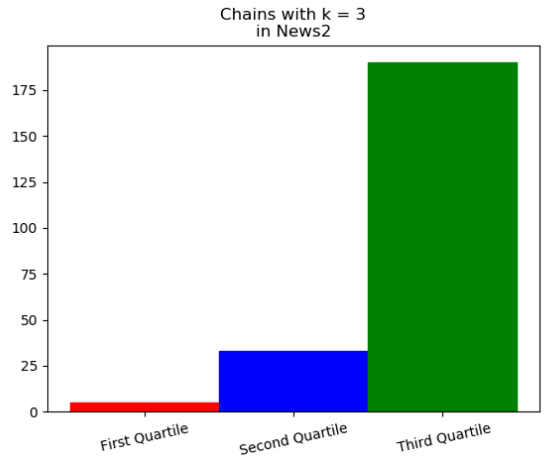

Ping-Pongs with k = 2
in News1
0
50
100
150
200
250
300
350
First Quartile
Second Quartile
Third Quartile

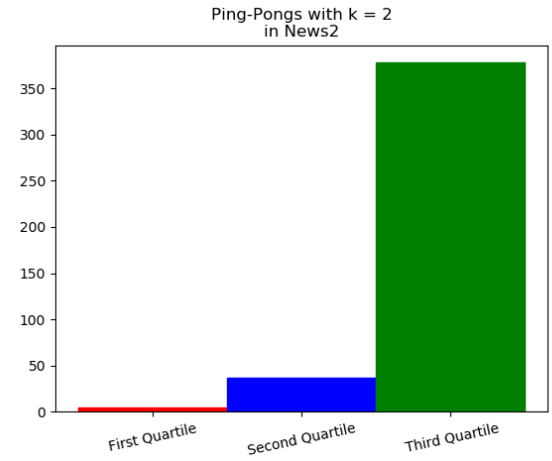

In-Stars with k = 3
in News1
0
10
20
30
40
First Quartile
Second Quartile
Third Quartile

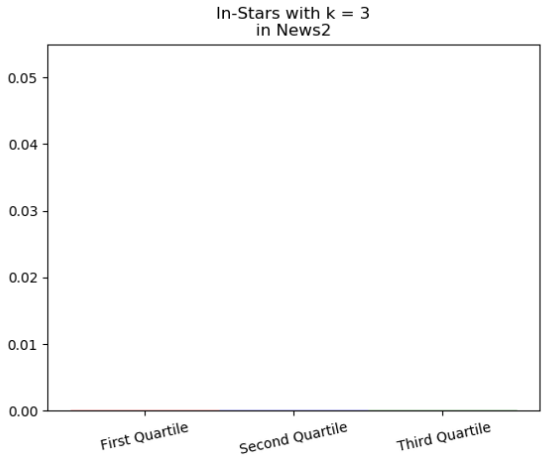

Out-Stars with k = 3
in News1
0
50
100
150
200
250
300
350
First Quartile
Second Quartile
Third Quartile

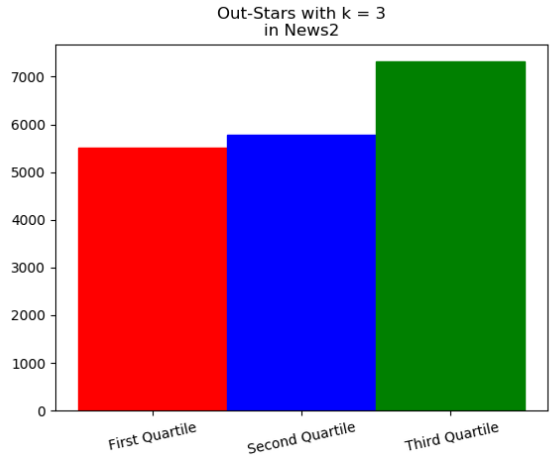

One-Way-Couples with k = 3
in News1
0
1
2
3
4
5
6
First Quartile
Second Quartile
Third Quartile

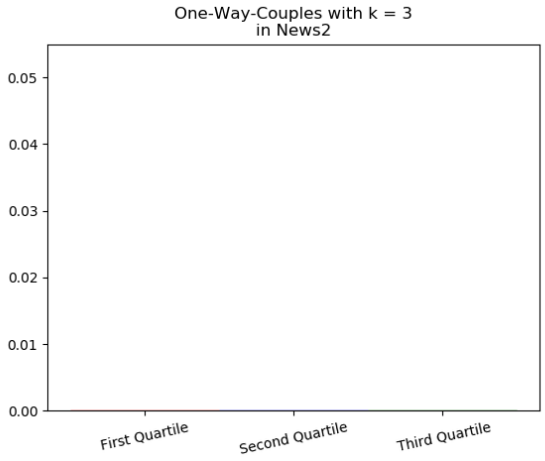

P-Triangles with k = 2
in News1
0
10
20
30
40
50
60
First Quartile
Second Quartile
Third Quartile

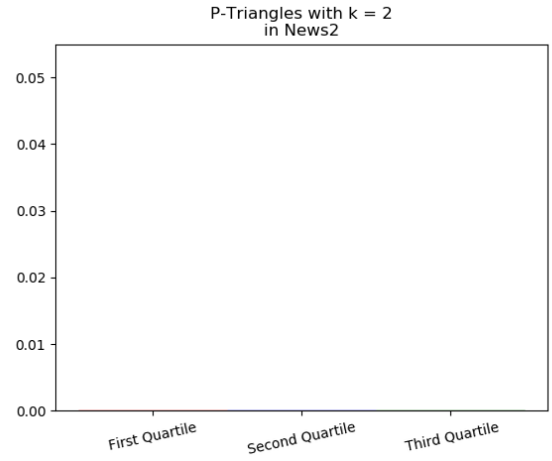

Chains with k = 3
in News2
175
150
125
100
75
50
25
0
First Quartile
Second Quartile
Third Quartile

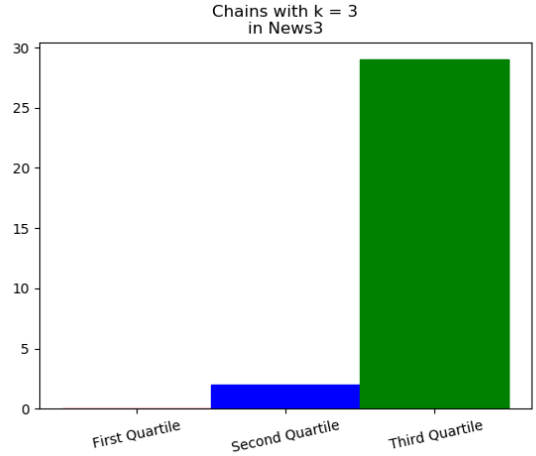

Ping-Pongs with k = 2
in News2
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

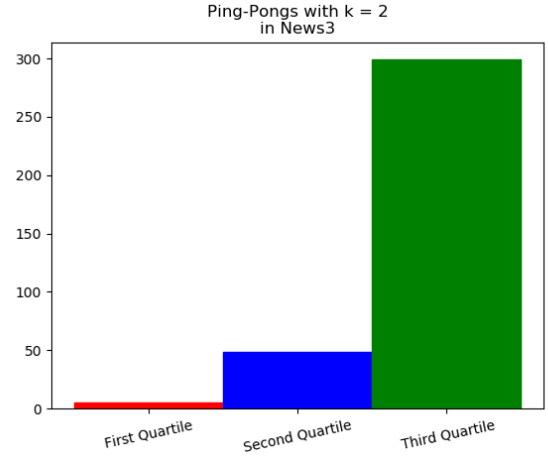

In-Stars with k = 3
in News2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

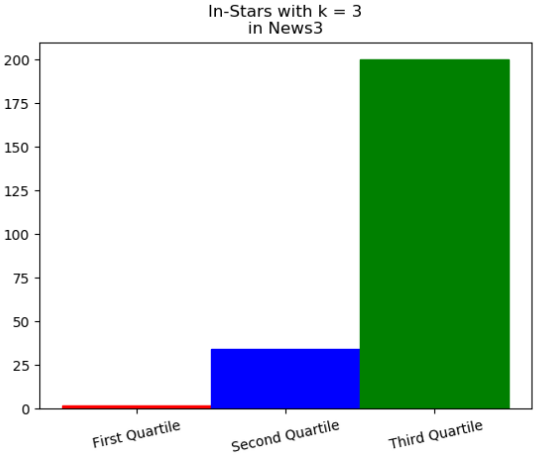

Out-Stars with k = 3
in News2
7000
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

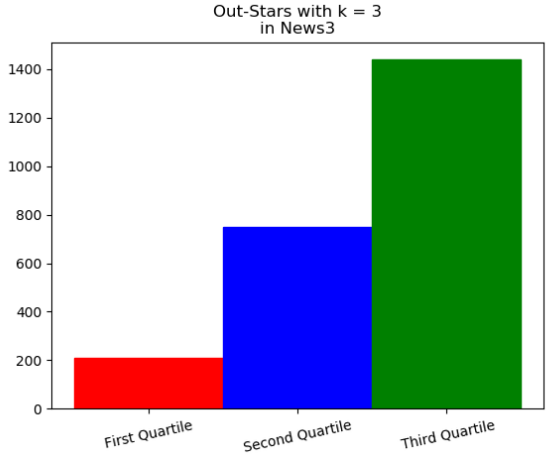

One-Way-Couples with k = 3
in News2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

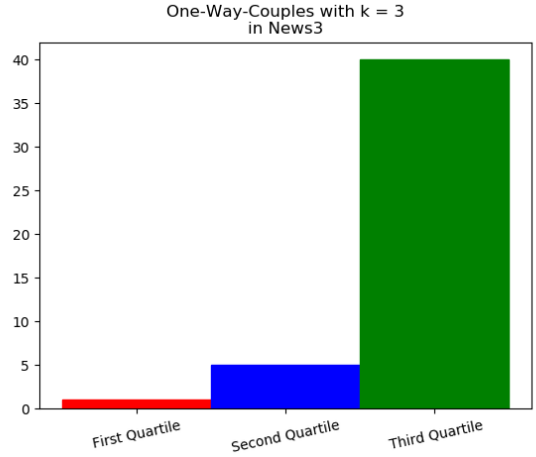

P-Triangles with k = 2
in News2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

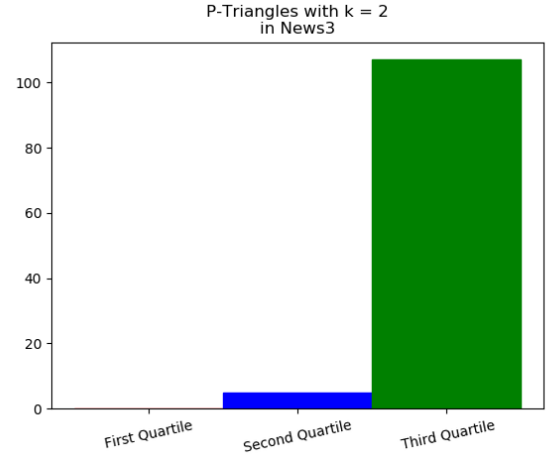

Chains with k = 3
in News3
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

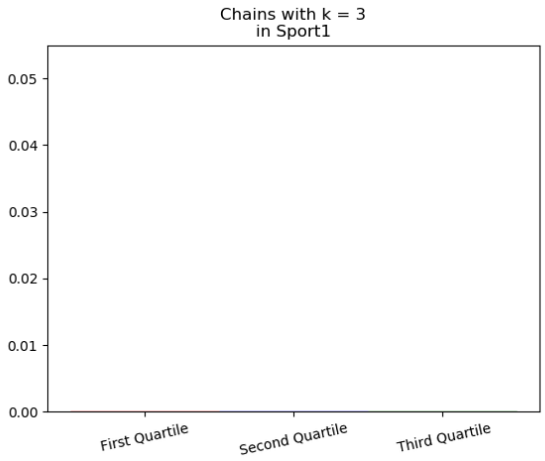

Ping-Pongs with k = 2
in News3
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

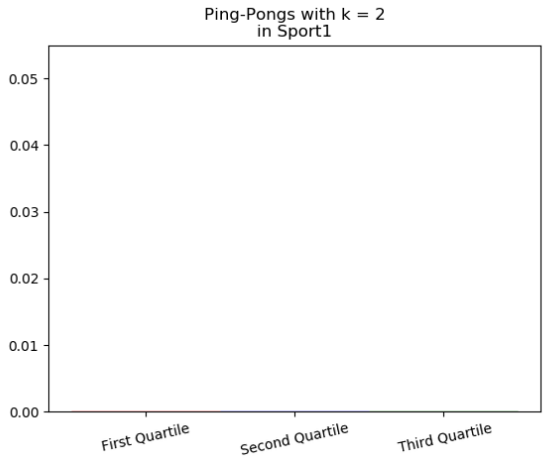

In-Stars with k = 3
in News3
200
175
150
125
100
75
50
25
0
First Quartile
Second Quartile
Third Quartile

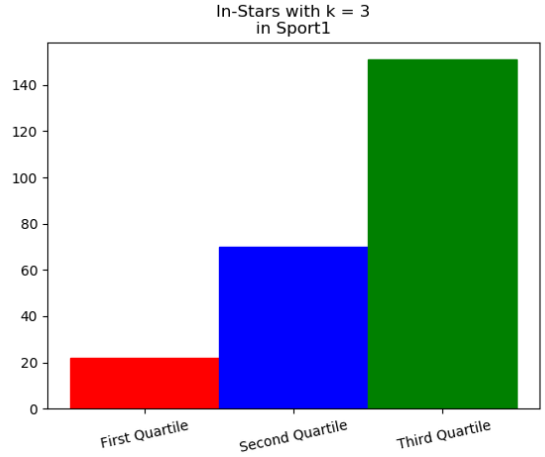

Out-Stars with k = 3
in News3
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

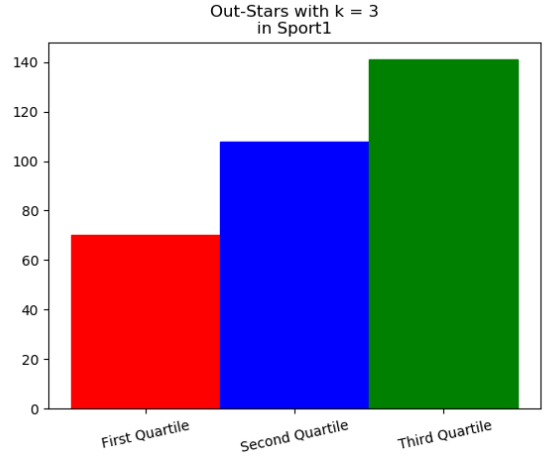

One-Way-Couples with k = 3
in News3
40
35
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

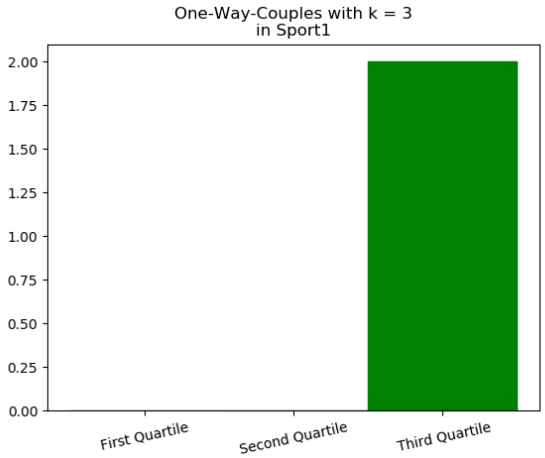

P-Triangles with k = 2
in News3
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

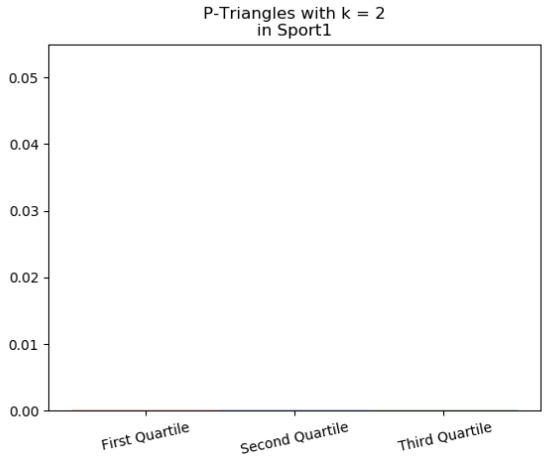

Chains with k = 3
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

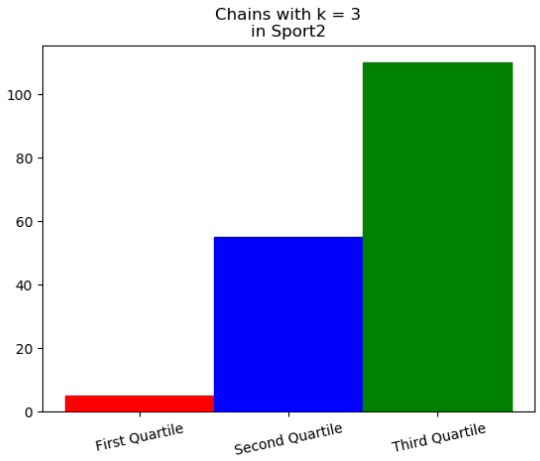

Ping-Pongs with k = 2
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

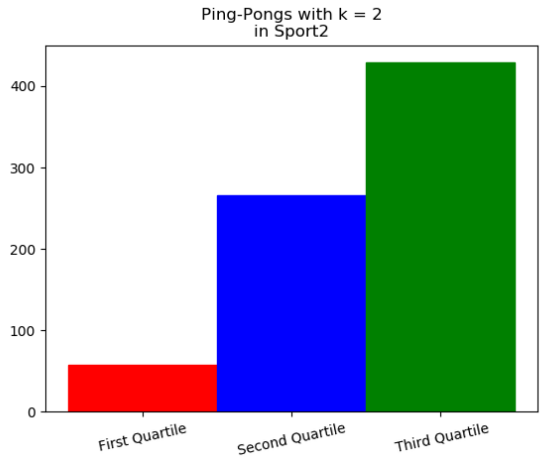

In-Stars with k = 3
in Sport1
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

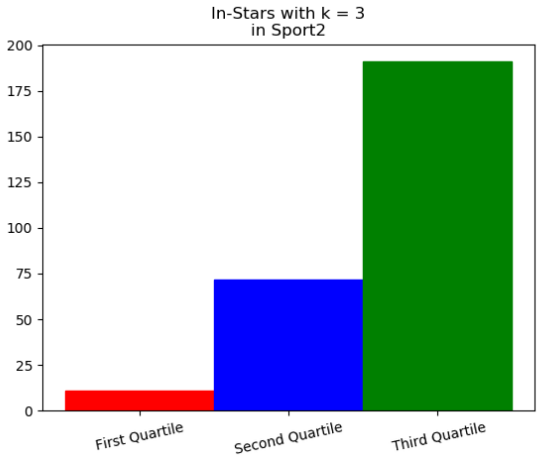

Out-Stars with k = 3
in Sport1
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

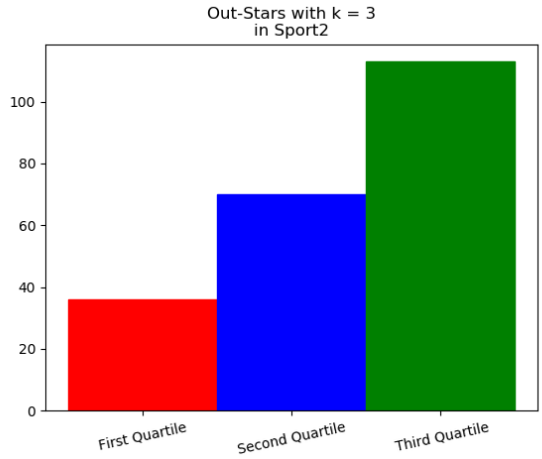

One-Way-Couples with k = 3
in Sport1
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
First Quartile
Second Quartile
Third Quartile

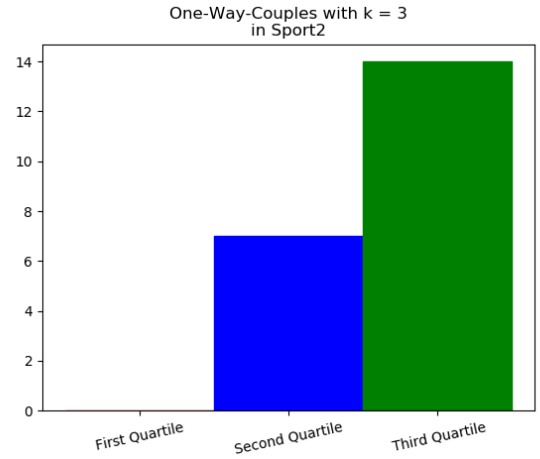

P-Triangles with k = 2
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

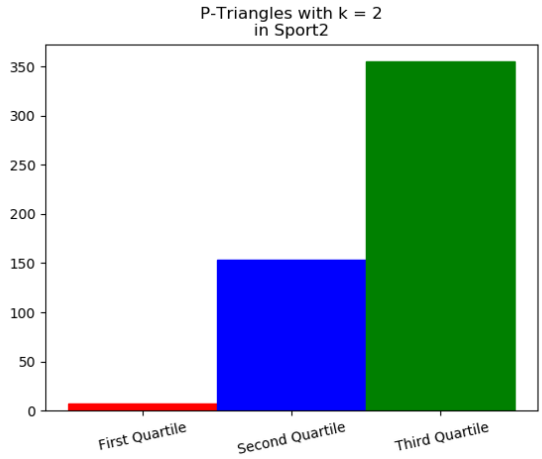

Chains with k = 3
in Sport2
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

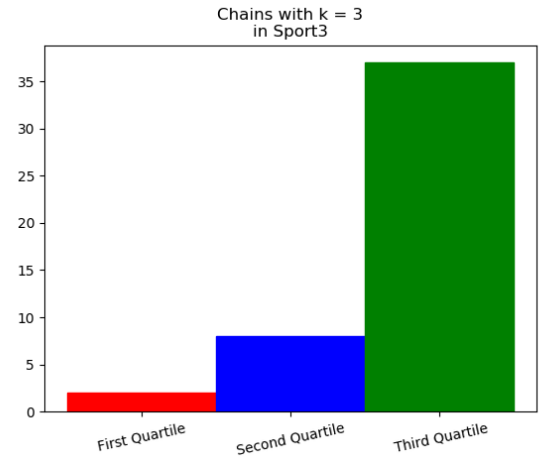

Ping-Pongs with k = 2
in Sport2
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

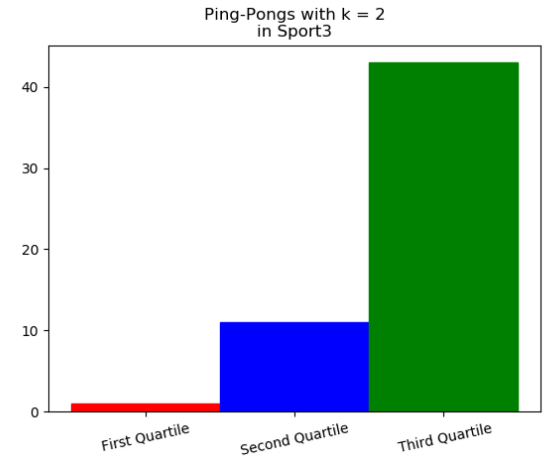

In-Stars with k = 3
in Sport2
200
175
150
125
100
75
50
25
0
First Quartile
Second Quartile
Third Quartile

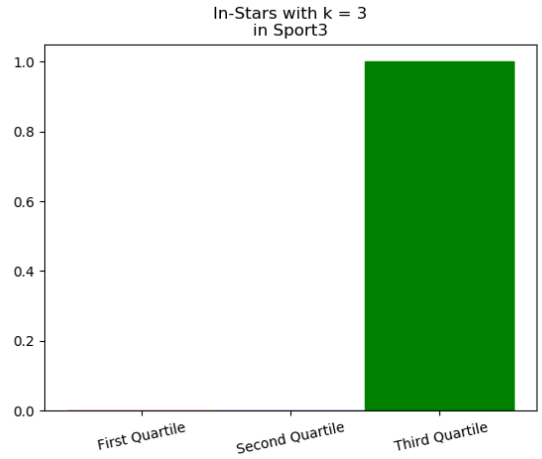

Out-Stars with k = 3
in Sport2
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

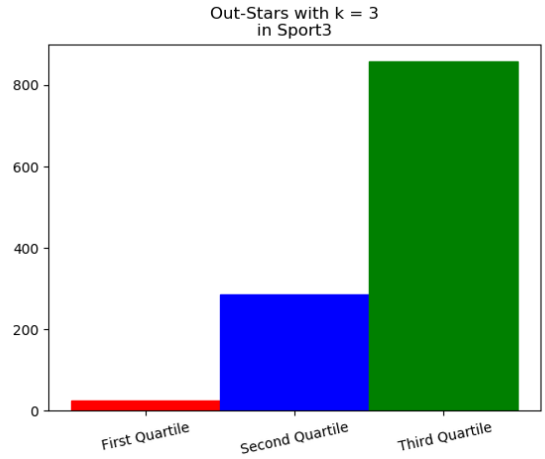

One-Way-Couples with k = 3
in Sport2
14
12
10
8
6
4
2
0
First Quartile
Second Quartile
Third Quartile

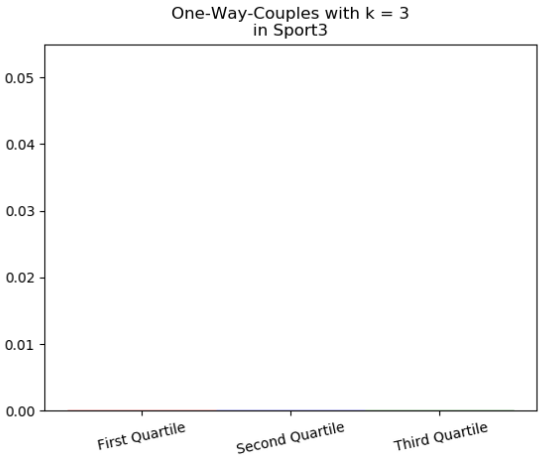

P-Triangles with k = 2
in Sport2
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

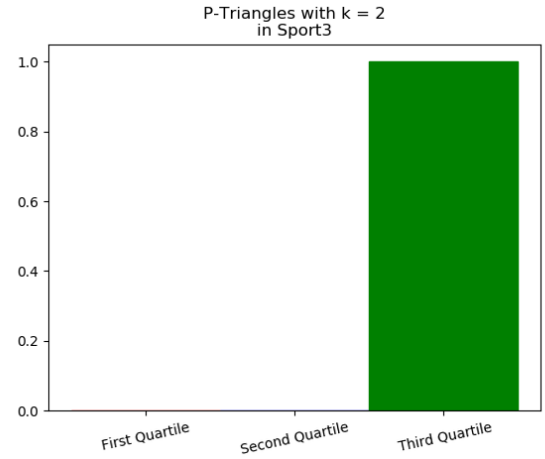

Chains with k = 3
in Sport3
35
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

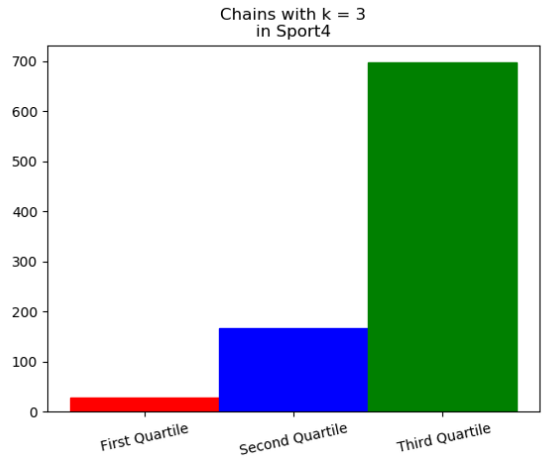

Ping-Pongs with k = 2
in Sport3
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

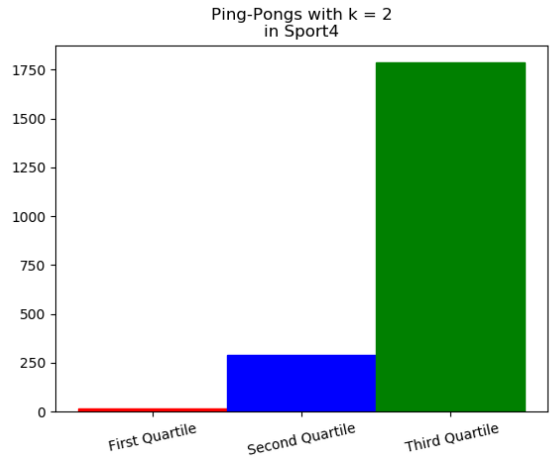

In-Stars with k = 3
in Sport3
1.0
0.8
0.6
0.4
0.2
0.0
First Quartile
Second Quartile
Third Quartile

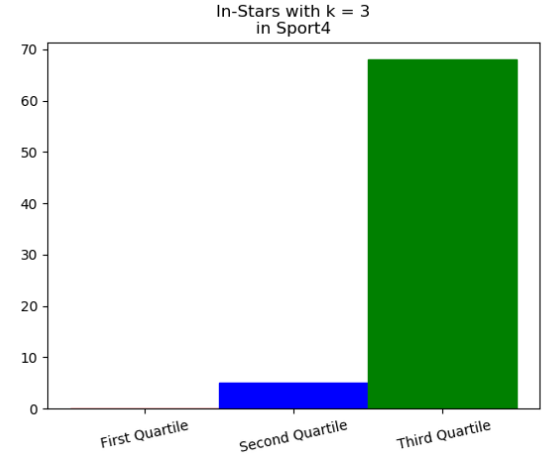

Out-Stars with k = 3
in Sport3
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

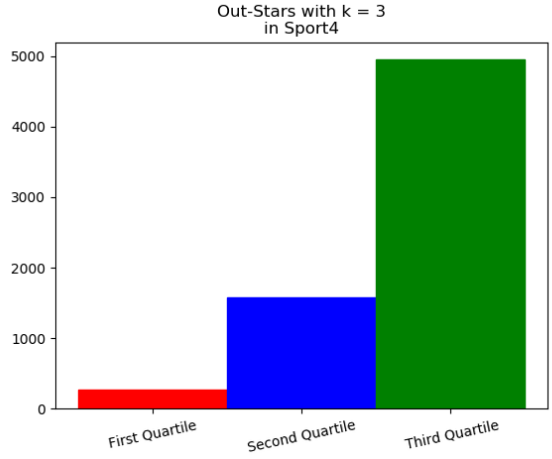

One-Way-Couples with k = 3
in Sport3
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

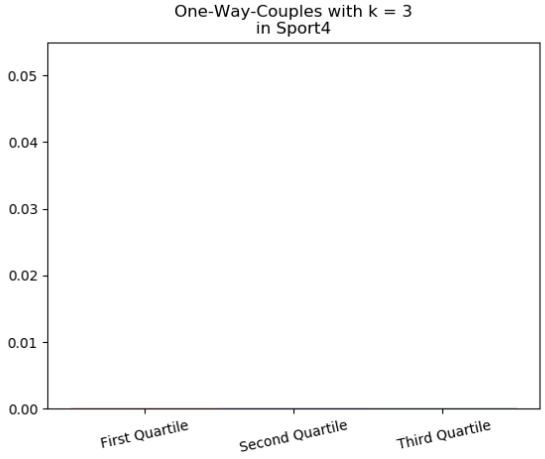

P-Triangles with k = 2
in Sport3
1.0
0.8
0.6
0.4
0.2
0.0
First Quartile
Second Quartile
Third Quartile

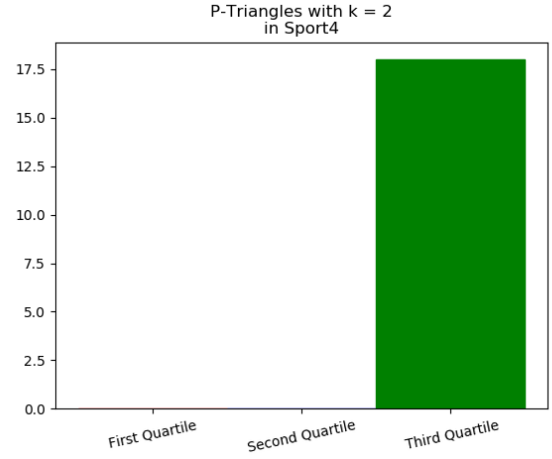

Chains with k = 3
in Sport4
700
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

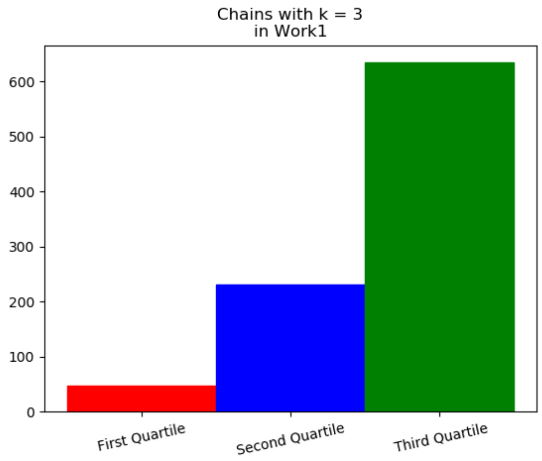

Ping-Pongs with k = 2
in Sport4
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

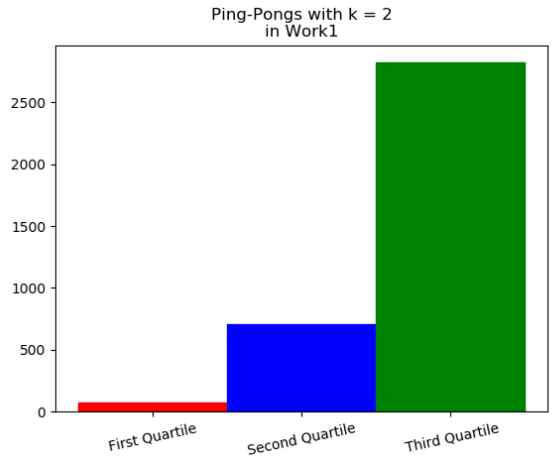

In-Stars with k = 3
in Sport4
70
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

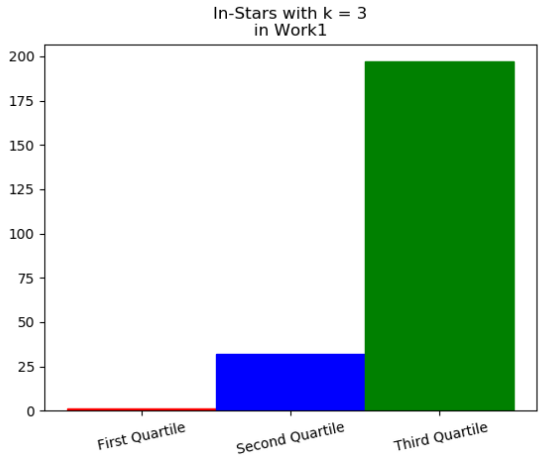

Out-Stars with k = 3
in Sport4
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

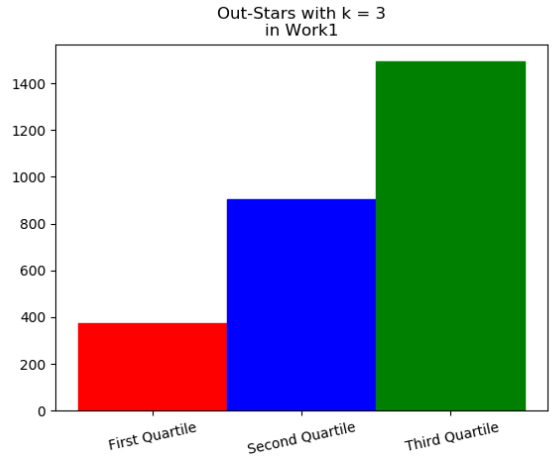

One-Way-Couples with k = 3
in Sport4
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

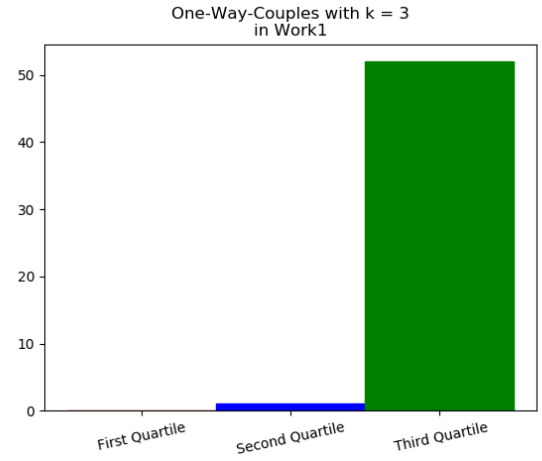

P-Triangles with k = 2
in Sport4
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
First Quartile
Second Quartile
Third Quartile

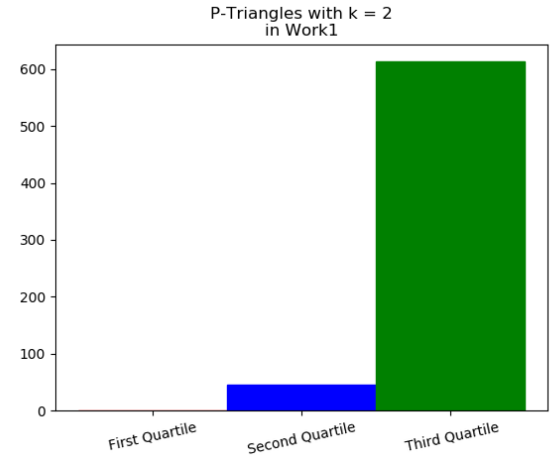

Chains with k = 3
in Work1
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

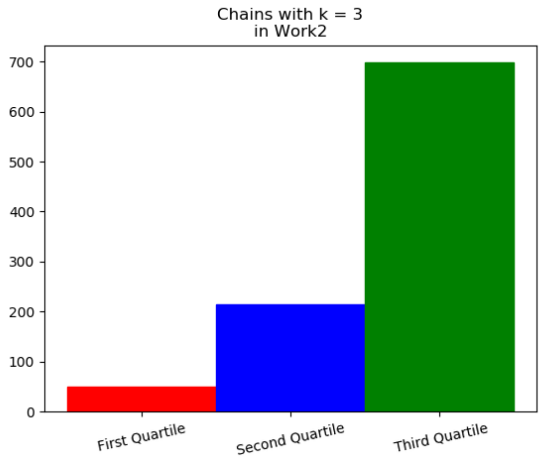

Ping-Pongs with k = 2
in Work1
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

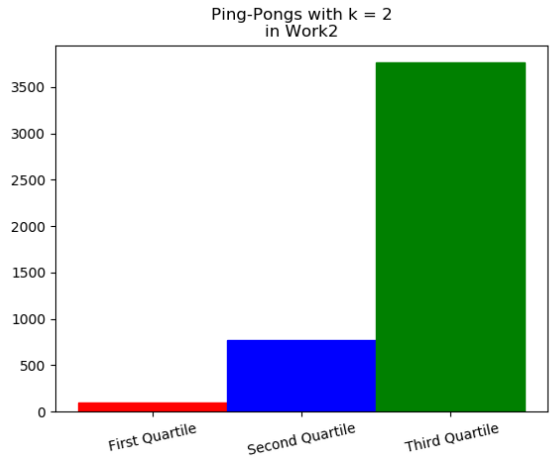

In-Stars with k = 3
in Work1
200
175
150
125
100
75
50
25
0
First Quartile
Second Quartile
Third Quartile

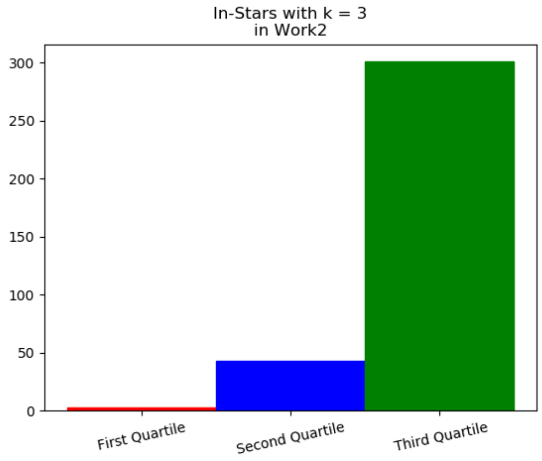

Out-Stars with k = 3
in Work1
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

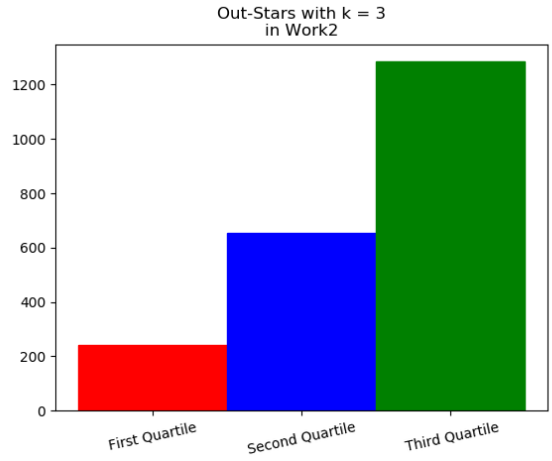

One-Way-Couples with k = 3
in Work1
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

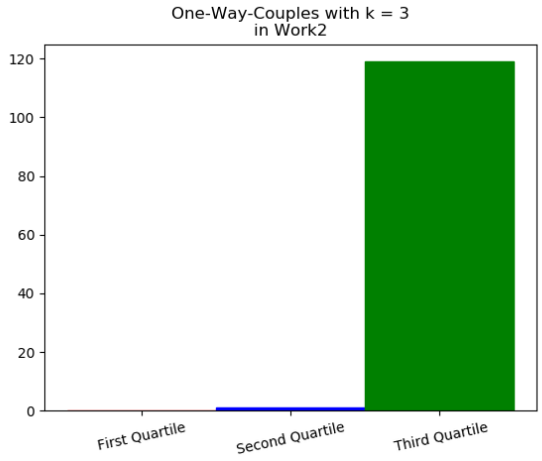

P-Triangles with k = 2
in Work1
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

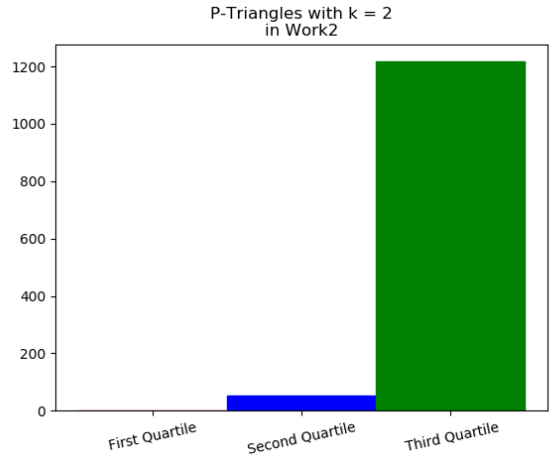

Chains with k = 3
in Work2
700
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

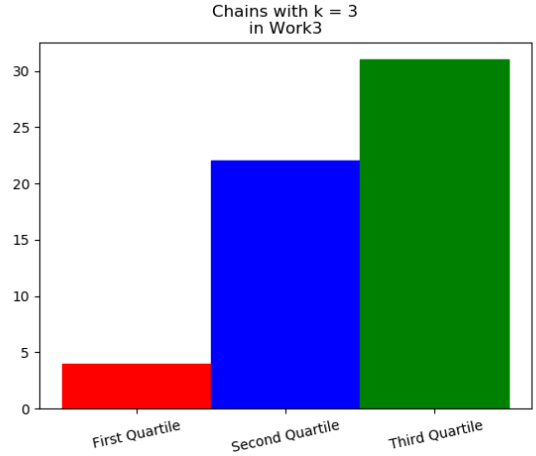

Ping-Pongs with k = 2
in Work2
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

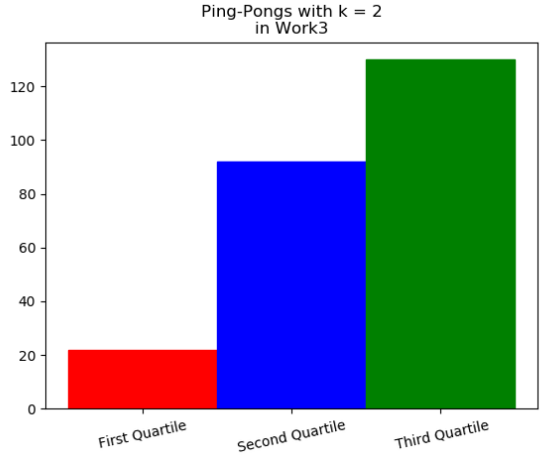

In-Stars with k = 3
in Work2
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

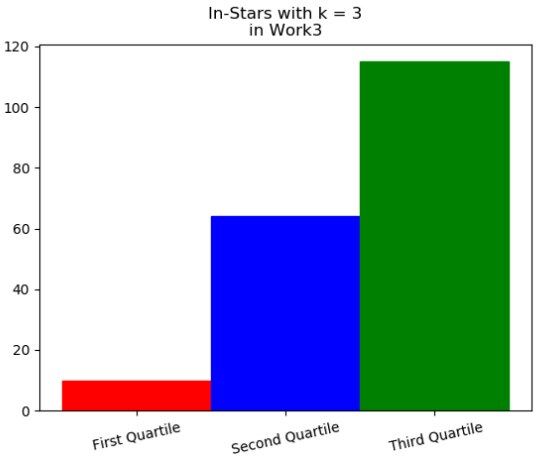

Out-Stars with k = 3
in Work2
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

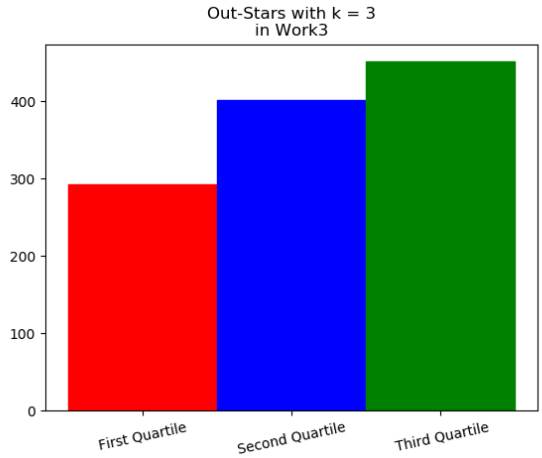

One-Way-Couples with k = 3
in Work2
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

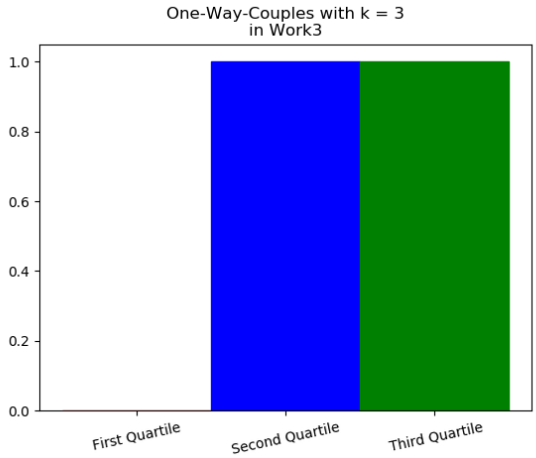

P-Triangles with k = 2
in Work2
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

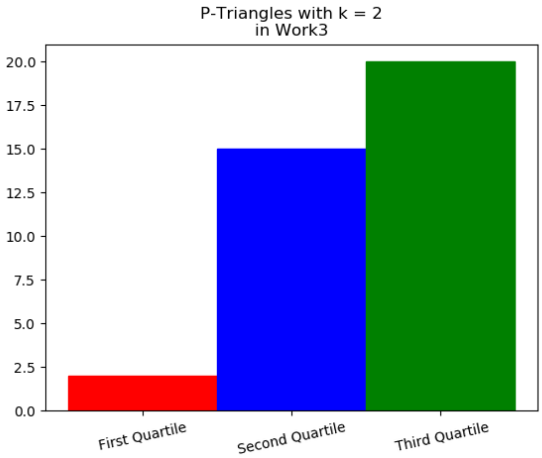

Chains with k = 3
in Work3
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile


Ping-Pongs with k = 2
in Work3
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile


In-Stars with k = 3
in Work3
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile


Out-Stars with k = 3
in Work3
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile


One-Way-Couples with k = 3
in Work3
1.0
0.8
0.6
0.4
0.2
0.0
First Quartile
Second Quartile
Third Quartile


P-Triangles with k = 2
in Work3
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
First Quartile
Second Quartile
Third Quartile

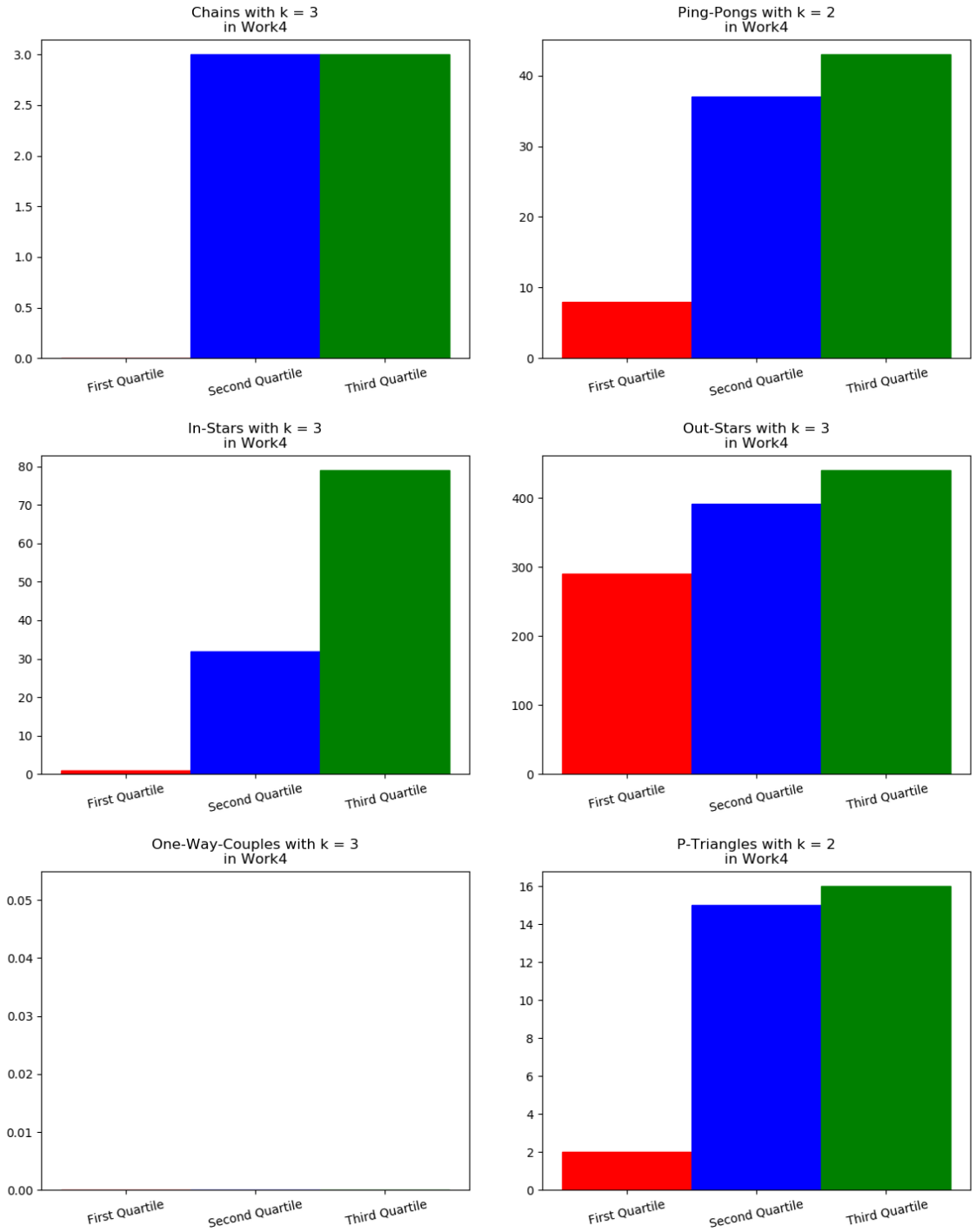
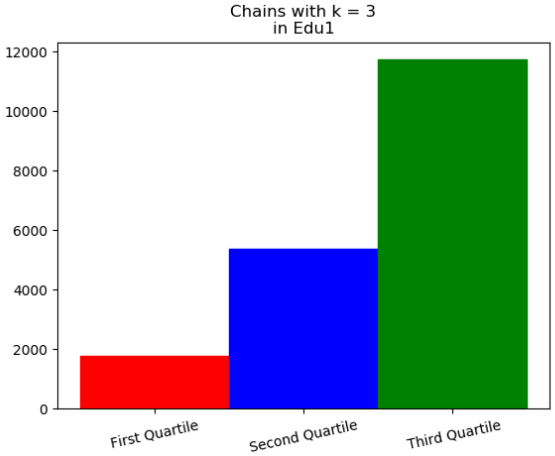
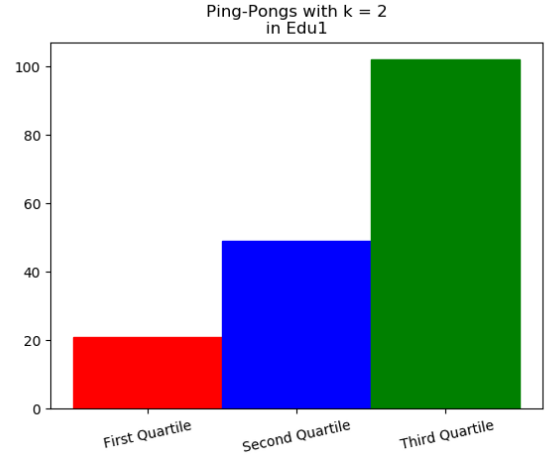
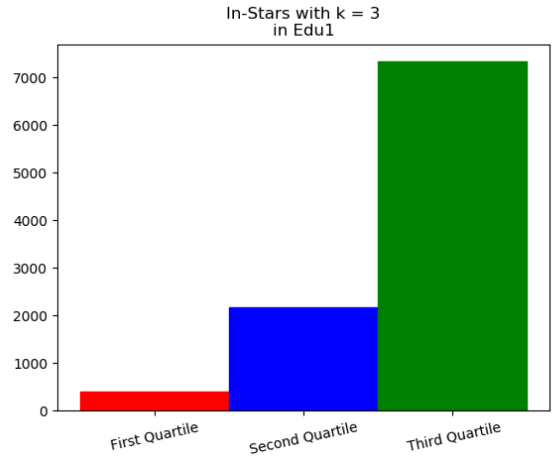


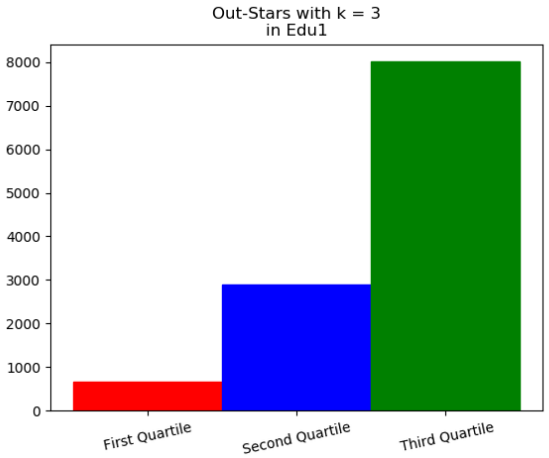
Figure A.2: The plots of the results obtained for all groups modeled with the Post Graph

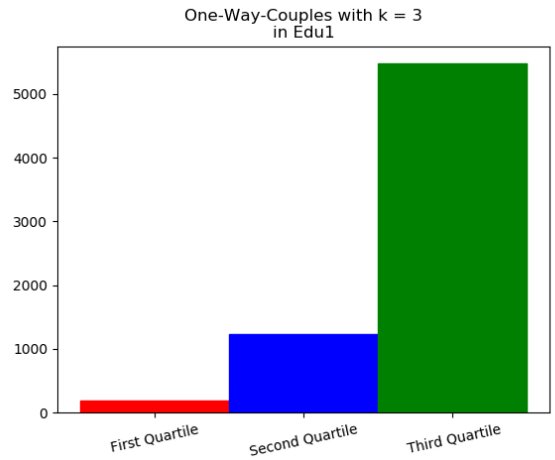
In **Figure A.3** we show the group-based diagrams elaborated on the results of the number of motifs found in the User Graph with the time window sliding approach.

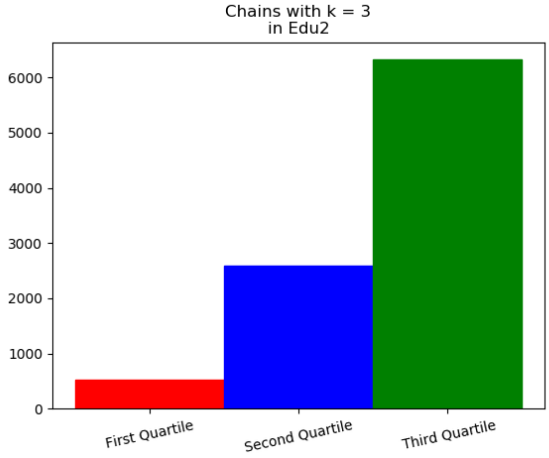

Chains with k = 3
in Edu1
12000
10000
8000
6000
4000
2000
0
First Quartile
Second Quartile
Third Quartile

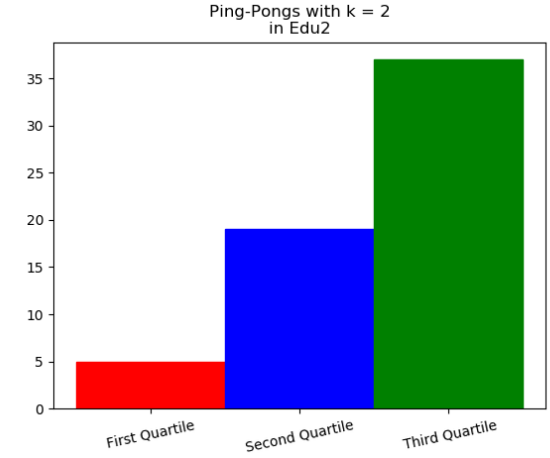

Ping-Pongs with k = 2
in Edu1
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

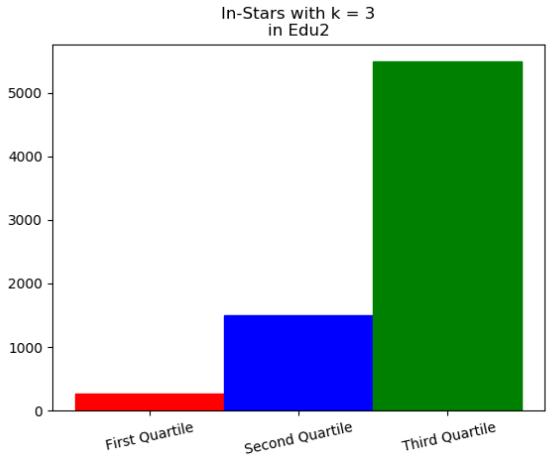

In-Stars with k = 3
in Edu1
7000
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

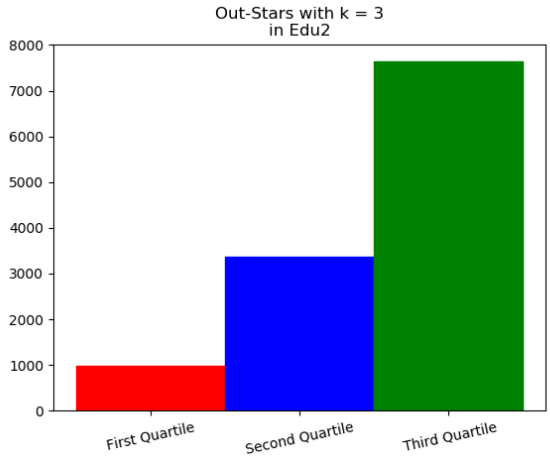

Out-Stars with k = 3
in Edu1
8000
7000
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

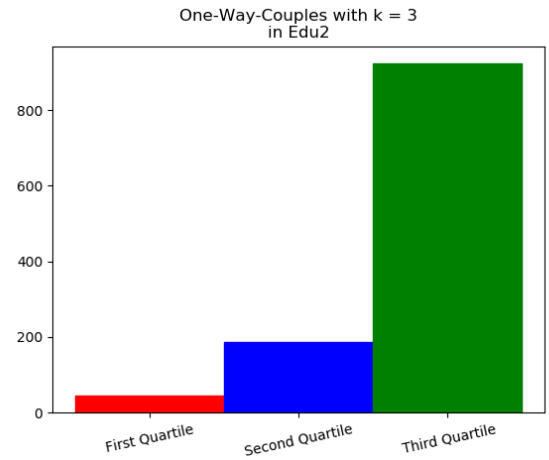

One-Way-Couples with k = 3
in Edu1
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

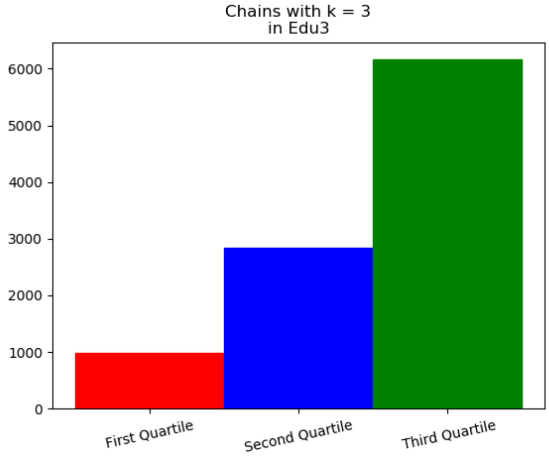

Chains with k = 3
in Edu2
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

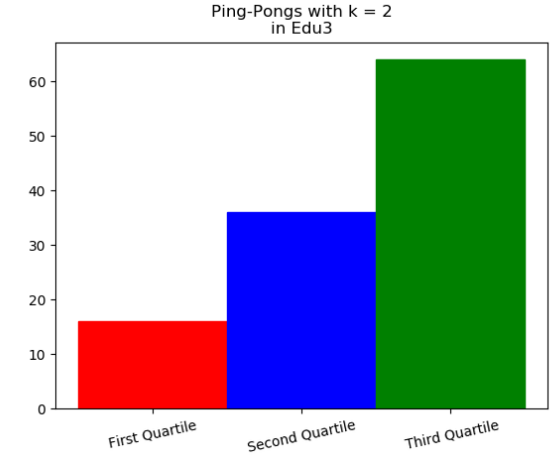

Ping-Pongs with k = 2
in Edu2
35
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

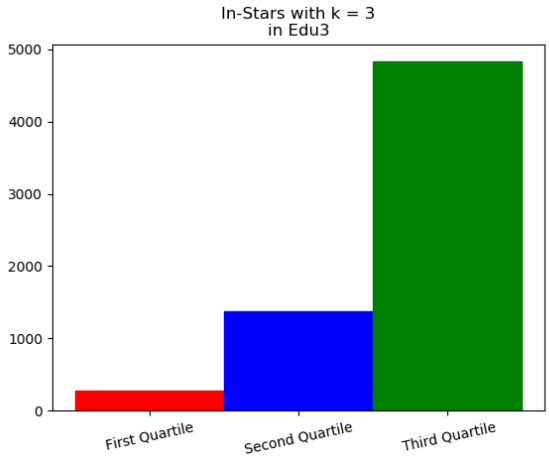

In-Stars with k = 3
in Edu2
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

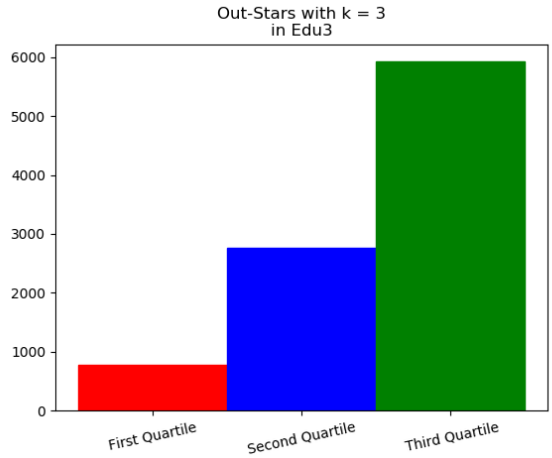

Out-Stars with k = 3
in Edu2
8000
7000
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

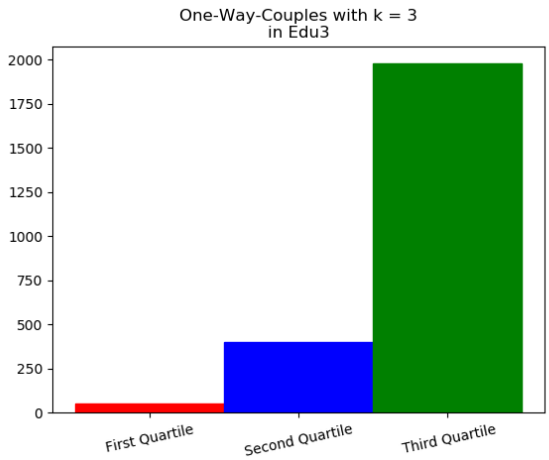

One-Way-Couples with k = 3
in Edu2
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

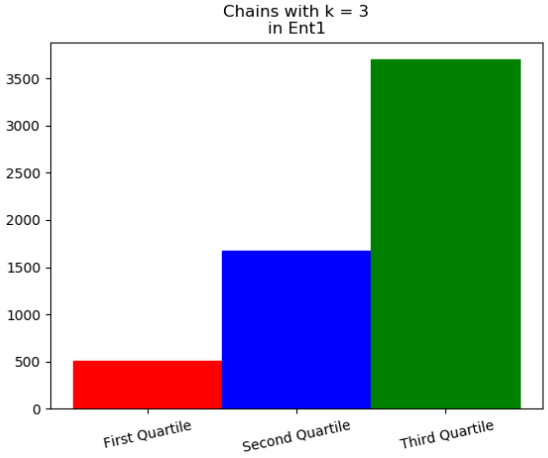

Chains with k = 3
in Edu3
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

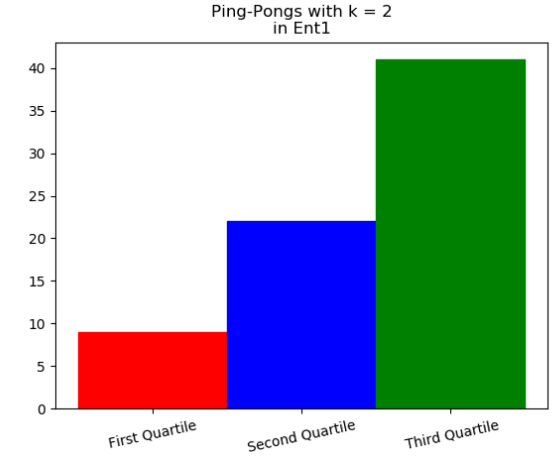

Ping-Pongs with k = 2
in Edu3
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

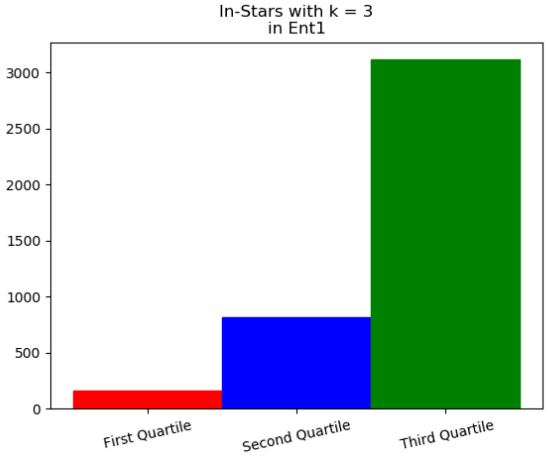

In-Stars with k = 3
in Edu3
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

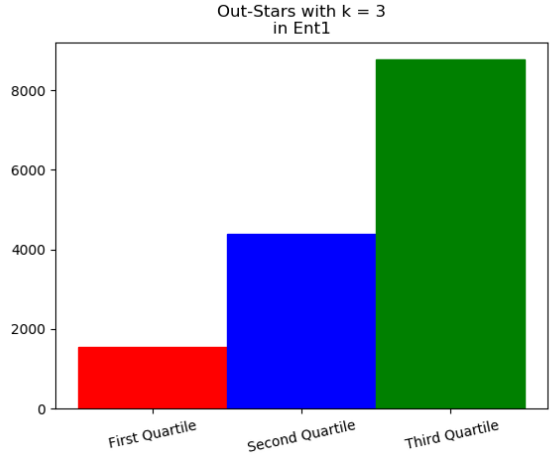

Out-Stars with k = 3
in Edu3
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

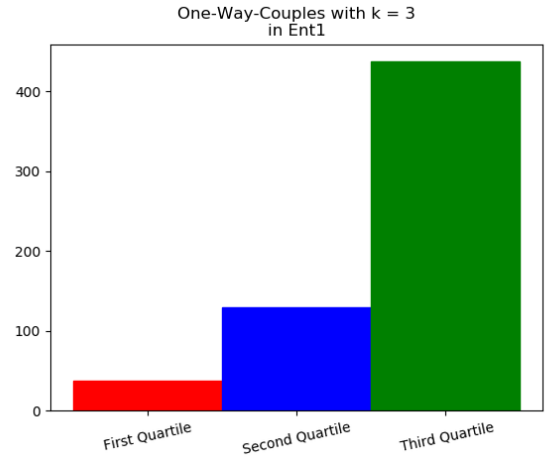

One-Way-Couples with k = 3
in Edu3
2000
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

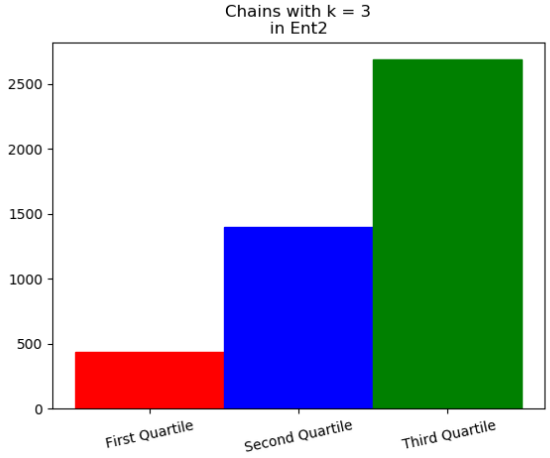

Chains with k = 3
in Ent1
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

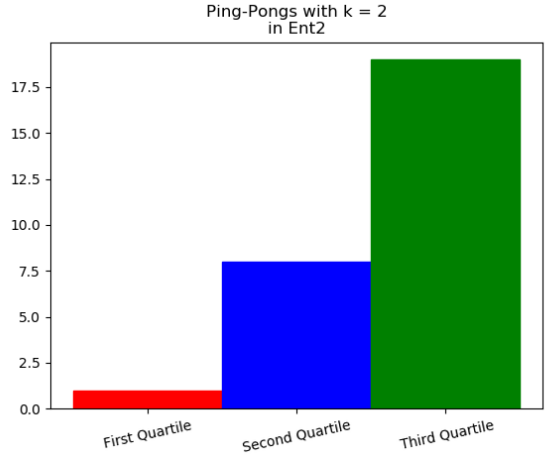

Ping-Pongs with k = 2
in Ent1
40
35
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

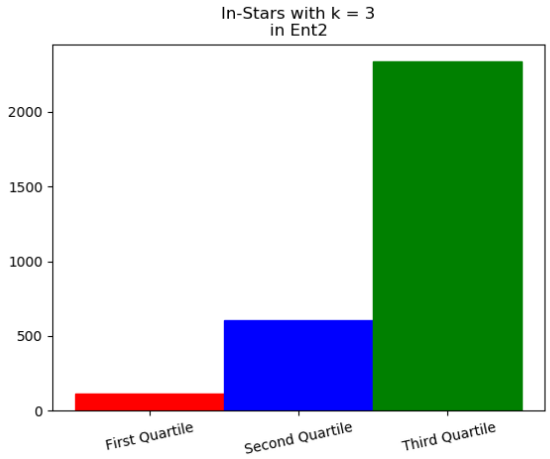

In-Stars with k = 3
in Ent1
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

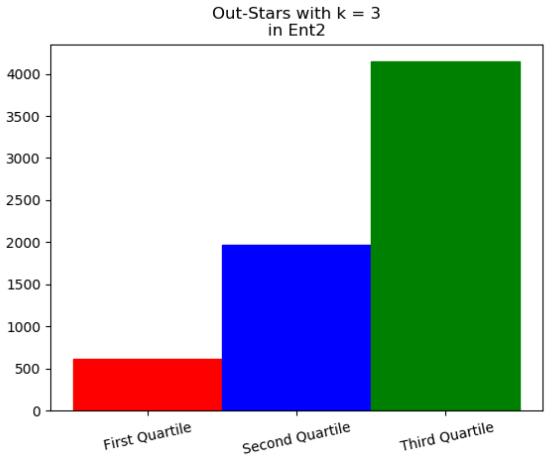

Out-Stars with k = 3
in Ent1
8000
6000
4000
2000
0
First Quartile
Second Quartile
Third Quartile

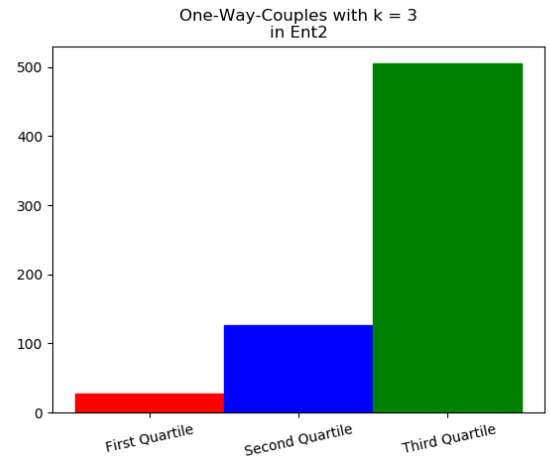

One-Way-Couples with k = 3
in Ent1
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

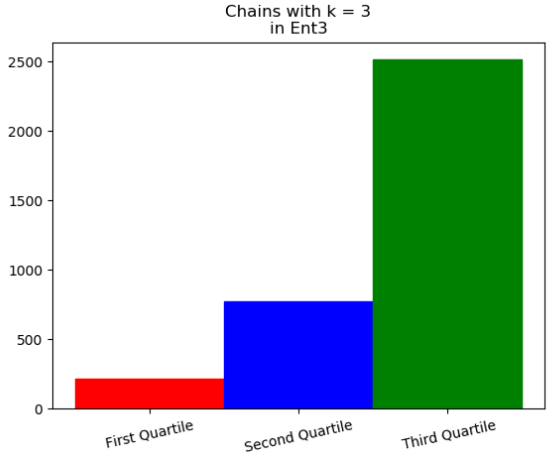

Chains with k = 3
in Ent2
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

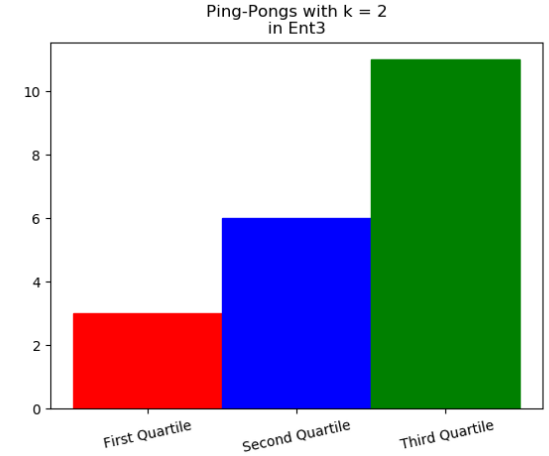

Ping-Pongs with k = 2
in Ent2
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
First Quartile
Second Quartile
Third Quartile

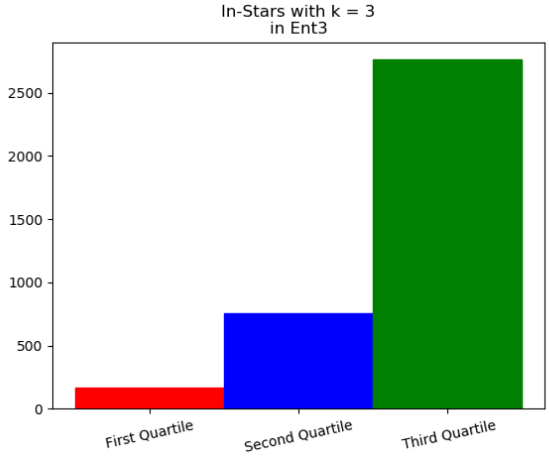

In-Stars with k = 3
in Ent2
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

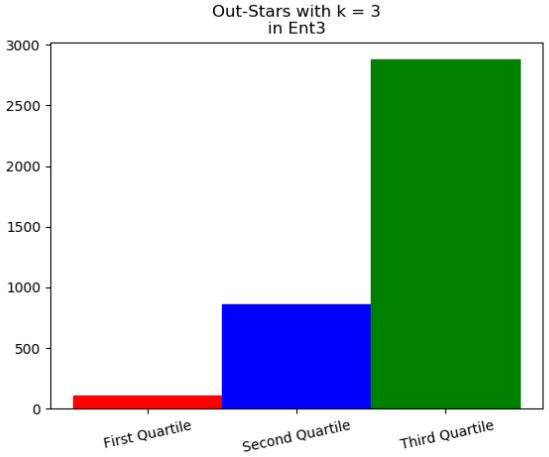

Out-Stars with k = 3
in Ent2
4000
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

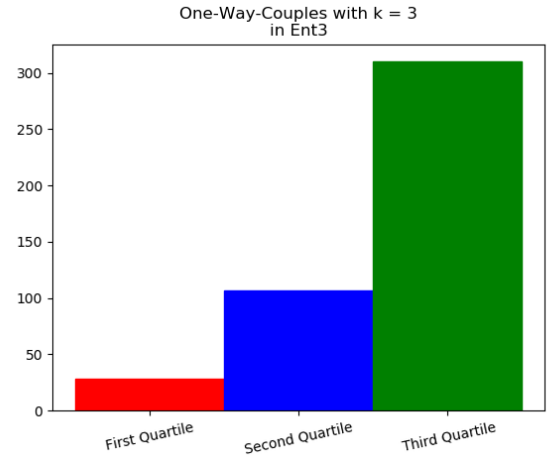

One-Way-Couples with k = 3
in Ent2
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

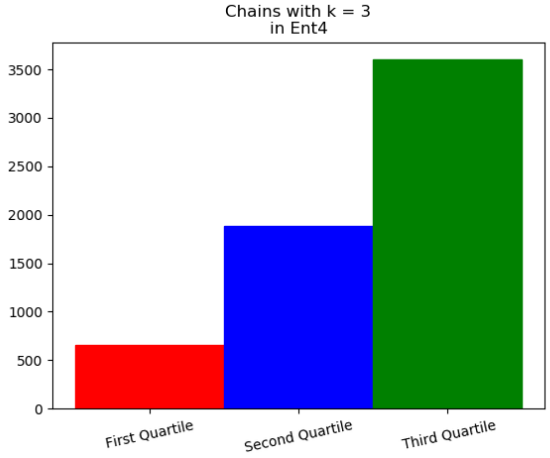

Chains with k = 3
in Ent3
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

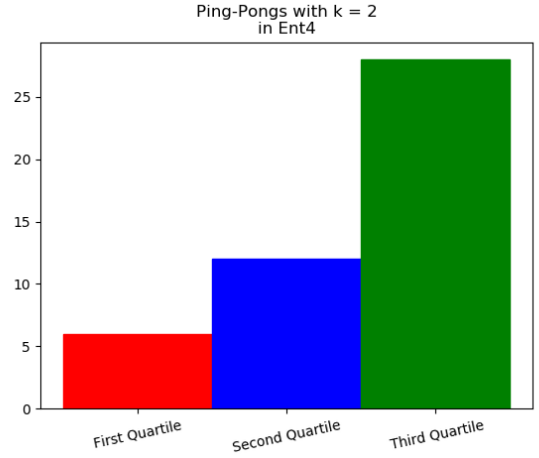

Ping-Pongs with k = 2
in Ent3
10
8
6
4
2
0
First Quartile
Second Quartile
Third Quartile

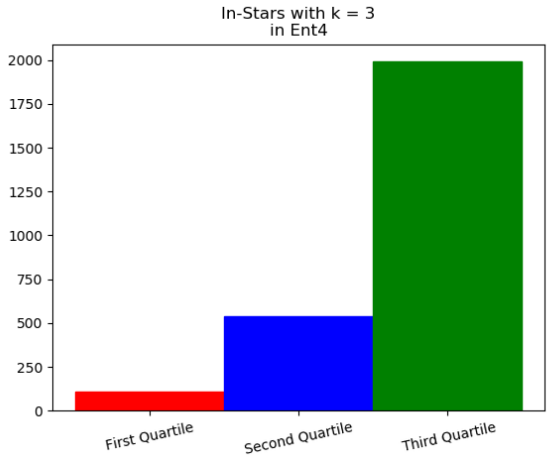

In-Stars with k = 3
in Ent3
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

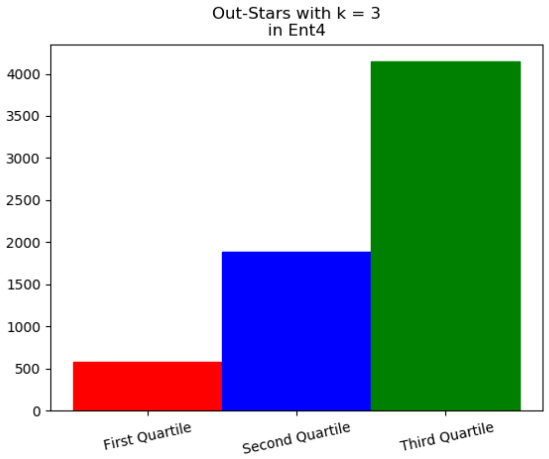

Out-Stars with k = 3
in Ent3
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

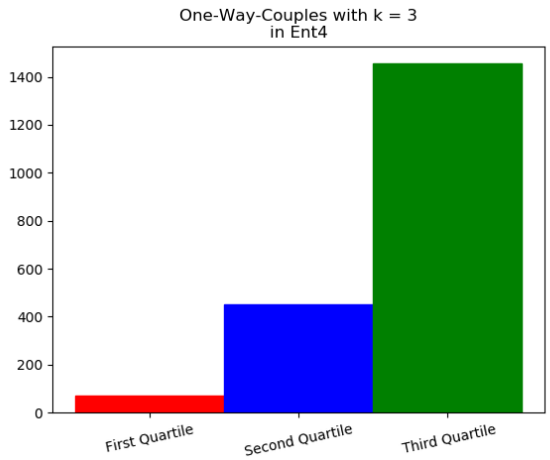

One-Way-Couples with k = 3
in Ent3
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

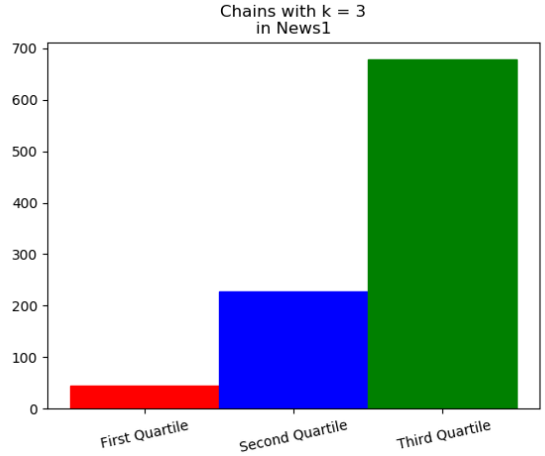

Chains with k = 3
in Ent4
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

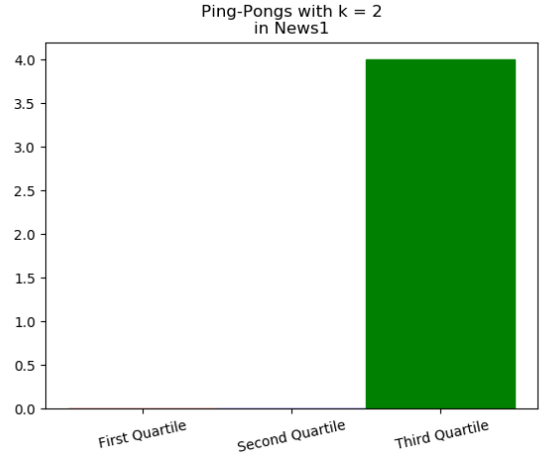

Ping-Pongs with k = 2
in Ent4
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

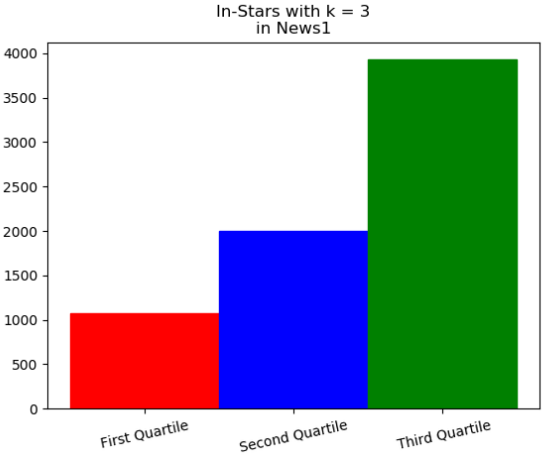

In-Stars with k = 3
in Ent4
2000
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

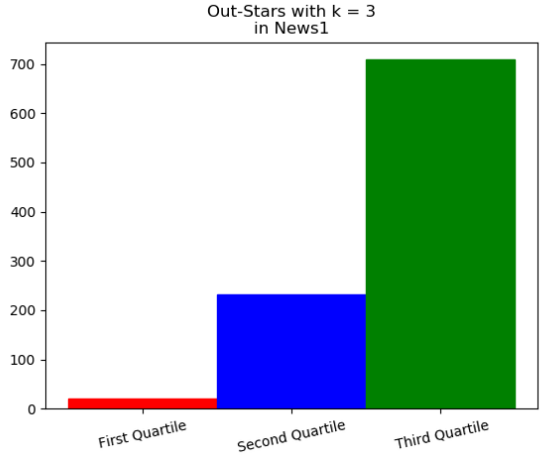

Out-Stars with k = 3
in Ent4
4000
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

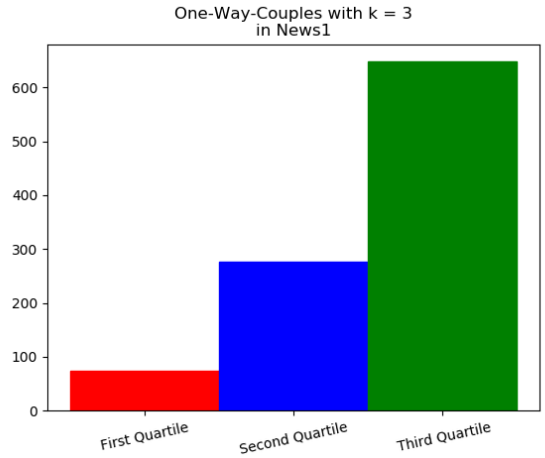

One-Way-Couples with k = 3
in Ent4
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

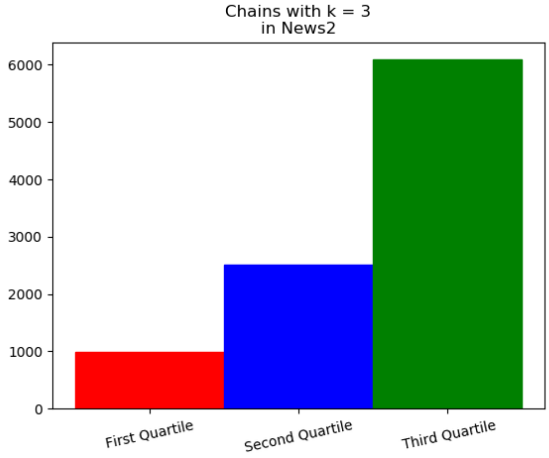

Chains with k = 3
in News1
700
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

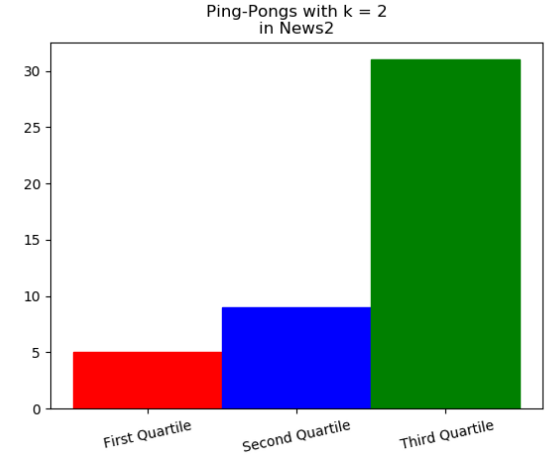

Ping-Pongs with k = 2
in News1
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
First Quartile
Second Quartile
Third Quartile

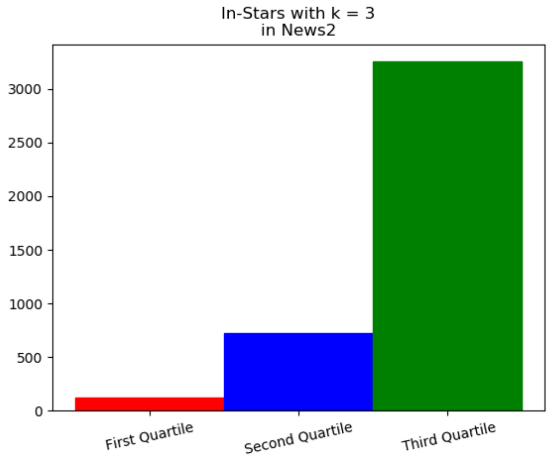

In-Stars with k = 3
in News1
4000
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

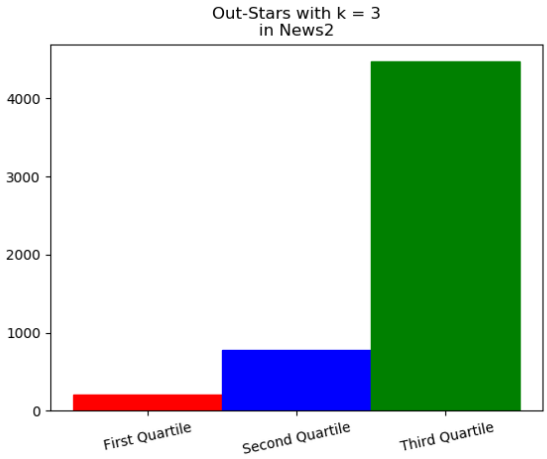

Out-Stars with k = 3
in News1
700
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

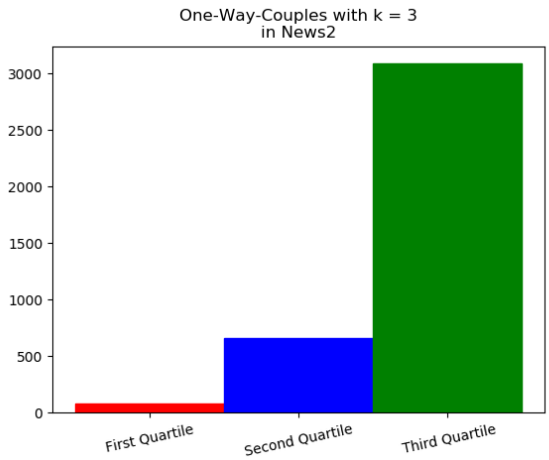

One-Way-Couples with k = 3
in News1
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

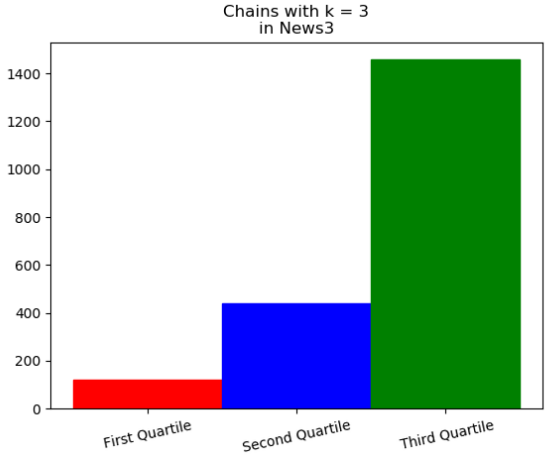

Chains with k = 3
in News2
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

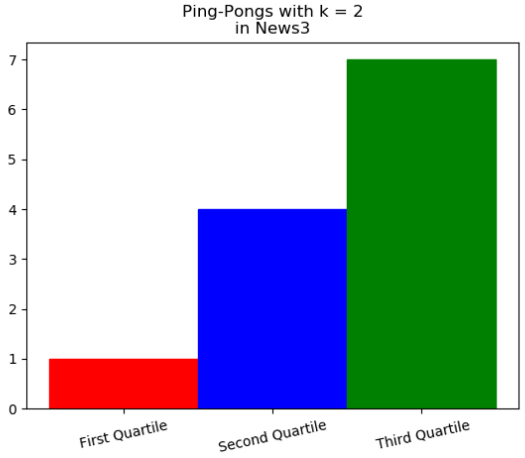

Ping-Pongs with k = 2
in News2
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

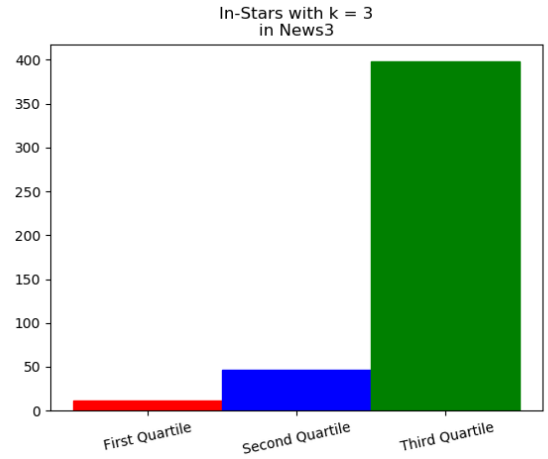

In-Stars with k = 3
in News2
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

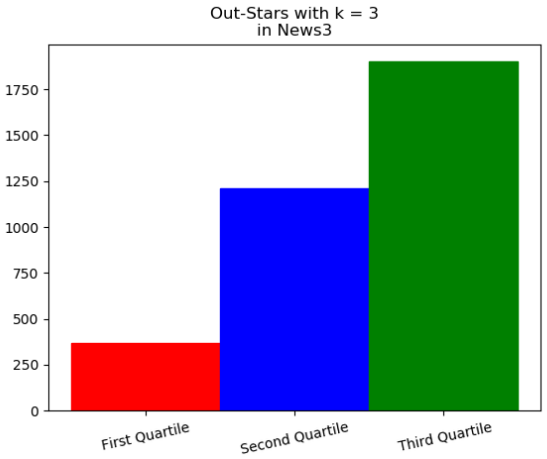

Out-Stars with k = 3
in News2
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

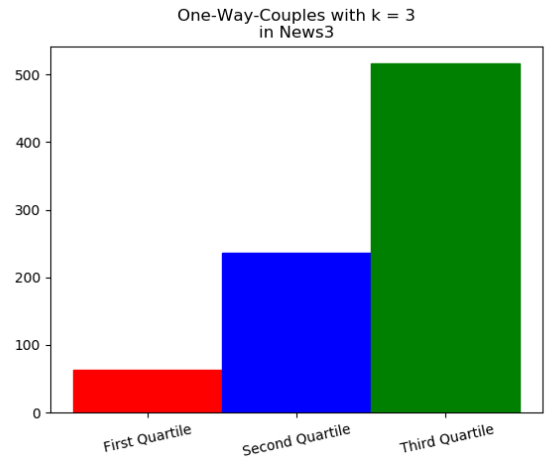

One-Way-Couples with k = 3
in News2
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

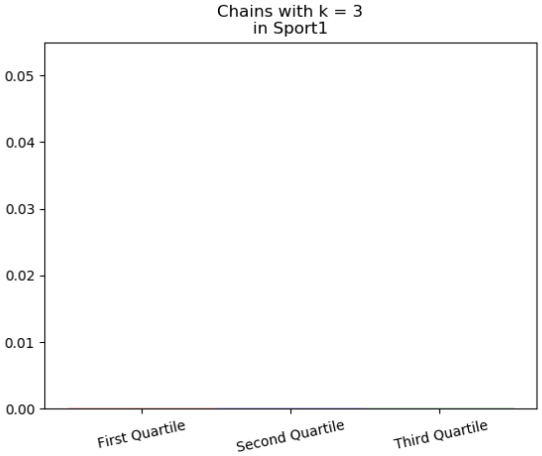

Chains with k = 3
in News3
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

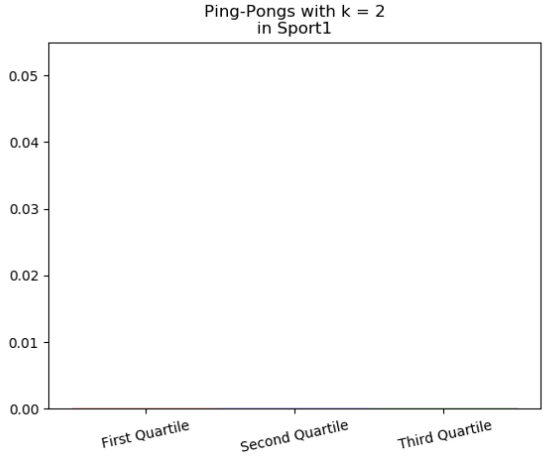

Ping-Pongs with k = 2
in News3
7
6
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

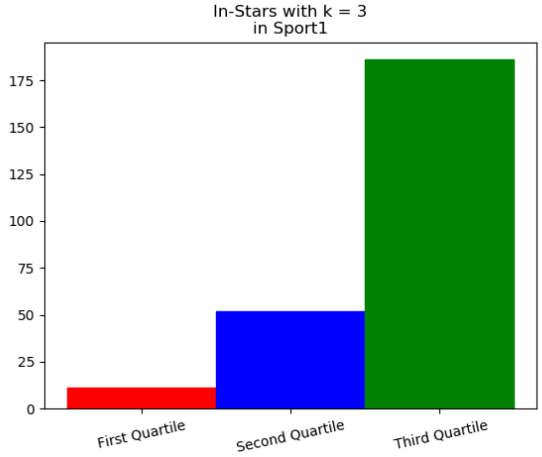

In-Stars with k = 3
in News3
400
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

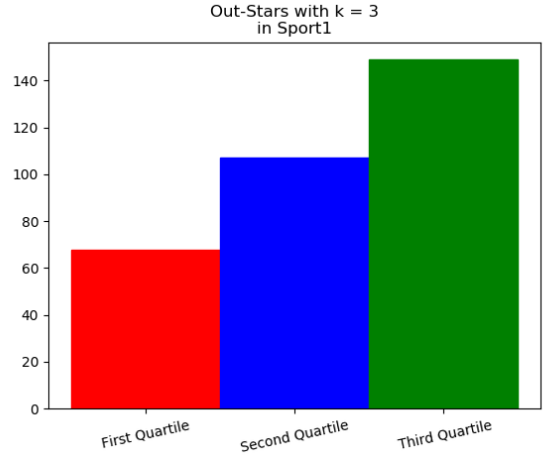

Out-Stars with k = 3
in News3
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

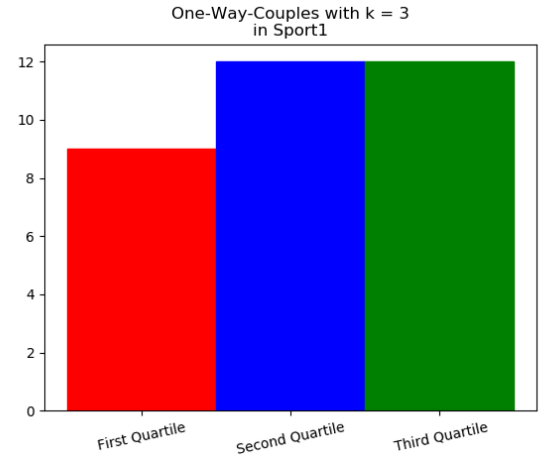

One-Way-Couples with k = 3
in News3
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

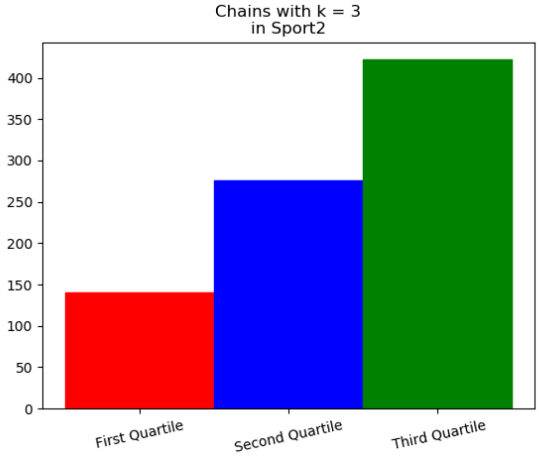

Chains with k = 3
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

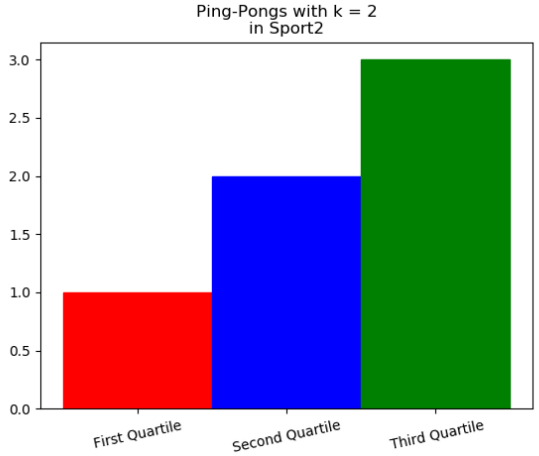

Ping-Pongs with k = 2
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

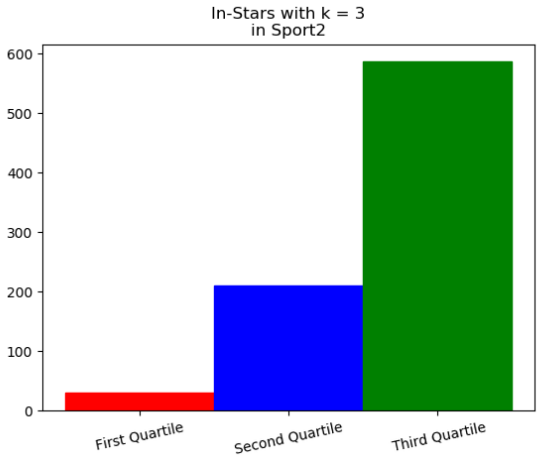

In-Stars with k = 3
in Sport1
175
150
125
100
75
50
25
0
First Quartile
Second Quartile
Third Quartile

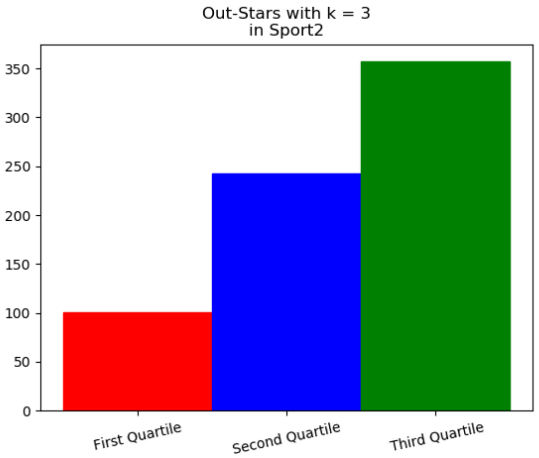

Out-Stars with k = 3
in Sport1
140
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

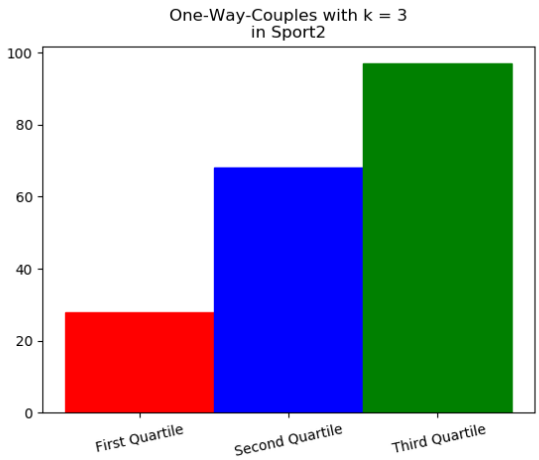

One-Way-Couples with k = 3
in Sport1
12
10
8
6
4
2
0
First Quartile
Second Quartile
Third Quartile

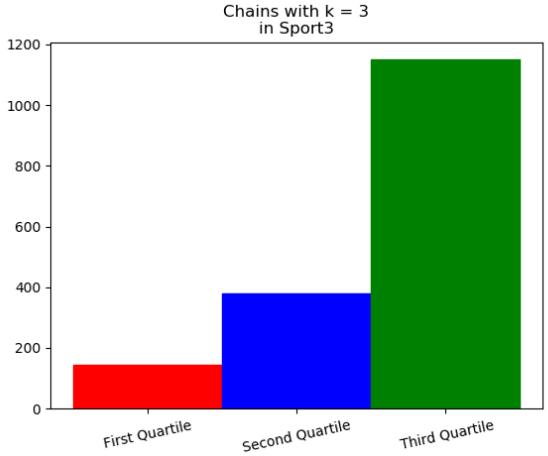

Chains with k = 3
in Sport2
400
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

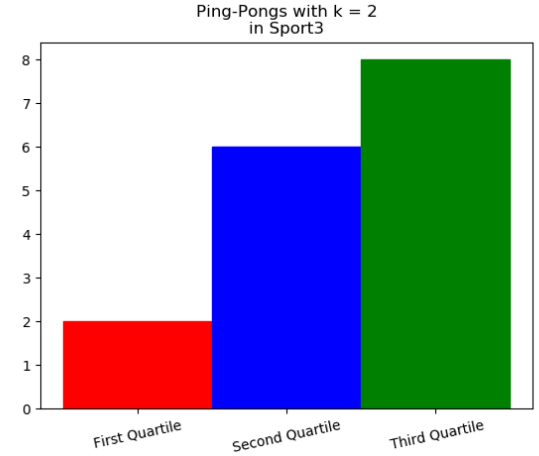

Ping-Pongs with k = 2
in Sport2
3.0
2.5
2.0
1.5
1.0
0.5
0.0
First Quartile
Second Quartile
Third Quartile

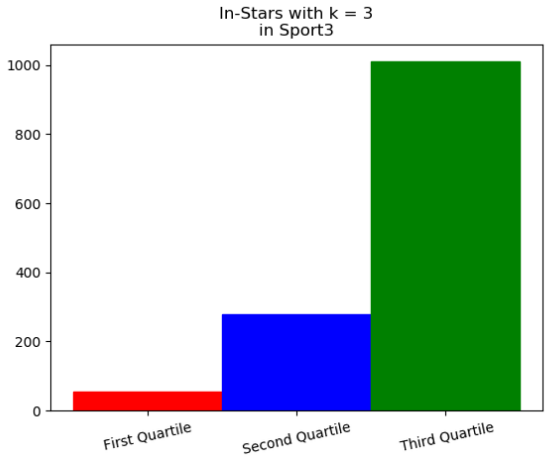

In-Stars with k = 3
in Sport2
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

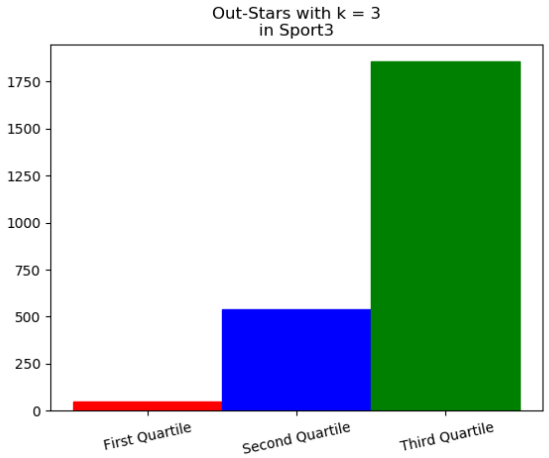

Out-Stars with k = 3
in Sport2
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

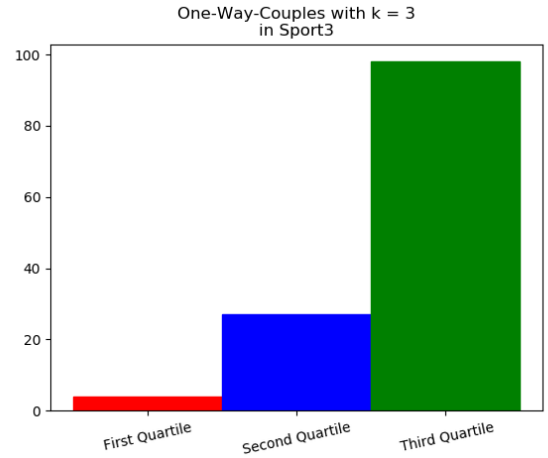

One-Way-Couples with k = 3
in Sport2
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

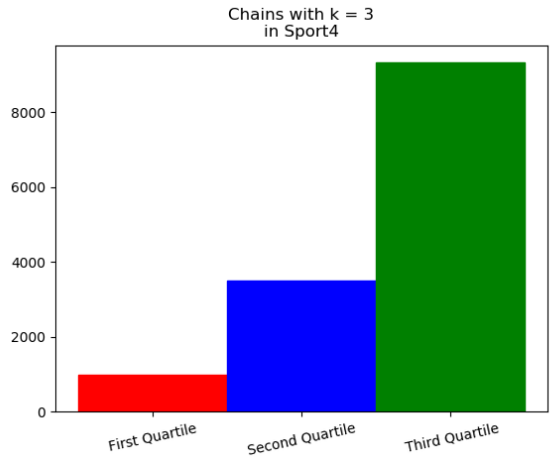

Chains with k = 3
in Sport3
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

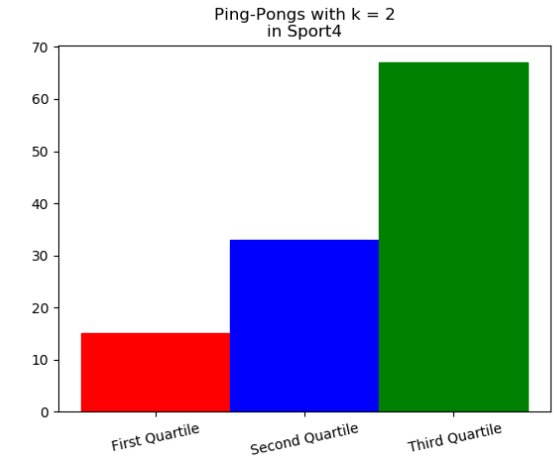

Ping-Pongs with k = 2
in Sport3
8
7
6
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

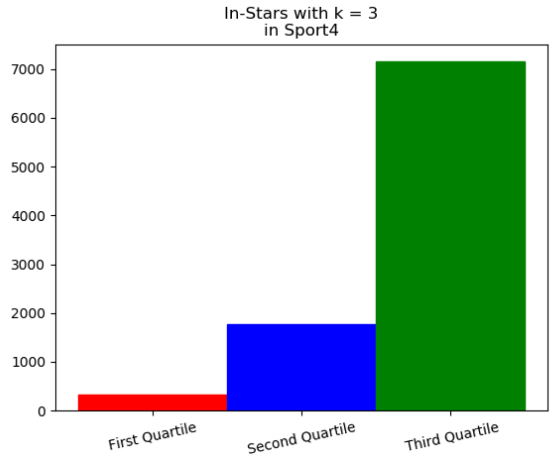

In-Stars with k = 3
in Sport3
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

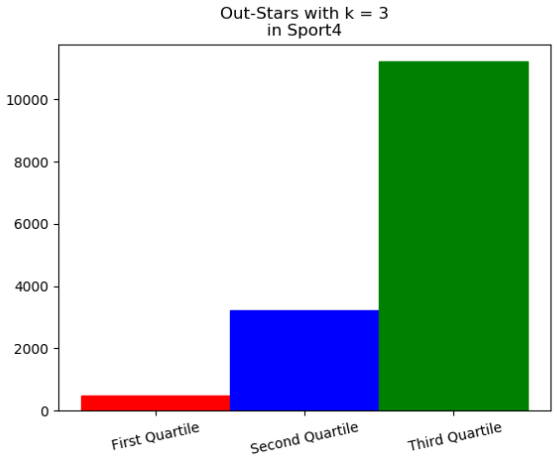

Out-Stars with k = 3
in Sport3
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

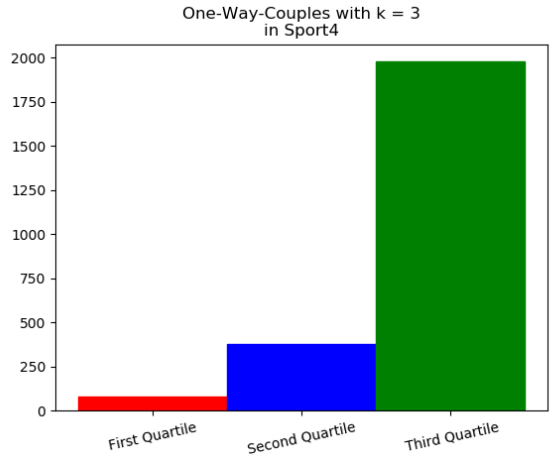

One-Way-Couples with k = 3
in Sport3
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

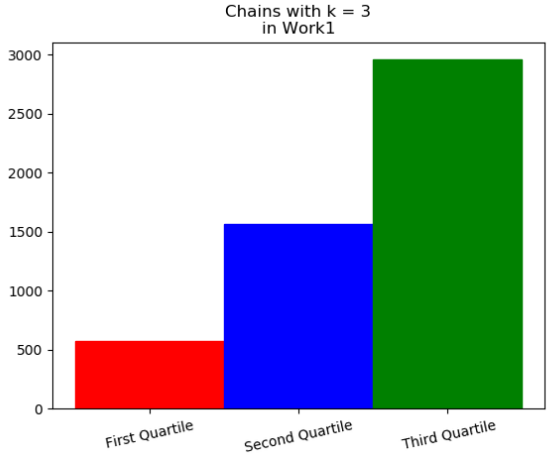

Chains with k = 3
in Sport4
8000
6000
4000
2000
0
First Quartile
Second Quartile
Third Quartile

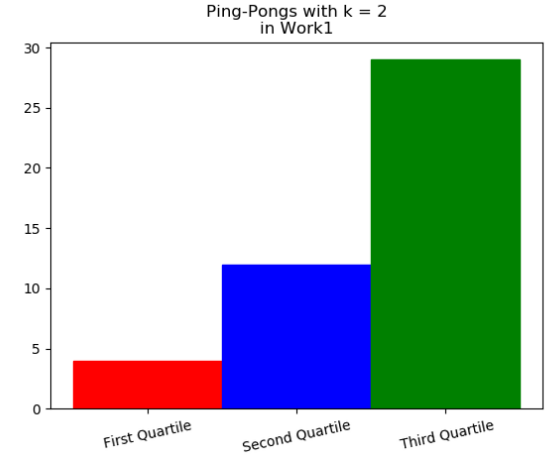

Ping-Pongs with k = 2
in Sport4
70
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

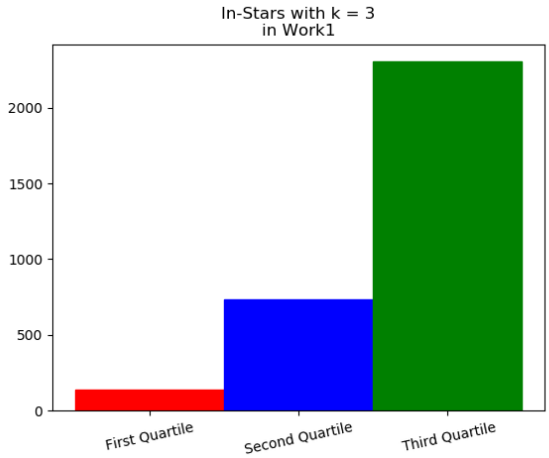

In-Stars with k = 3
in Sport4
7000
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

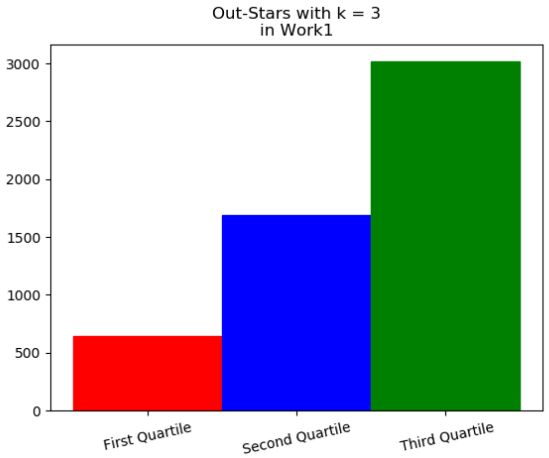

Out-Stars with k = 3
in Sport4
10000
8000
6000
4000
2000
0
First Quartile
Second Quartile
Third Quartile

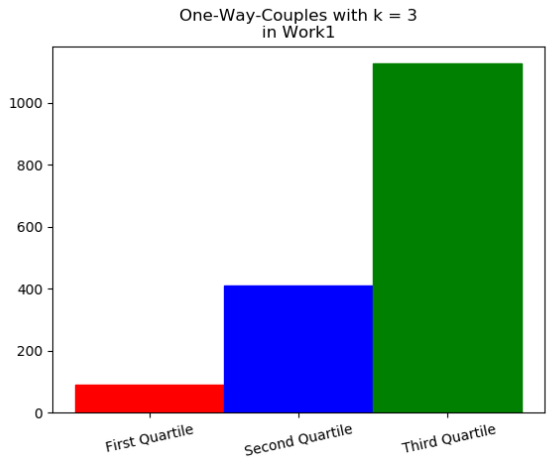

One-Way-Couples with k = 3
in Sport4
2000
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

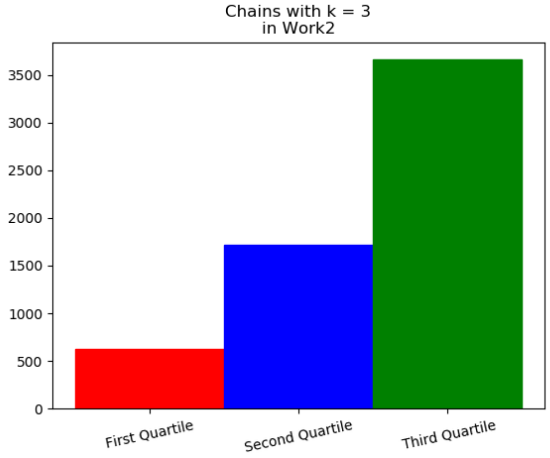

Chains with k = 3
in Work1
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

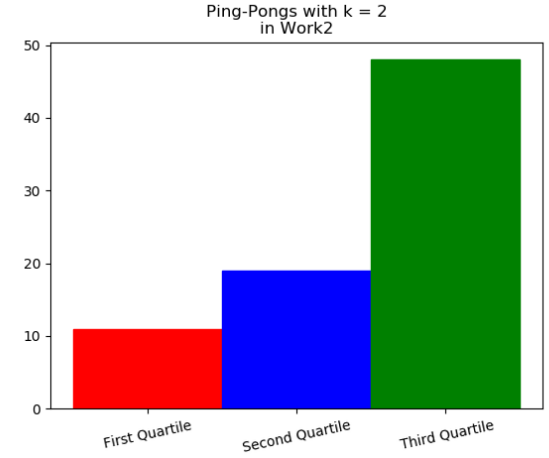

Ping-Pongs with k = 2
in Work1
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

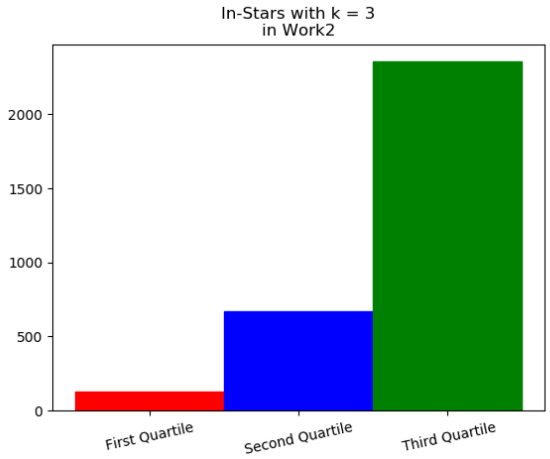

In-Stars with k = 3
in Work1
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

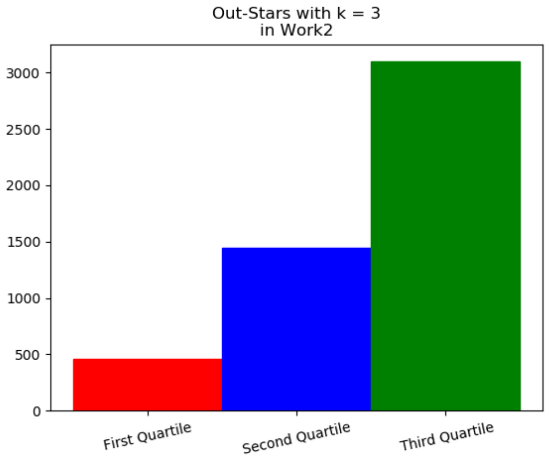

Out-Stars with k = 3
in Work1
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

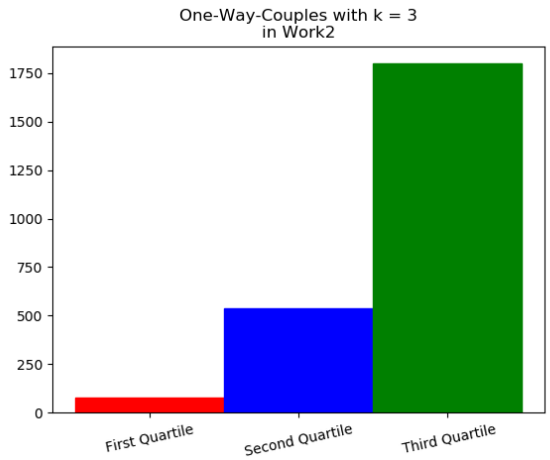

One-Way-Couples with k = 3
in Work1
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

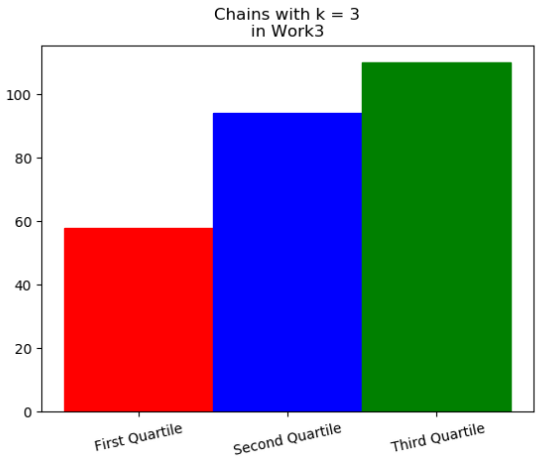

Chains with k = 3
in Work2
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

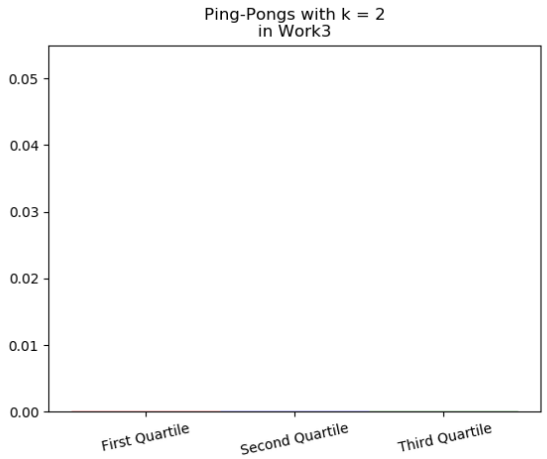

Ping-Pongs with k = 2
in Work2
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

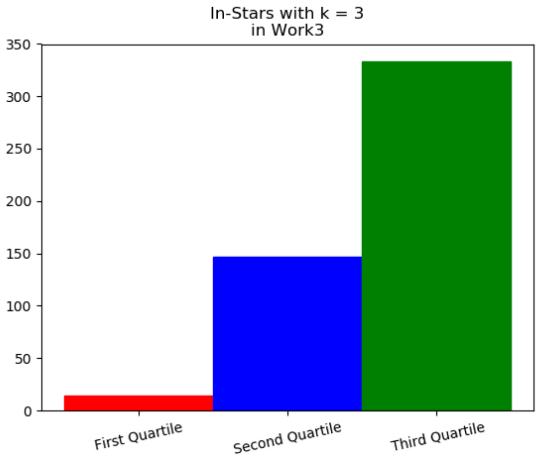

In-Stars with k = 3
in Work2
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

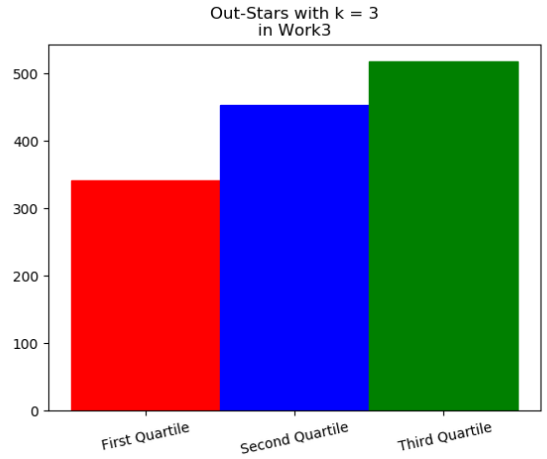

Out-Stars with k = 3
in Work2
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

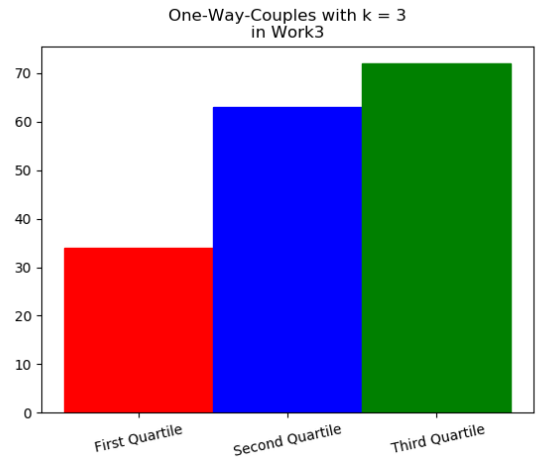

One-Way-Couples with k = 3
in Work2
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile


Chains with k = 3
in Work3
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile


Ping-Pongs with k = 2
in Work3
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile


In-Stars with k = 3
in Work3
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile


Out-Stars with k = 3
in Work3
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile


One-Way-Couples with k = 3
in Work3
70
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

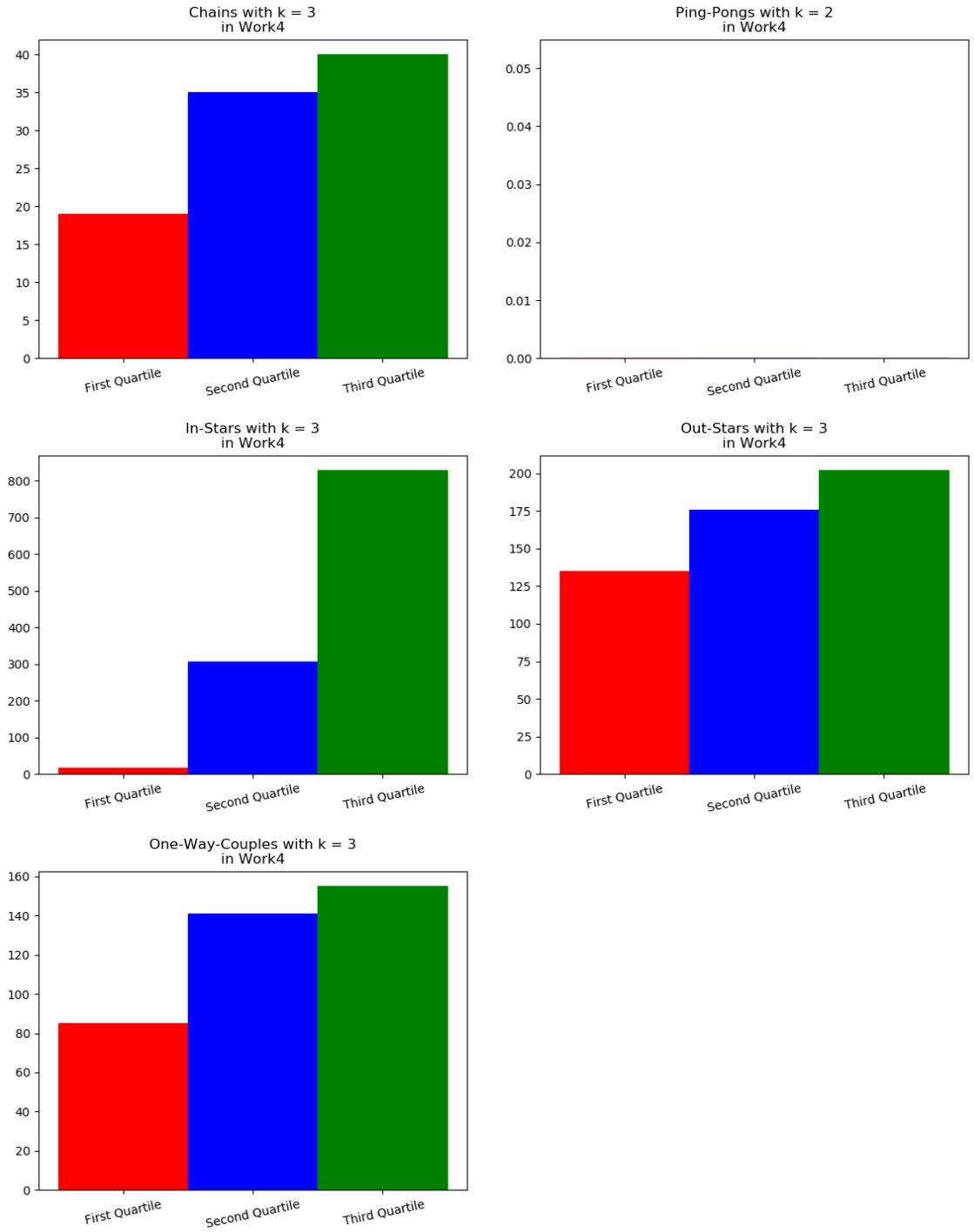
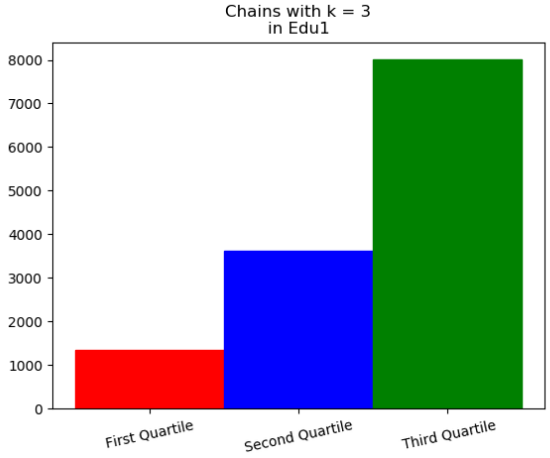
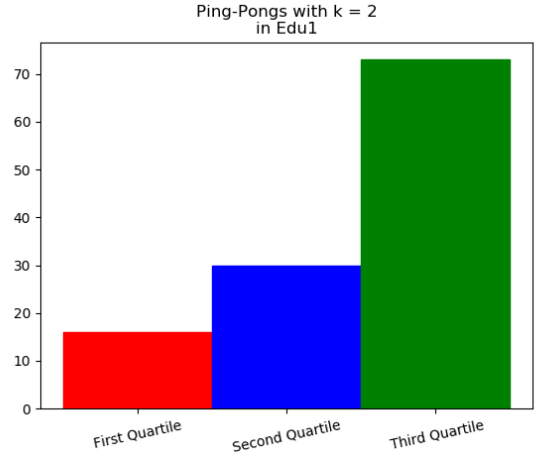
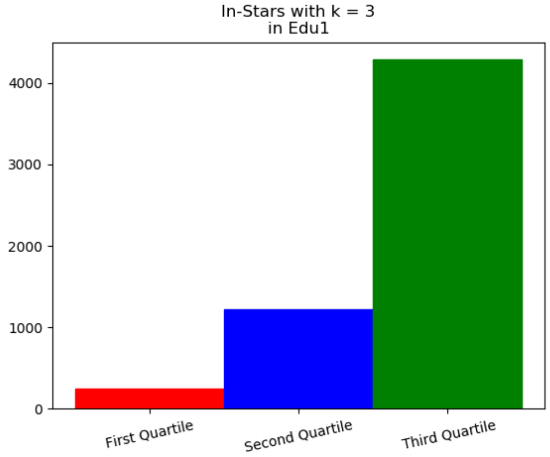


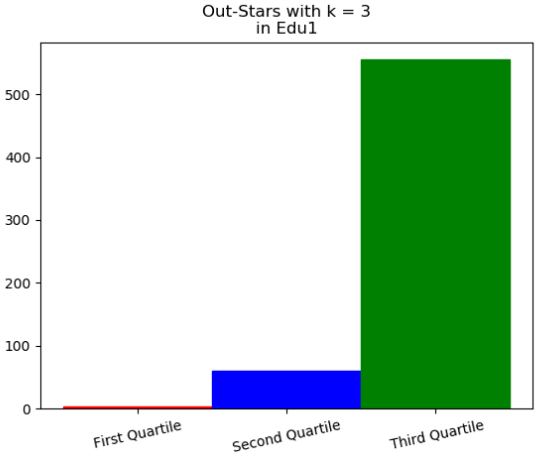
Figure A.3: The plots of the results obtained for all groups modeled with the User Graph with the time window sliding approach

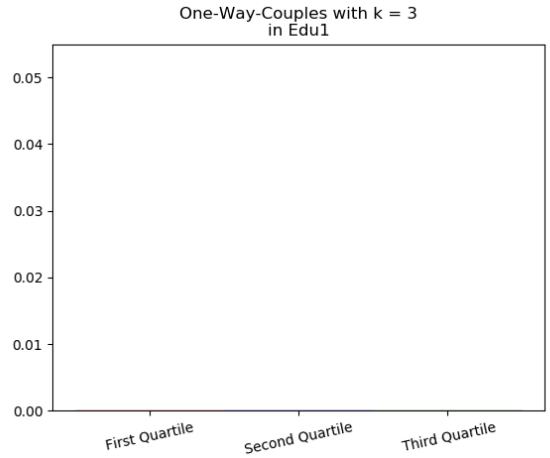
Finally, **Figure A.4** contains the group-based plots relating to the number of motifs we found in the User Graph using the snapshots approach.

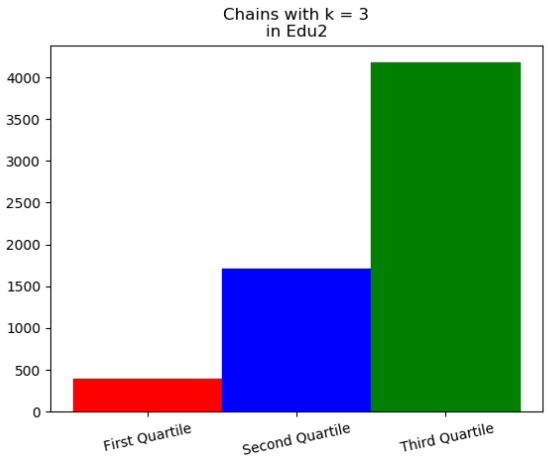

Chains with k = 3
in Edu1
8000
7000
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

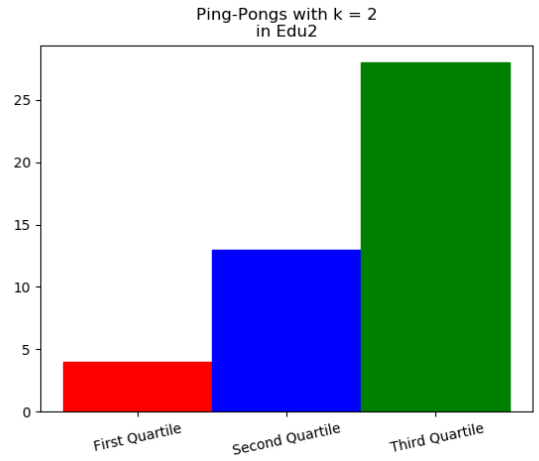

Ping-Pongs with k = 2
in Edu1
70
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

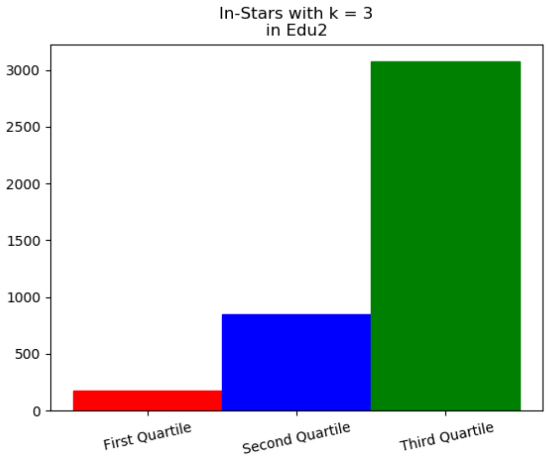

In-Stars with k = 3
in Edu1
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

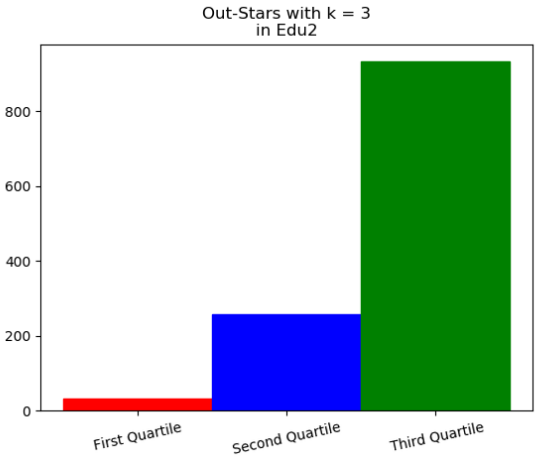

Out-Stars with k = 3
in Edu1
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

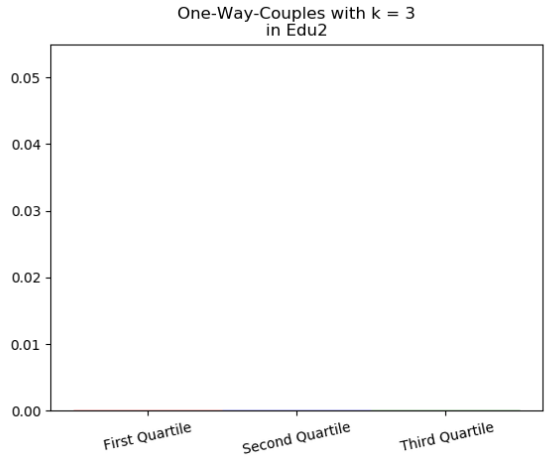

One-Way-Couples with k = 3
in Edu1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

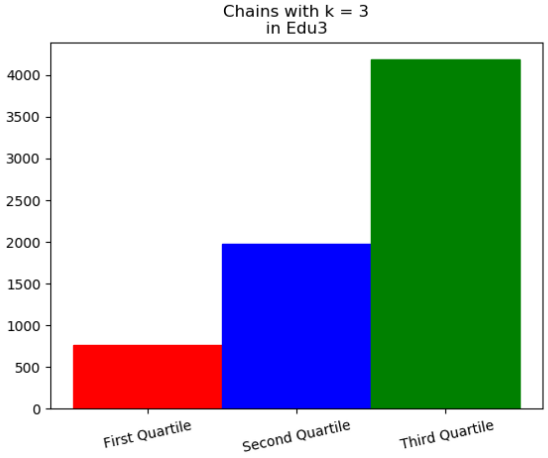

Chains with k = 3
in Edu2
4000
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

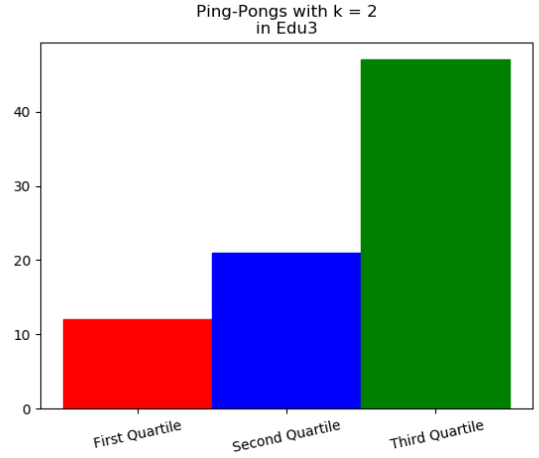

Ping-Pongs with k = 2
in Edu2
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

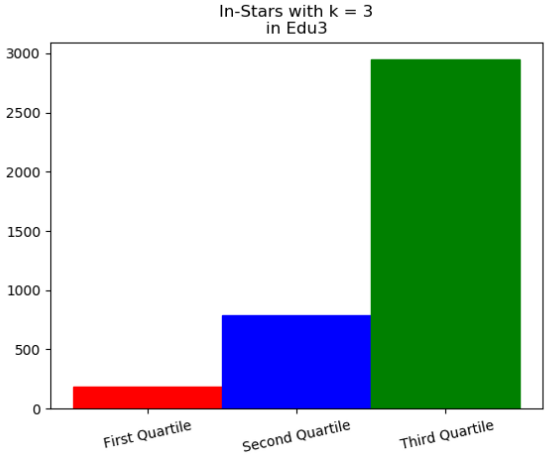

In-Stars with k = 3
in Edu2
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

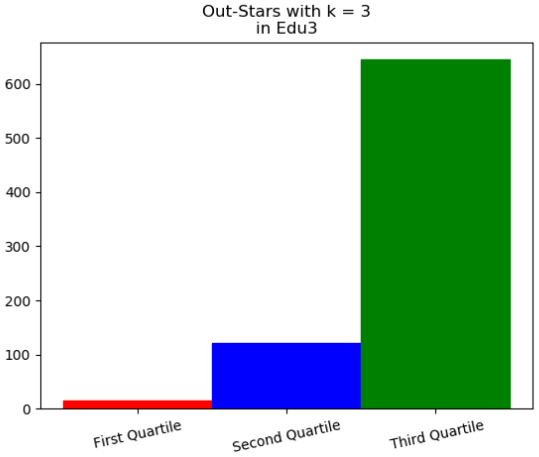

Out-Stars with k = 3
in Edu2
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

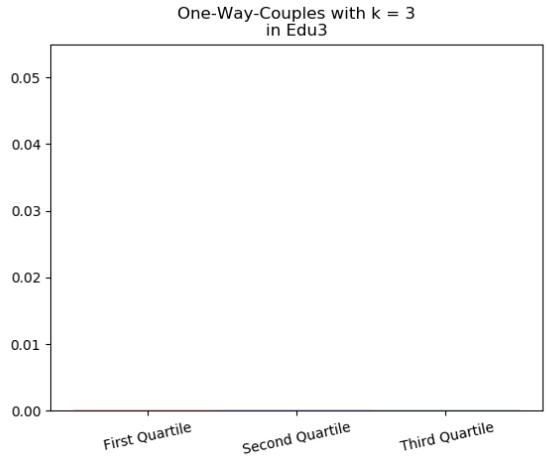

One-Way-Couples with k = 3
in Edu2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

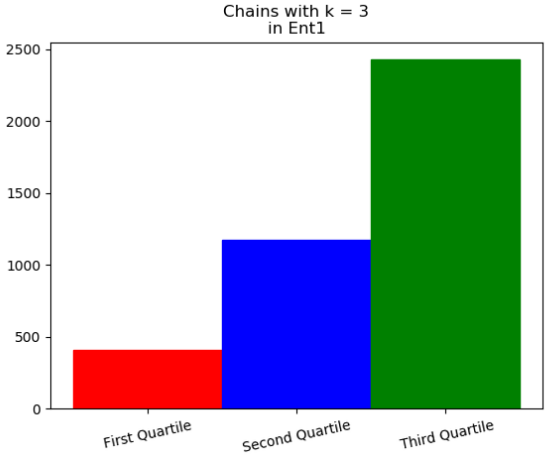

Chains with k = 3
in Edu3
4000
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

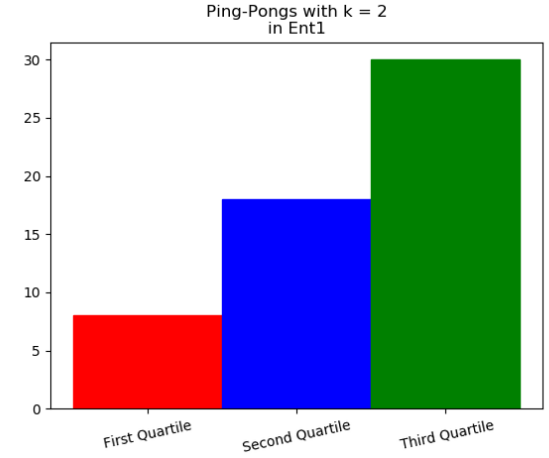

Ping-Pongs with k = 2
in Edu3
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

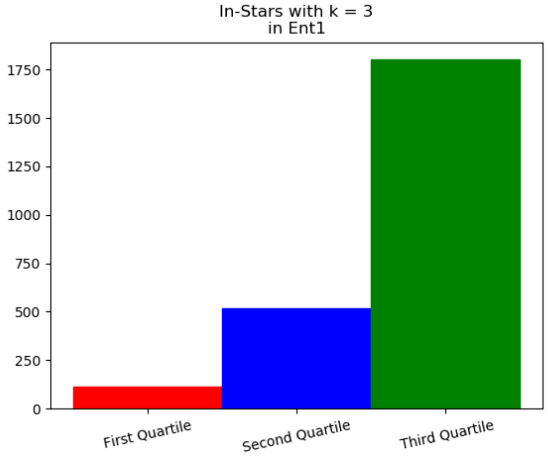

In-Stars with k = 3
in Edu3
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

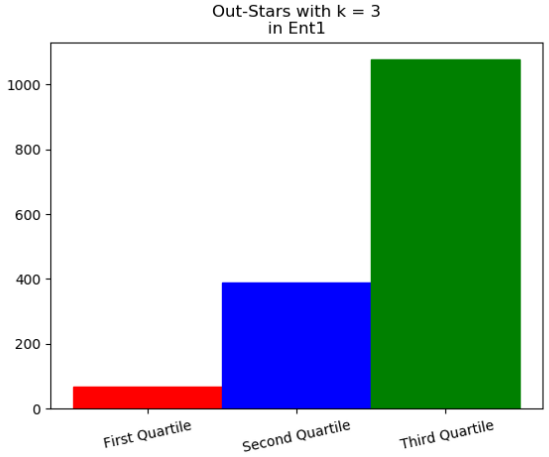

Out-Stars with k = 3
in Edu3
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

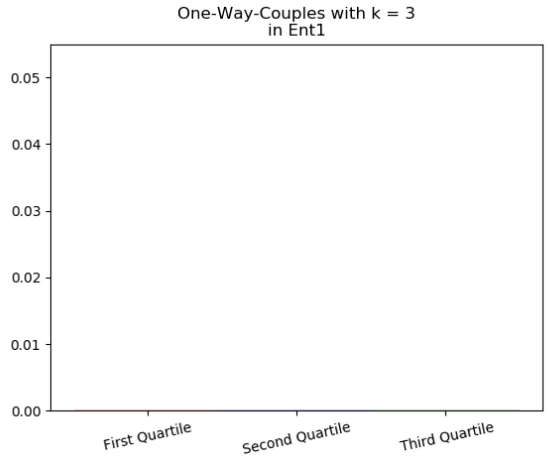

One-Way-Couples with k = 3
in Edu3
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

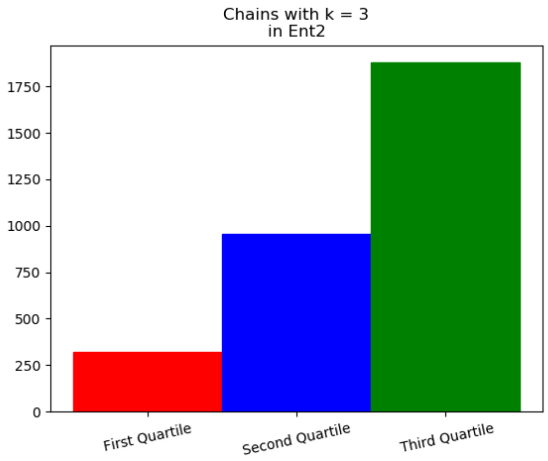

Chains with k = 3
in Ent1
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

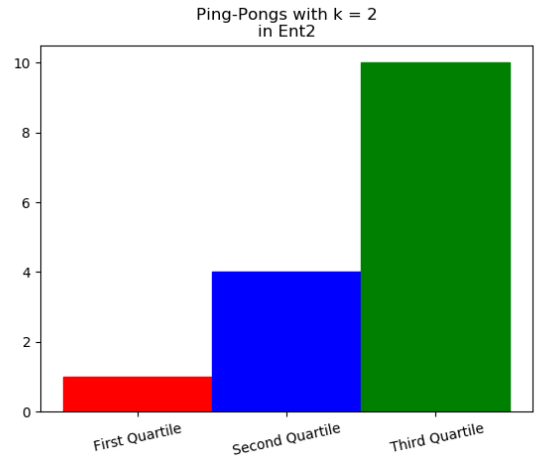

Ping-Pongs with k = 2
in Ent1
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

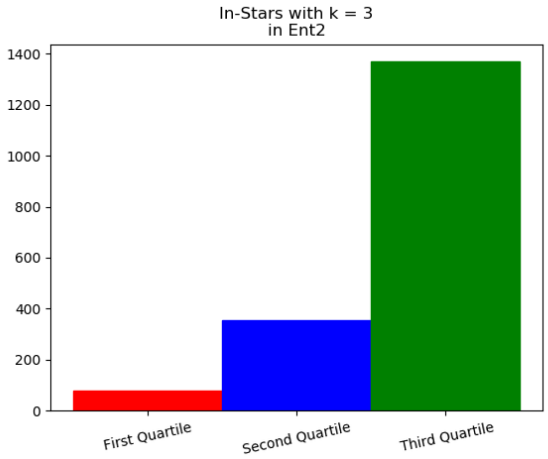

In-Stars with k = 3
in Ent1
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

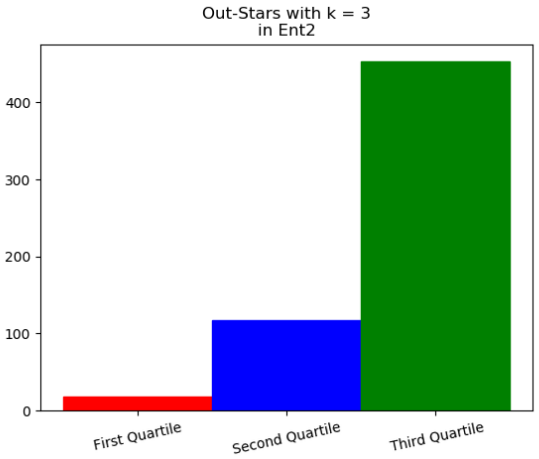

Out-Stars with k = 3
in Ent1
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

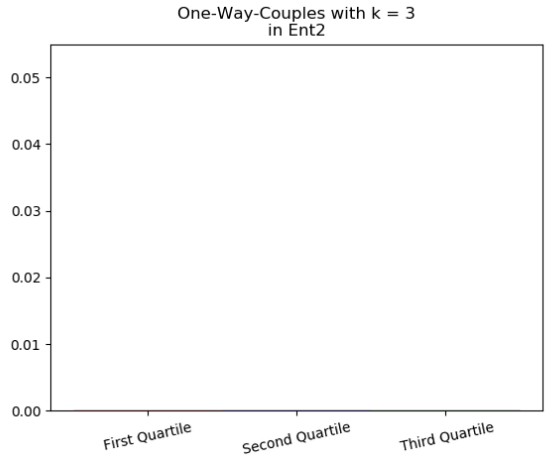

One-Way-Couples with k = 3
in Ent1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

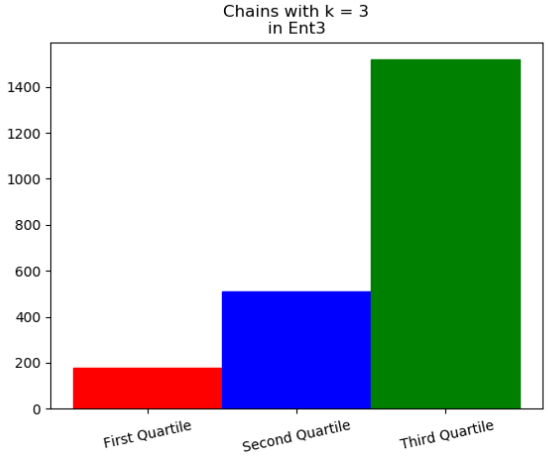

Chains with k = 3
in Ent2
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

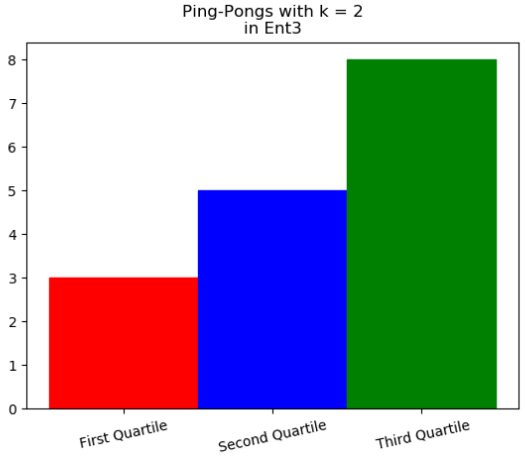

Ping-Pongs with k = 2
in Ent2
10
8
6
4
2
0
First Quartile
Second Quartile
Third Quartile

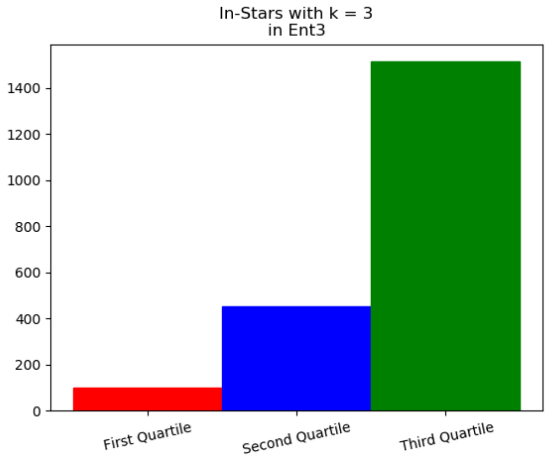

In-Stars with k = 3
in Ent2
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

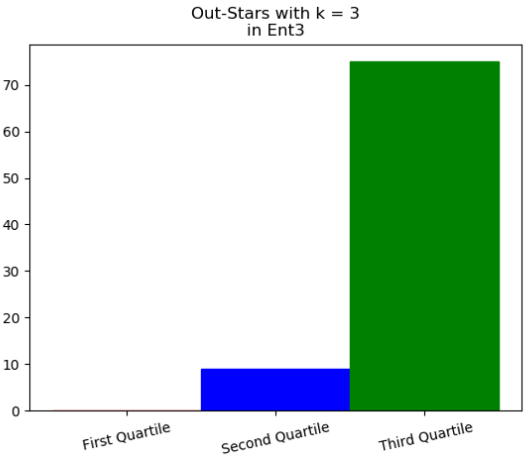

Out-Stars with k = 3
in Ent2
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

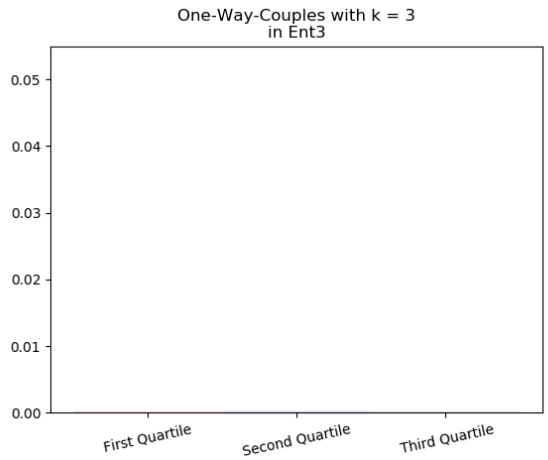

One-Way-Couples with k = 3
in Ent2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

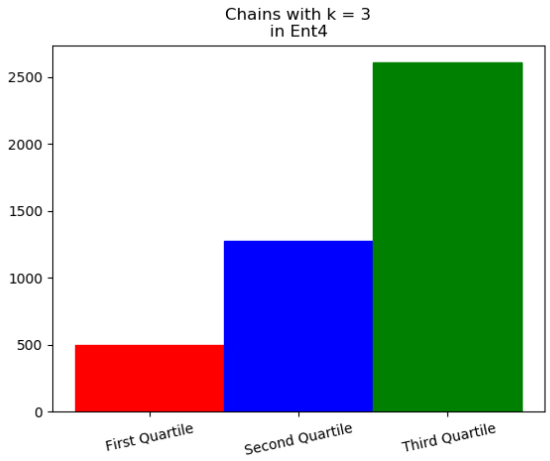

Chains with k = 3
in Ent3
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

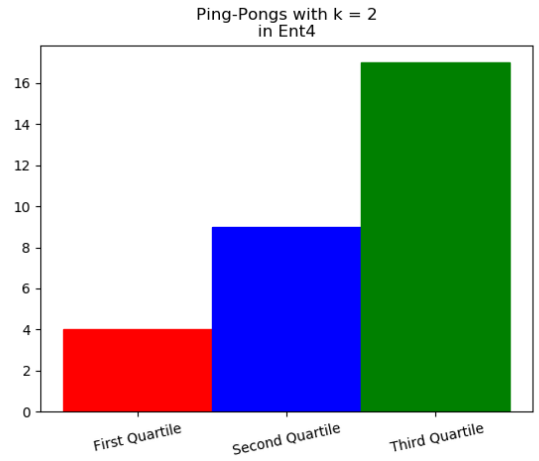

Ping-Pongs with k = 2
in Ent3
8
7
6
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

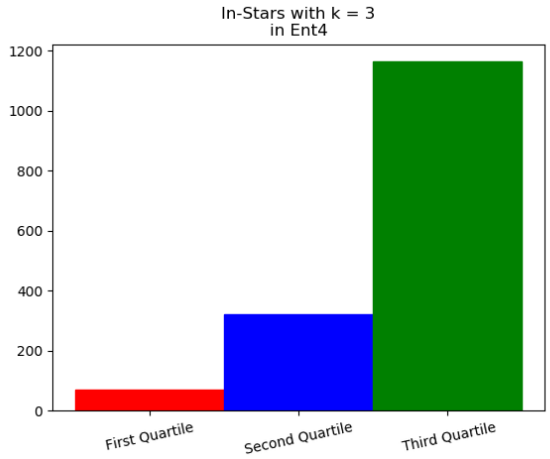

In-Stars with k = 3
in Ent3
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

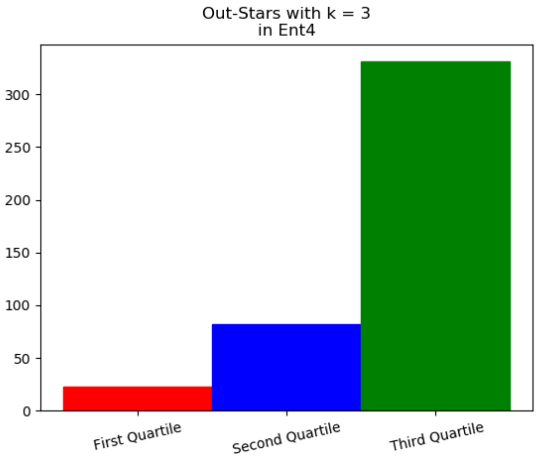

Out-Stars with k = 3
in Ent3
70
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

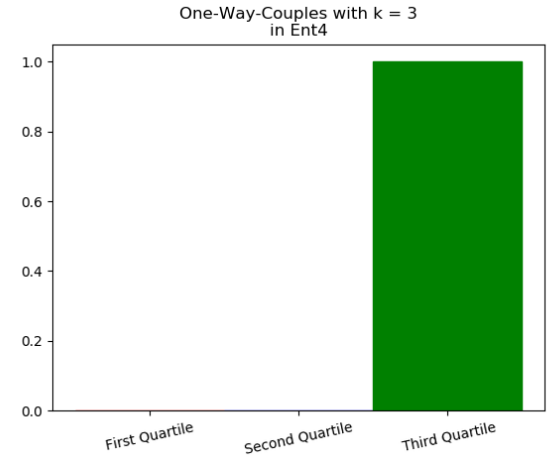

One-Way-Couples with k = 3
in Ent3
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

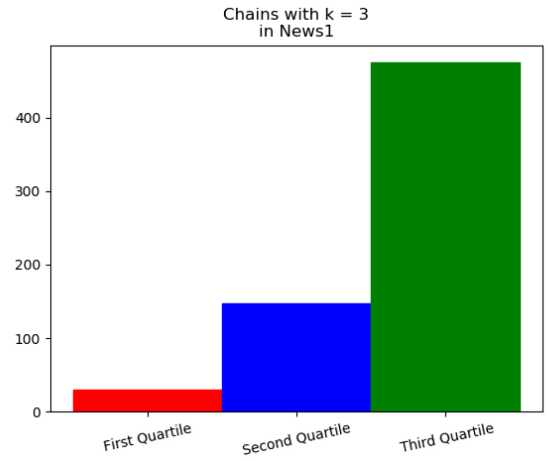

Chains with k = 3
in Ent4
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

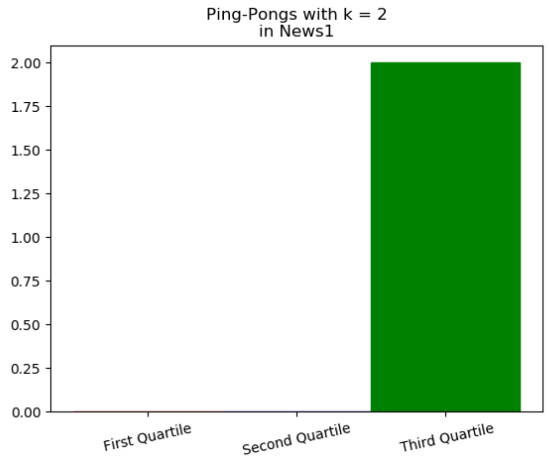

Ping-Pongs with k = 2
in Ent4
16
14
12
10
8
6
4
2
0
First Quartile
Second Quartile
Third Quartile

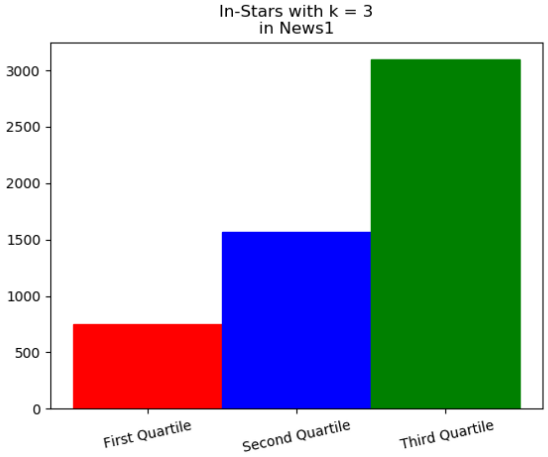

In-Stars with k = 3
in Ent4
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

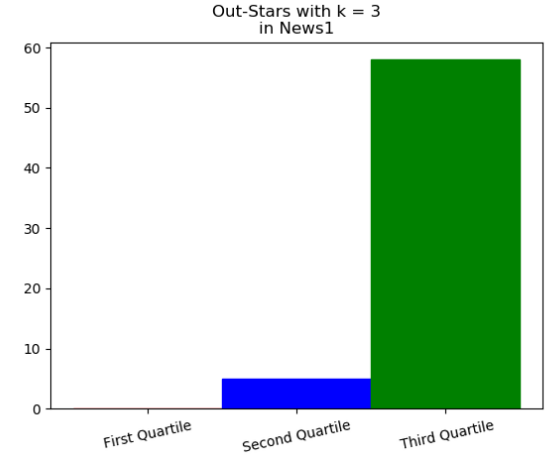

Out-Stars with k = 3
in Ent4
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

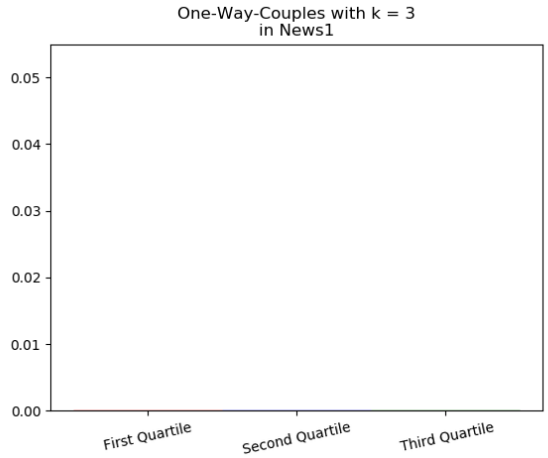

One-Way-Couples with k = 3
in Ent4
1.0
0.8
0.6
0.4
0.2
0.0
First Quartile
Second Quartile
Third Quartile

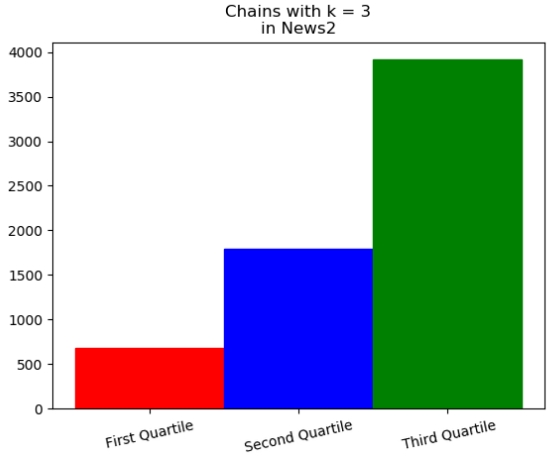

Chains with k = 3
in News1
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

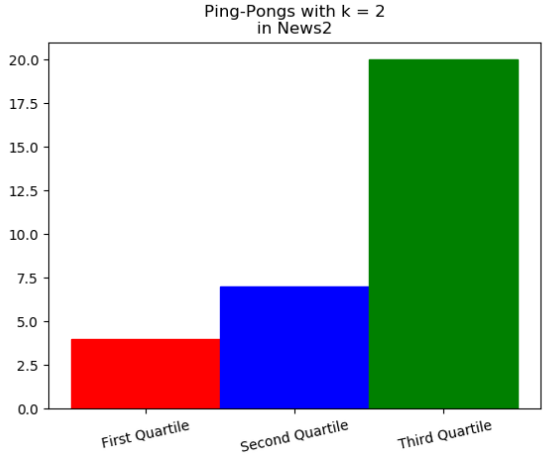

Ping-Pongs with k = 2
in News1
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
First Quartile
Second Quartile
Third Quartile

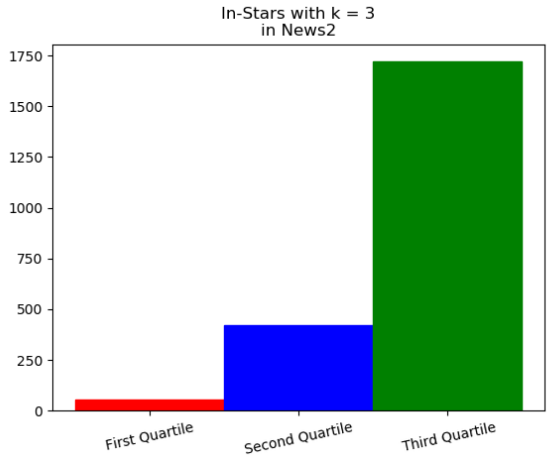

In-Stars with k = 3
in News1
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

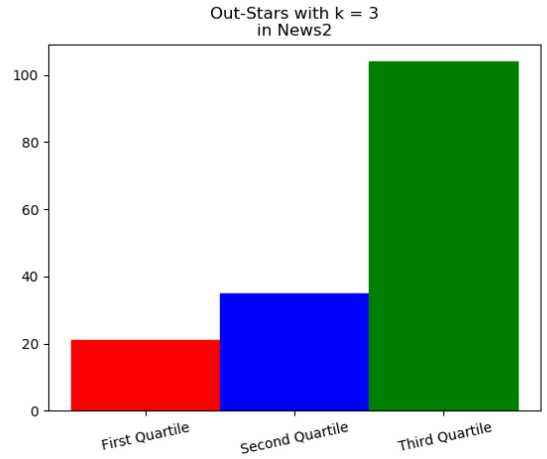

Out-Stars with k = 3
in News1
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

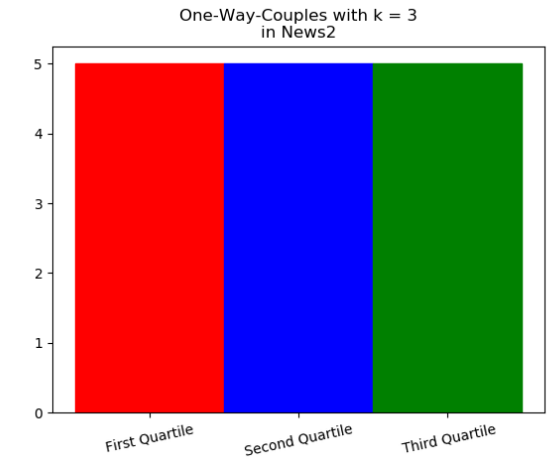

One-Way-Couples with k = 3
in News1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

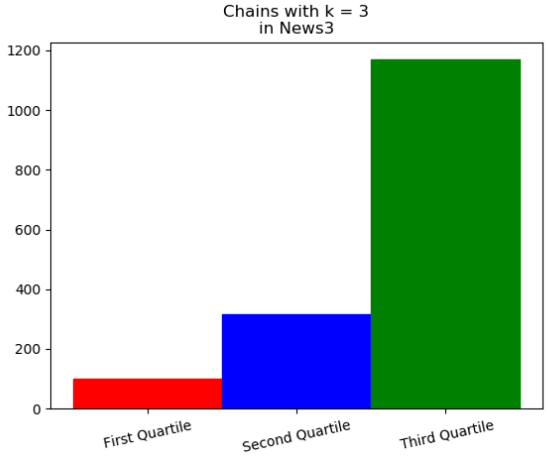

Chains with k = 3
in News2
4000
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

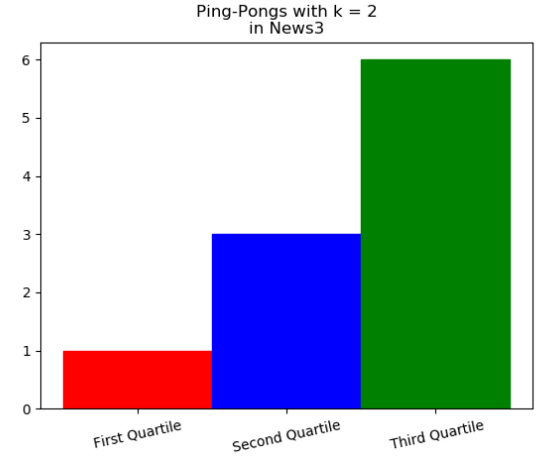

Ping-Pongs with k = 2
in News2
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
First Quartile
Second Quartile
Third Quartile

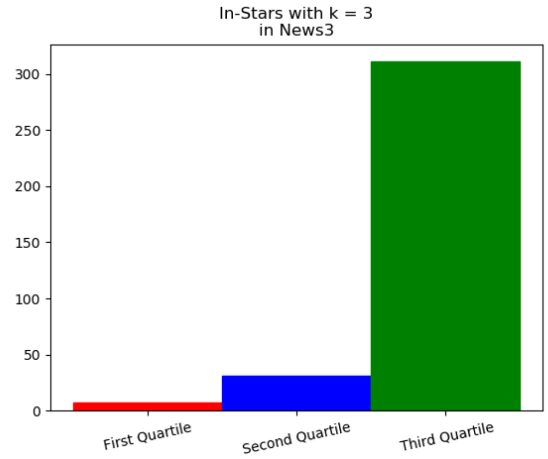

In-Stars with k = 3
in News2
1750
1500
1250
1000
750
500
250
0
First Quartile
Second Quartile
Third Quartile

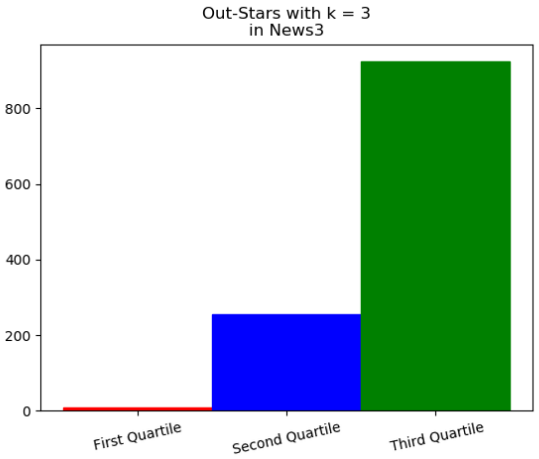

Out-Stars with k = 3
in News2
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

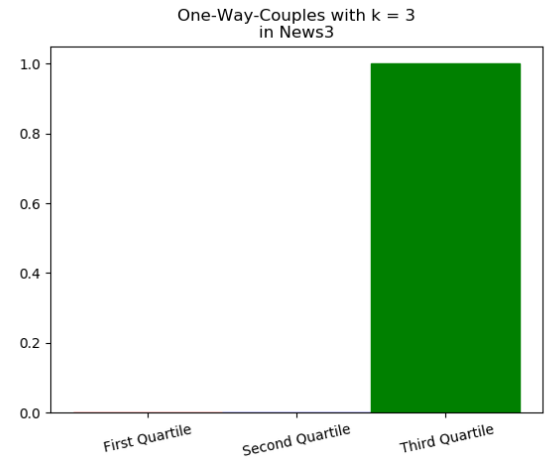

One-Way-Couples with k = 3
in News2
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

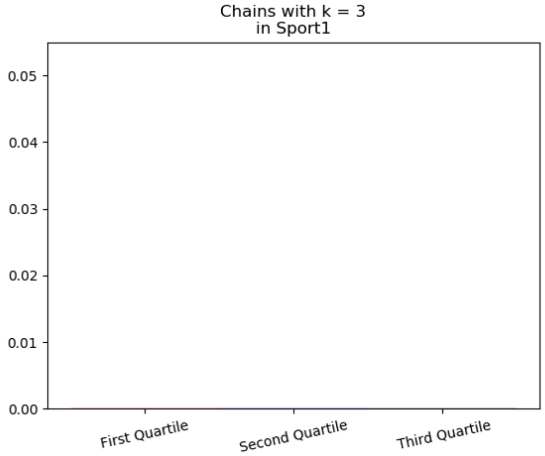

Chains with k = 3
in News3
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

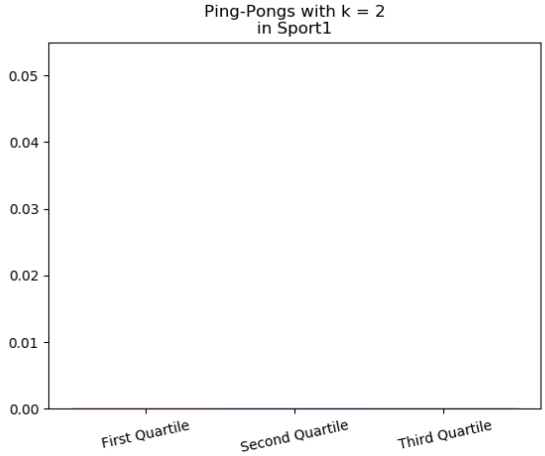

Ping-Pongs with k = 2
in News3
6
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

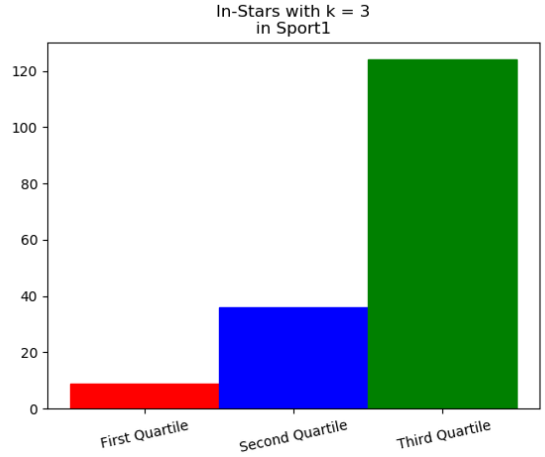

In-Stars with k = 3
in News3
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

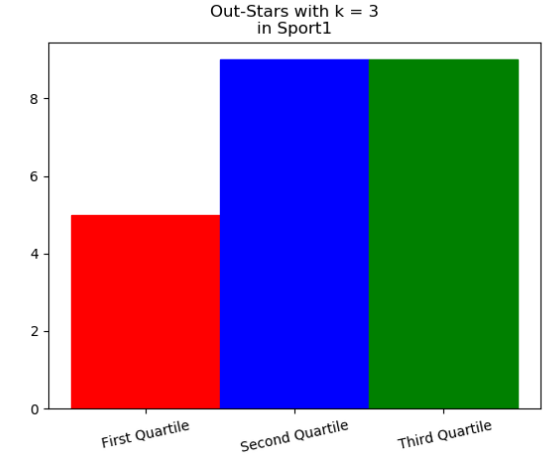

Out-Stars with k = 3
in News3
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

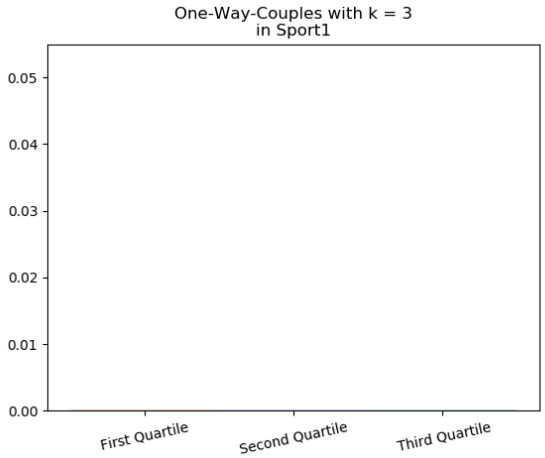

One-Way-Couples with k = 3
in News3
1.0
0.8
0.6
0.4
0.2
0.0
First Quartile
Second Quartile
Third Quartile

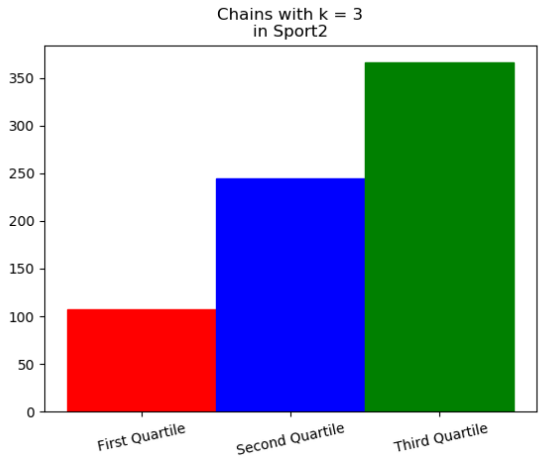

Chains with k = 3
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

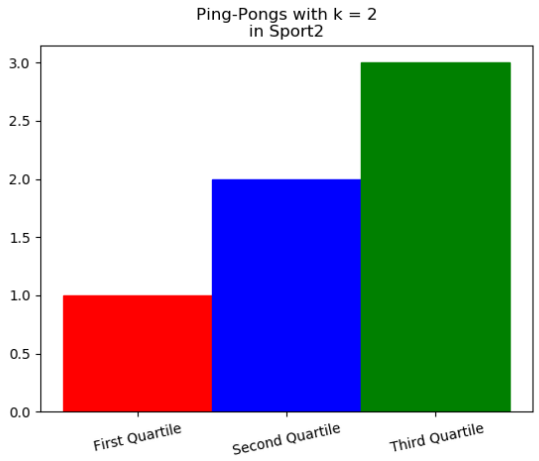

Ping-Pongs with k = 2
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

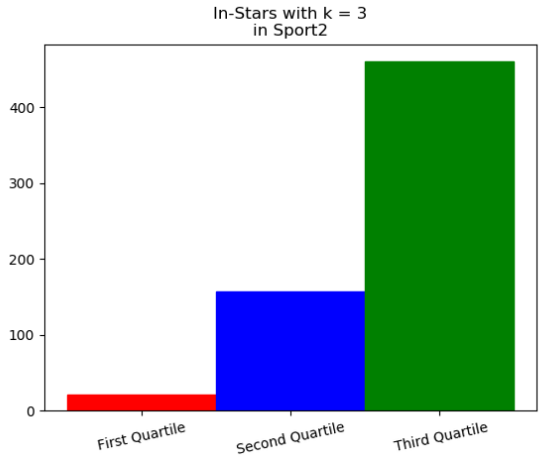

In-Stars with k = 3
in Sport1
120
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile

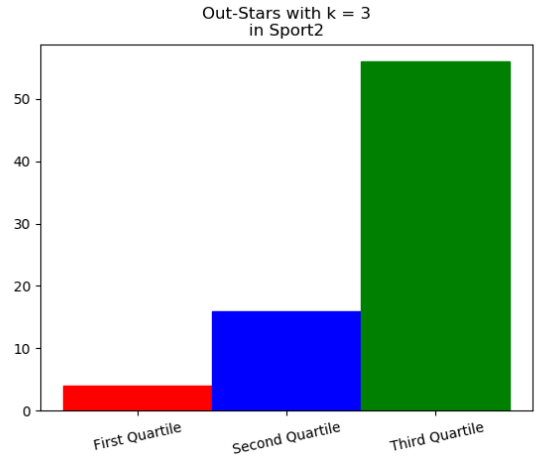

Out-Stars with k = 3
in Sport1
8
6
4
2
0
First Quartile
Second Quartile
Third Quartile

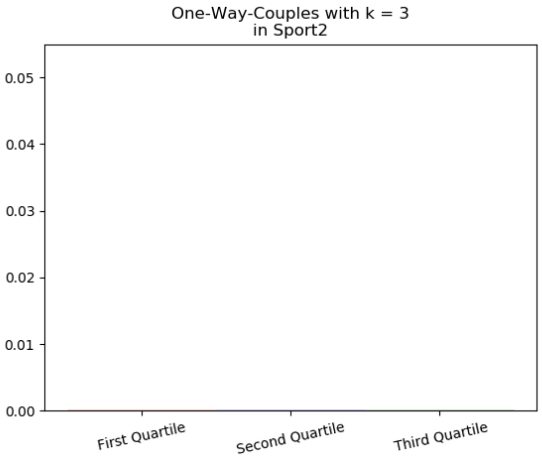

One-Way-Couples with k = 3
in Sport1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

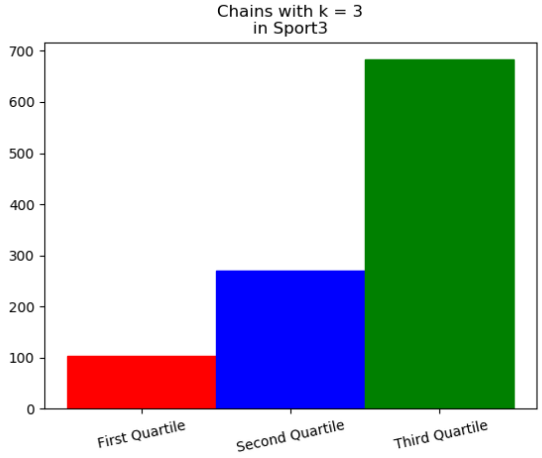

Chains with k = 3
in Sport2
350
300
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile

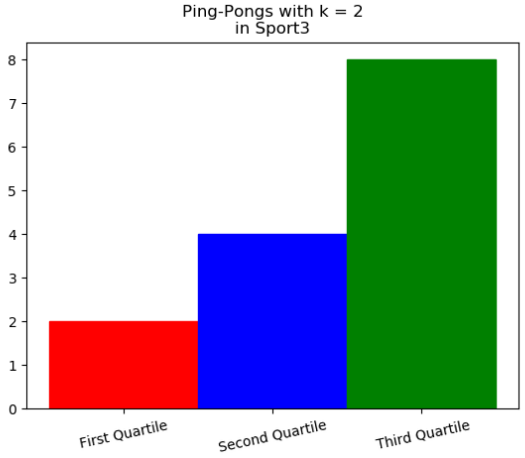

Ping-Pongs with k = 2
in Sport2
3.0
2.5
2.0
1.5
1.0
0.5
0.0
First Quartile
Second Quartile
Third Quartile

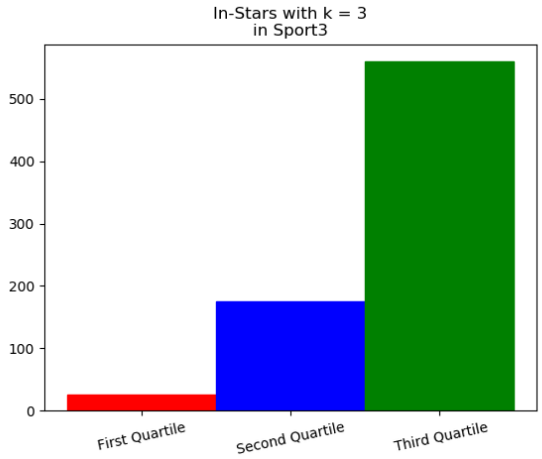

In-Stars with k = 3
in Sport2
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

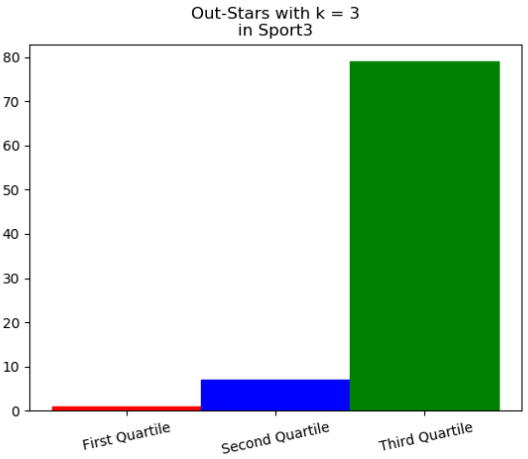

Out-Stars with k = 3
in Sport2
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

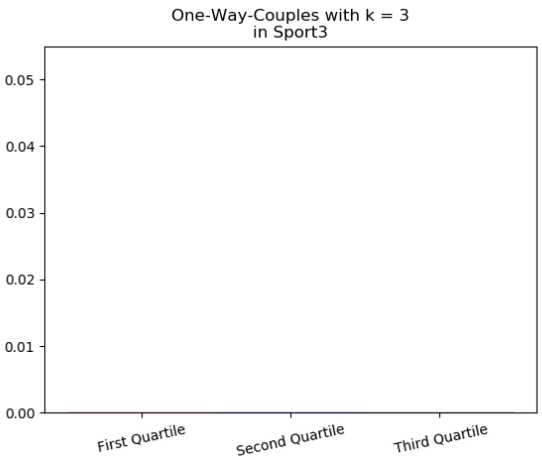

One-Way-Couples with k = 3
in Sport2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

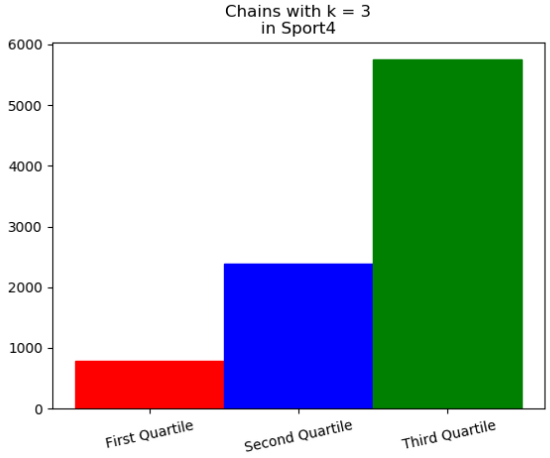

Chains with k = 3
in Sport3
700
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

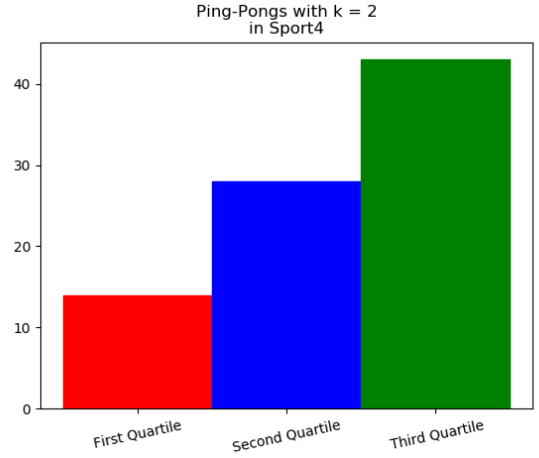

Ping-Pongs with k = 2
in Sport3
8
7
6
5
4
3
2
1
0
First Quartile
Second Quartile
Third Quartile

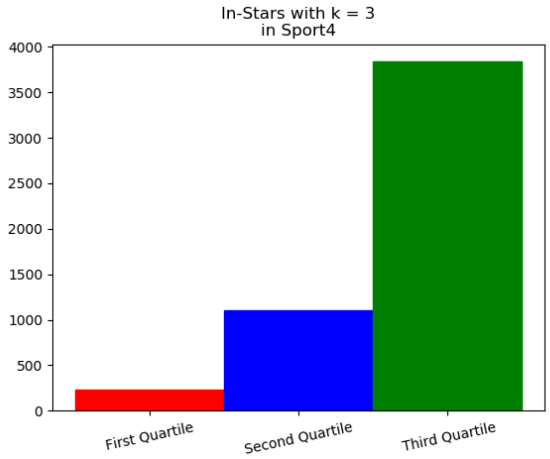

In-Stars with k = 3
in Sport3
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

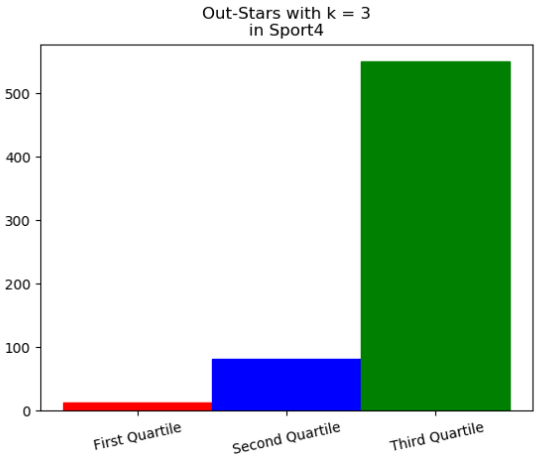

Out-Stars with k = 3
in Sport3
80
70
60
50
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

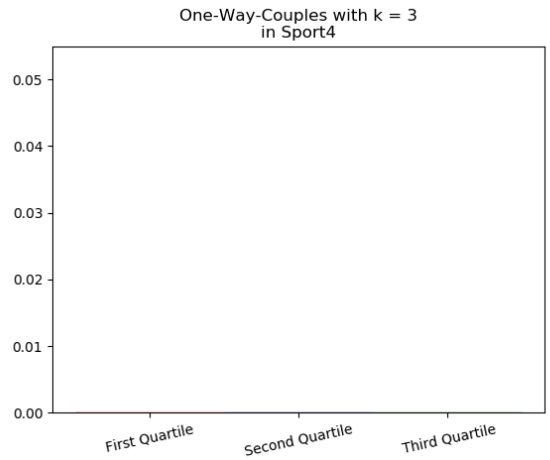

One-Way-Couples with k = 3
in Sport3
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

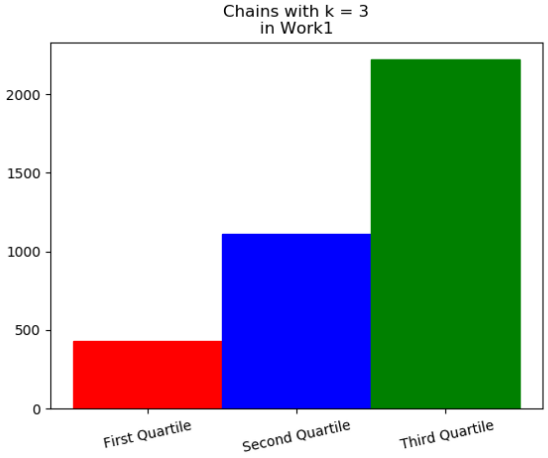

Chains with k = 3
in Sport4
6000
5000
4000
3000
2000
1000
0
First Quartile
Second Quartile
Third Quartile

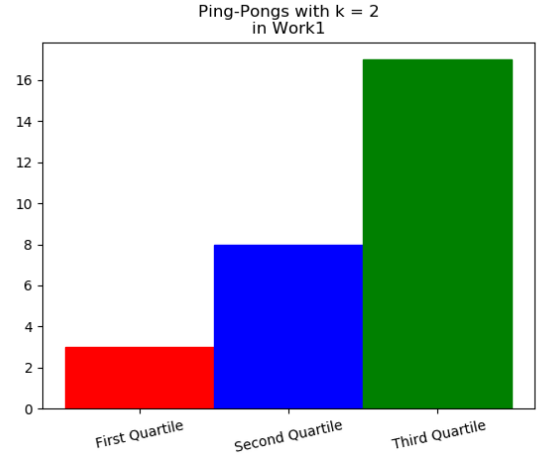

Ping-Pongs with k = 2
in Sport4
40
30
20
10
0
First Quartile
Second Quartile
Third Quartile

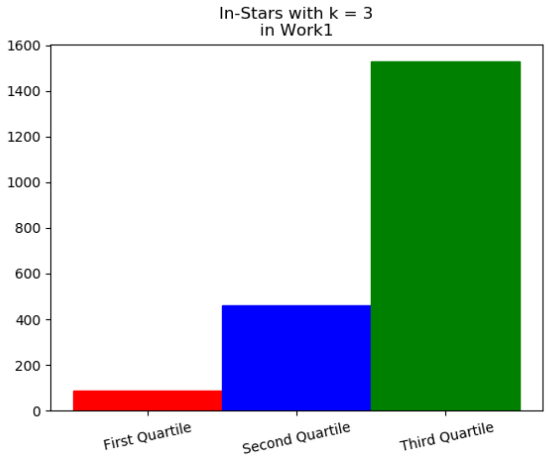

In-Stars with k = 3
in Sport4
4000
3500
3000
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

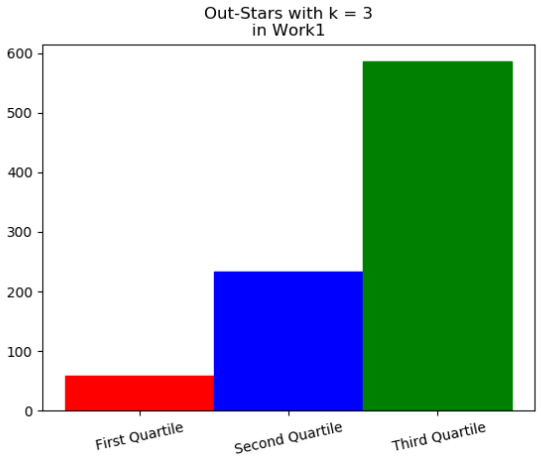

Out-Stars with k = 3
in Sport4
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

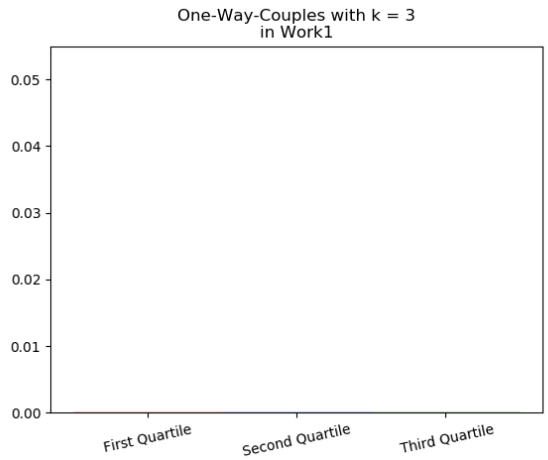

One-Way-Couples with k = 3
in Sport4
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

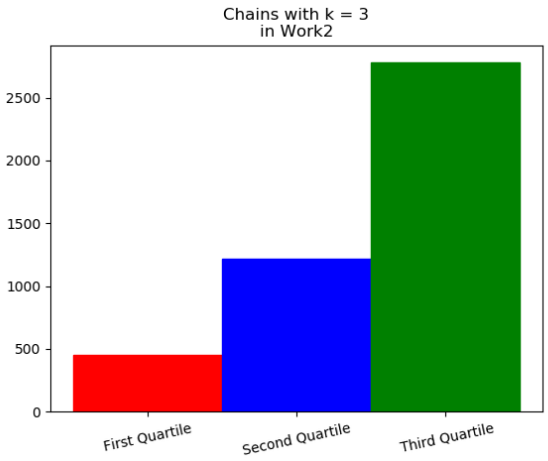

Chains with k = 3
in Work1
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

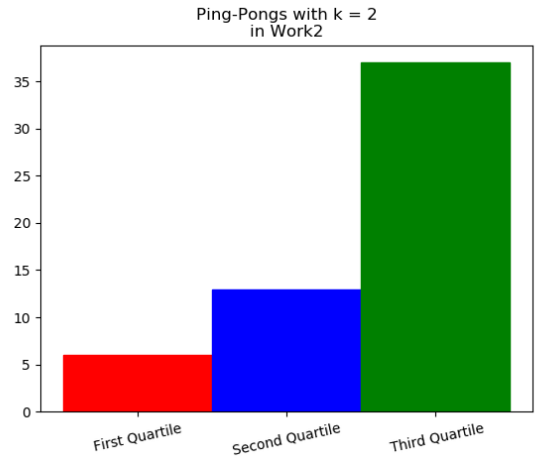

Ping-Pongs with k = 2
in Work1
16
14
12
10
8
6
4
2
0
First Quartile
Second Quartile
Third Quartile

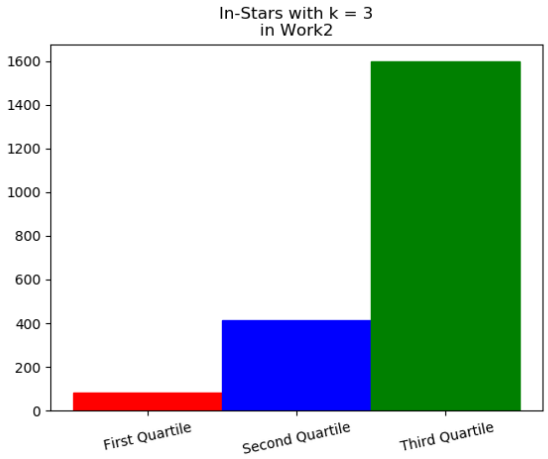

In-Stars with k = 3
in Work1
1600
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

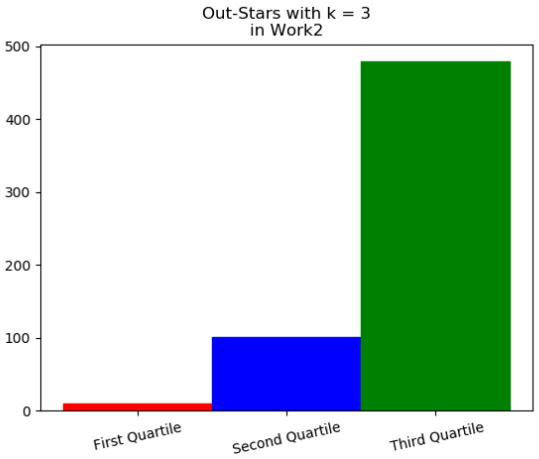

Out-Stars with k = 3
in Work1
600
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

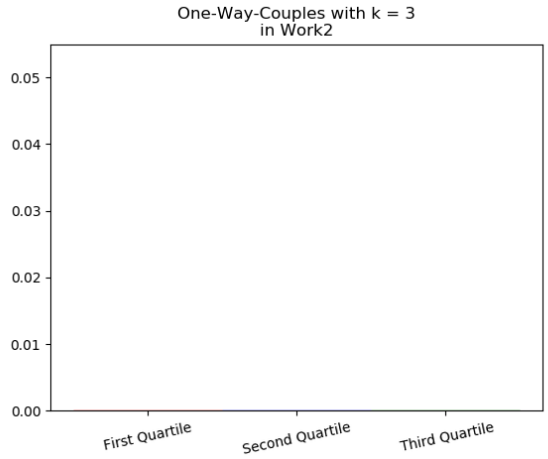

One-Way-Couples with k = 3
in Work1
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

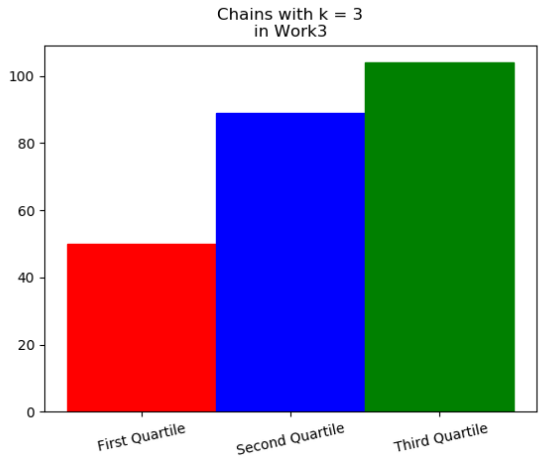

Chains with k = 3
in Work2
2500
2000
1500
1000
500
0
First Quartile
Second Quartile
Third Quartile

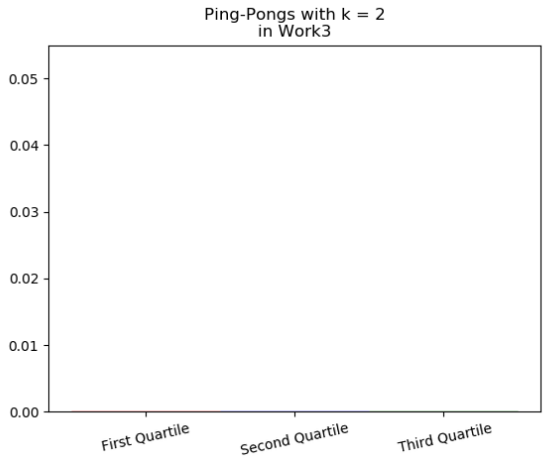

Ping-Pongs with k = 2
in Work2
35
30
25
20
15
10
5
0
First Quartile
Second Quartile
Third Quartile

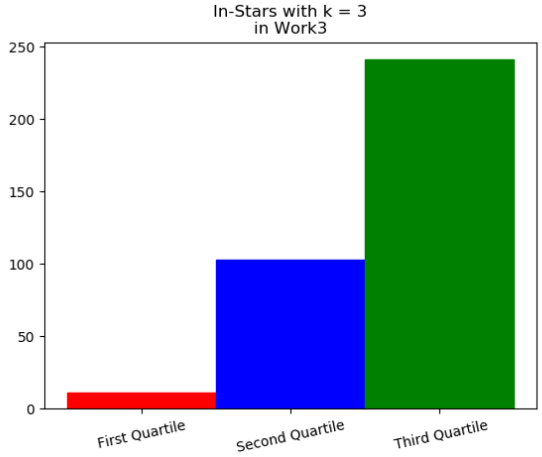

In-Stars with k = 3
in Work2
1600
1400
1200
1000
800
600
400
200
0
First Quartile
Second Quartile
Third Quartile

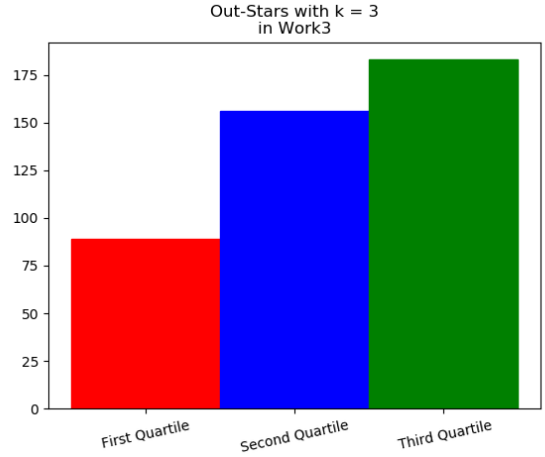

Out-Stars with k = 3
in Work2
500
400
300
200
100
0
First Quartile
Second Quartile
Third Quartile

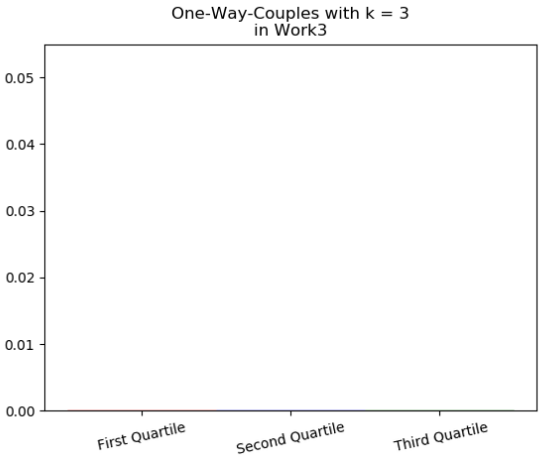

One-Way-Couples with k = 3
in Work2
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile


Chains with k = 3
in Work3
100
80
60
40
20
0
First Quartile
Second Quartile
Third Quartile


Ping-Pongs with k = 2
in Work3
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile


In-Stars with k = 3
in Work3
250
200
150
100
50
0
First Quartile
Second Quartile
Third Quartile


Out-Stars with k = 3
in Work3
175
150
125
100
75
50
25
0
First Quartile
Second Quartile
Third Quartile


One-Way-Couples with k = 3
in Work3
0.05
0.04
0.03
0.02
0.01
0.00
First Quartile
Second Quartile
Third Quartile

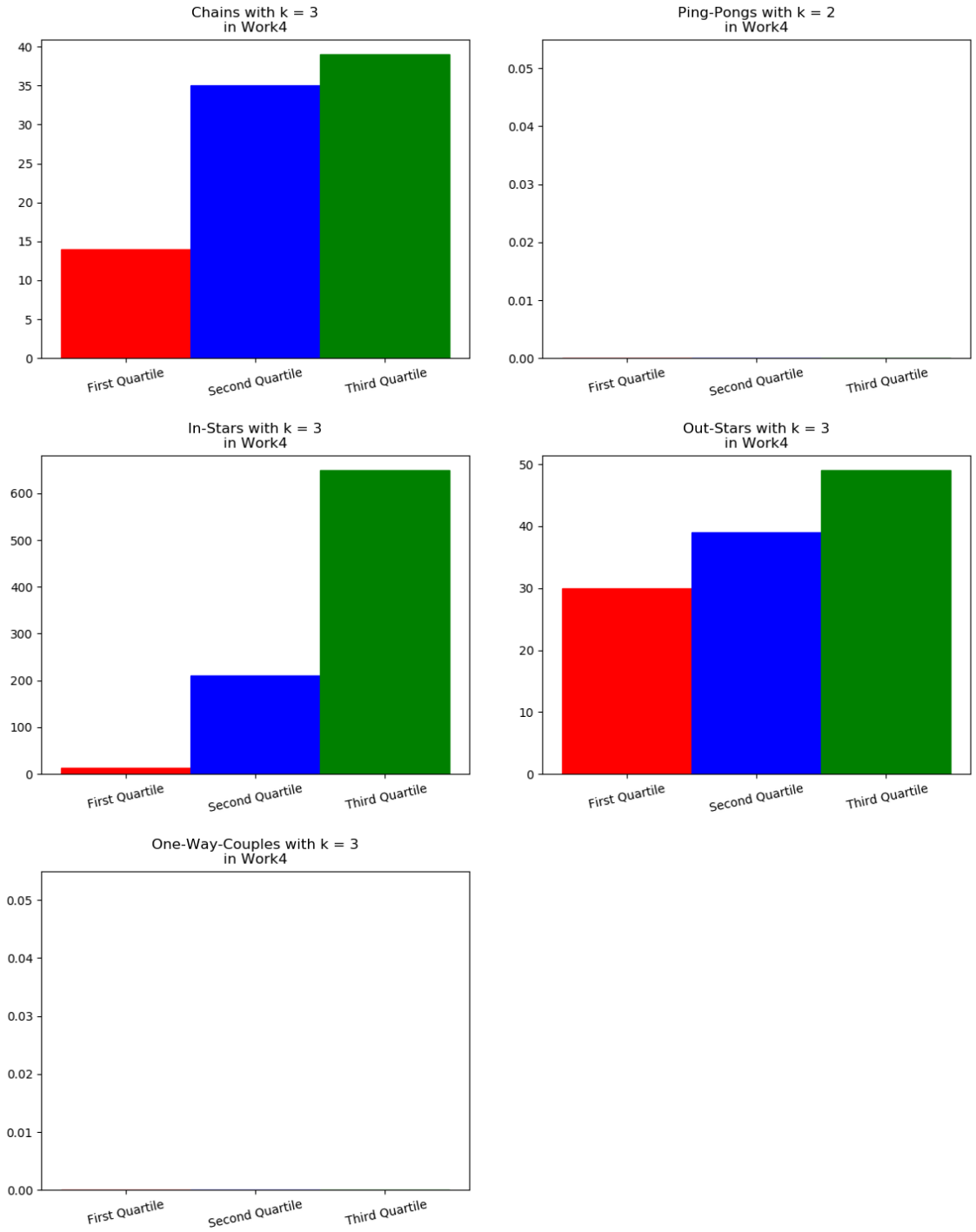
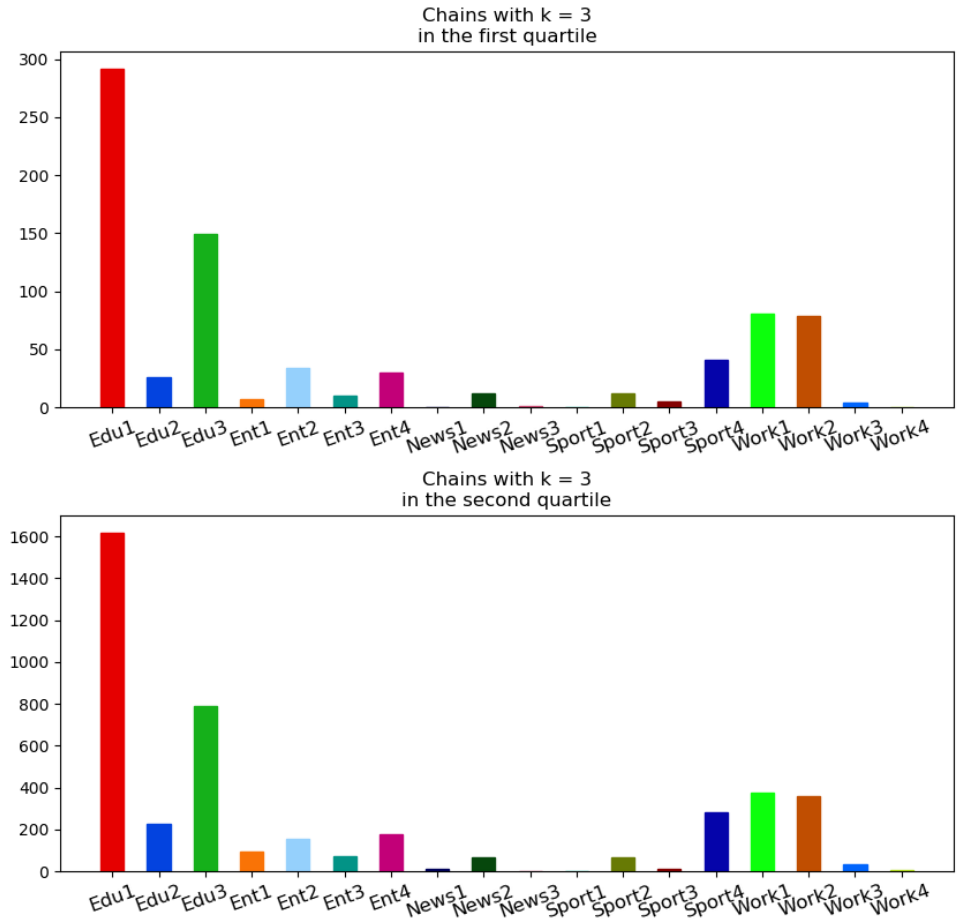


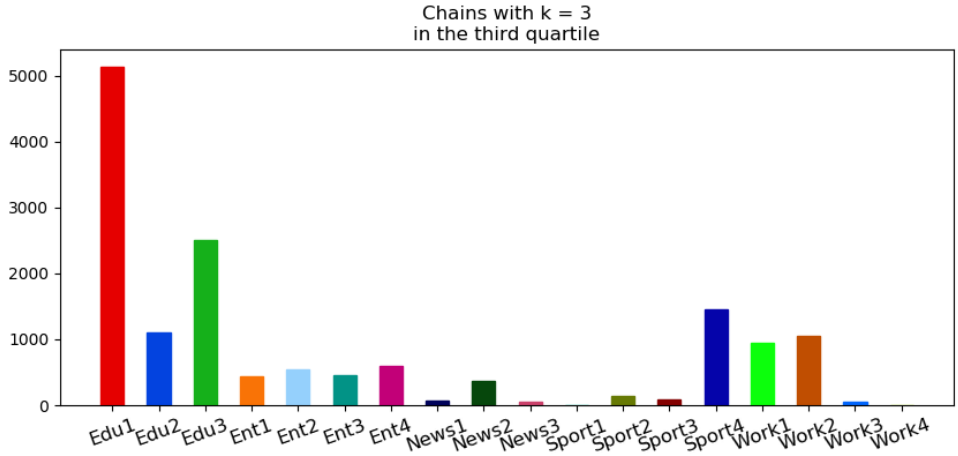
Figure A.4: The plots of the results obtained for all groups modeled with the User Graph with the snapshots approach

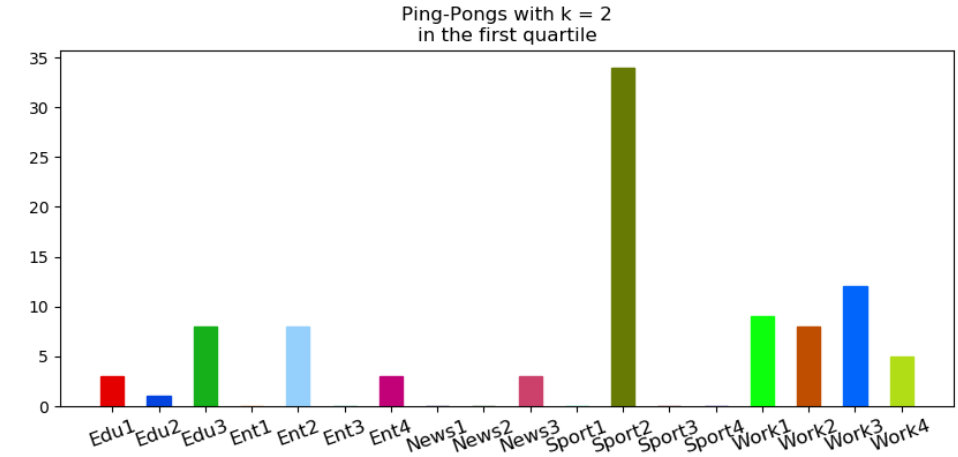
A.2 Quartile-based Diagrams

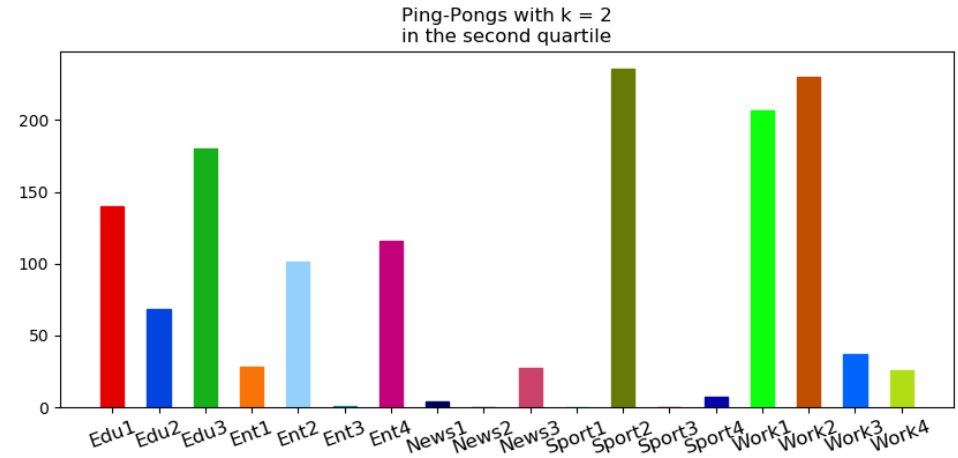
In this section we present the quartile-based plots, that we obtained with all the four possible combinations between the two graphs formulations and the two time window approaches. Each diagram is related to a particular temporal motif in a specific quartile, and compares.

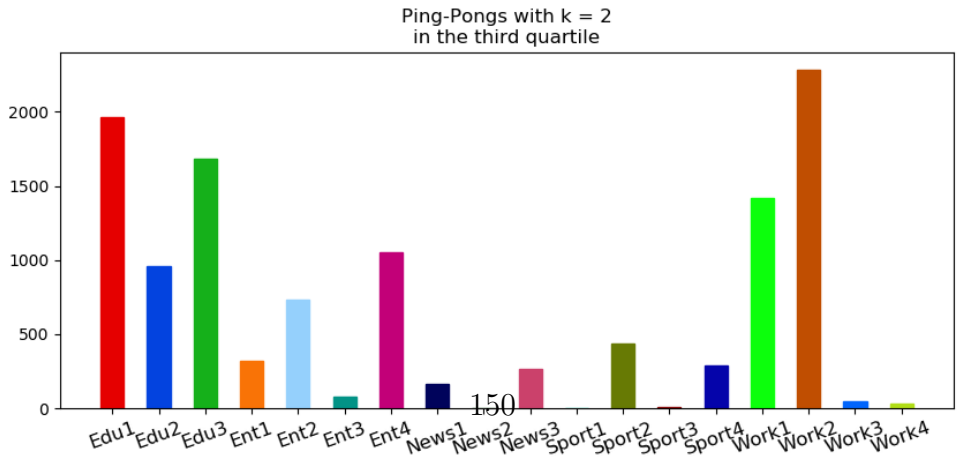
The first set of diagrams, in **Figure A.5**, contains the quartile-based plots relating to the number of motifs we found in the three quartiles in the Post Graph using the time window sliding approach.

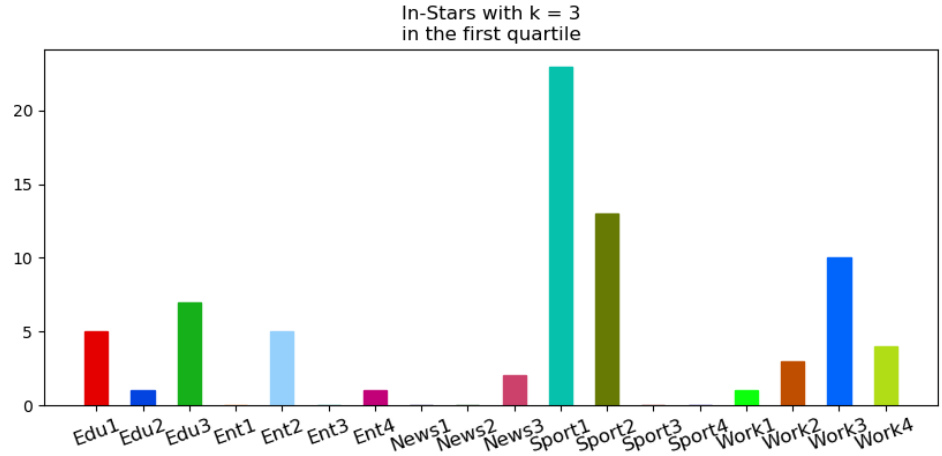


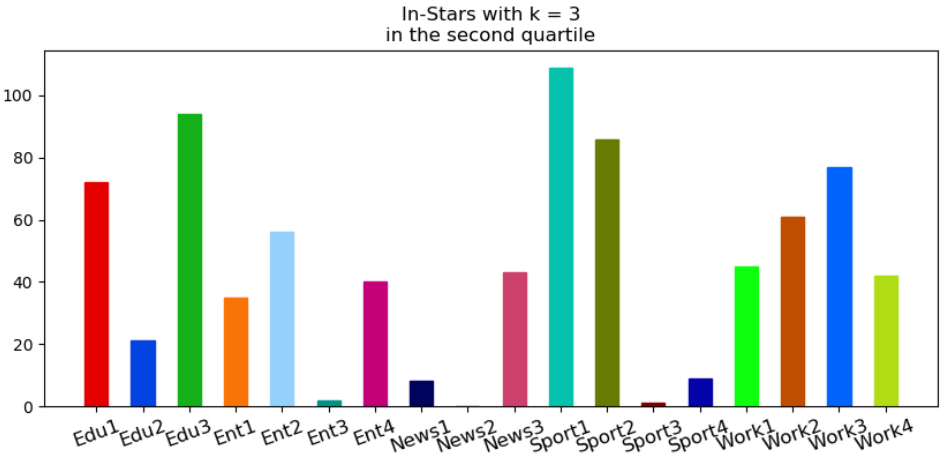

Chains with k = 3
in the third quartile
5000
4000
3000
2000
1000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

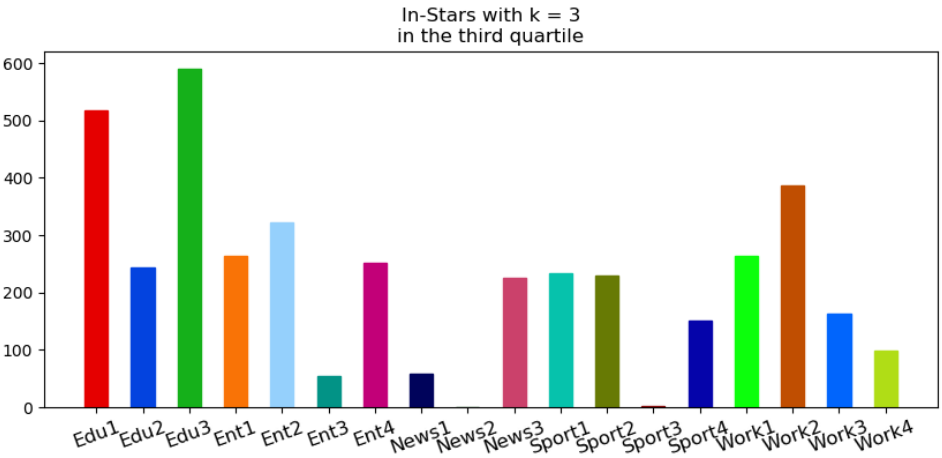

Ping-Pongs with k = 2
in the first quartile
35
30
25
20
15
10
5
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

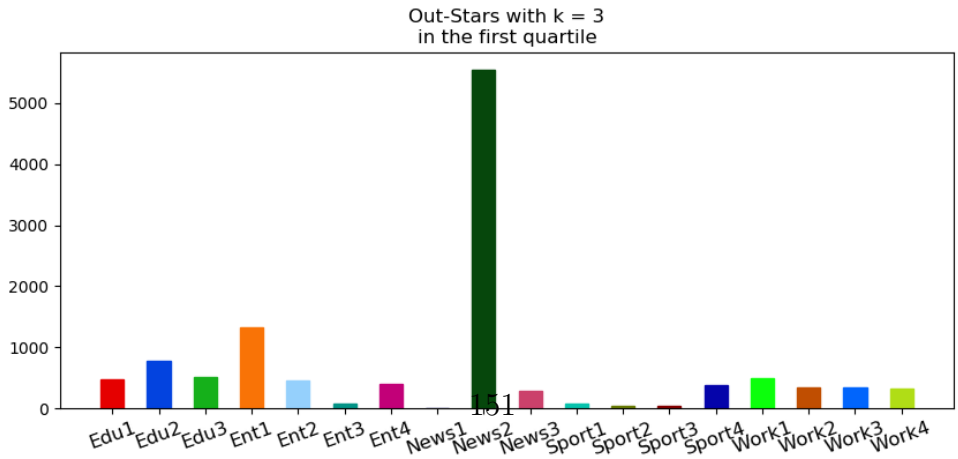

Ping-Pongs with k = 2
in the second quartile
200
150
100
50
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

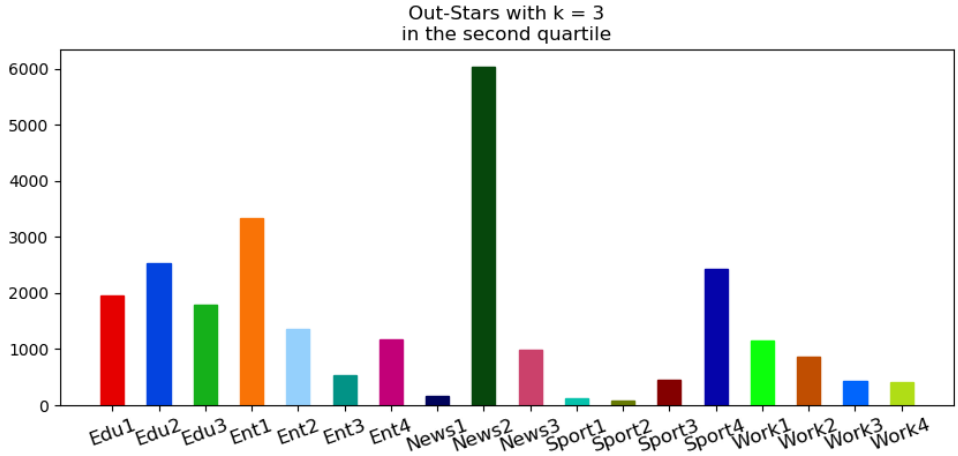

Ping-Pongs with k = 2
in the third quartile
2000
1500
1000
500
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

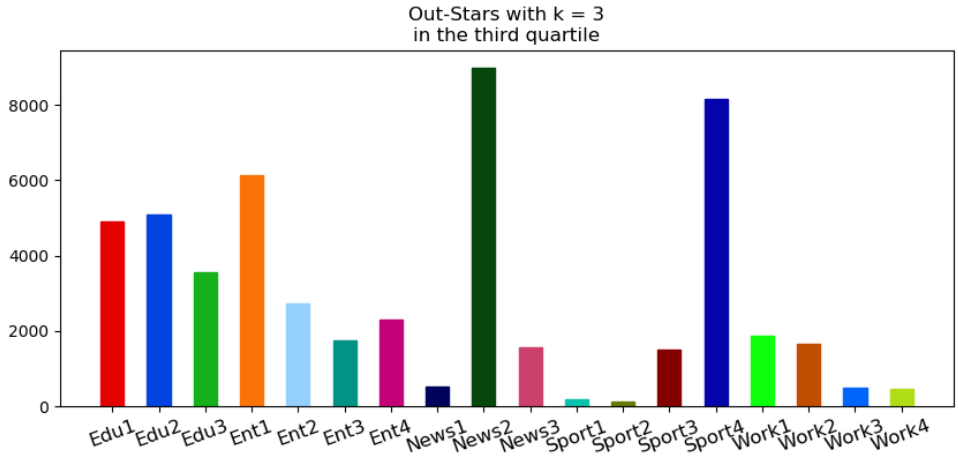

In-Stars with k = 3
in the first quartile
0
5
10
15
20
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


In-Stars with k = 3
in the second quartile
0
20
40
60
80
100
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


In-Stars with k = 3
in the third quartile
0
100
200
300
400
500
600
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

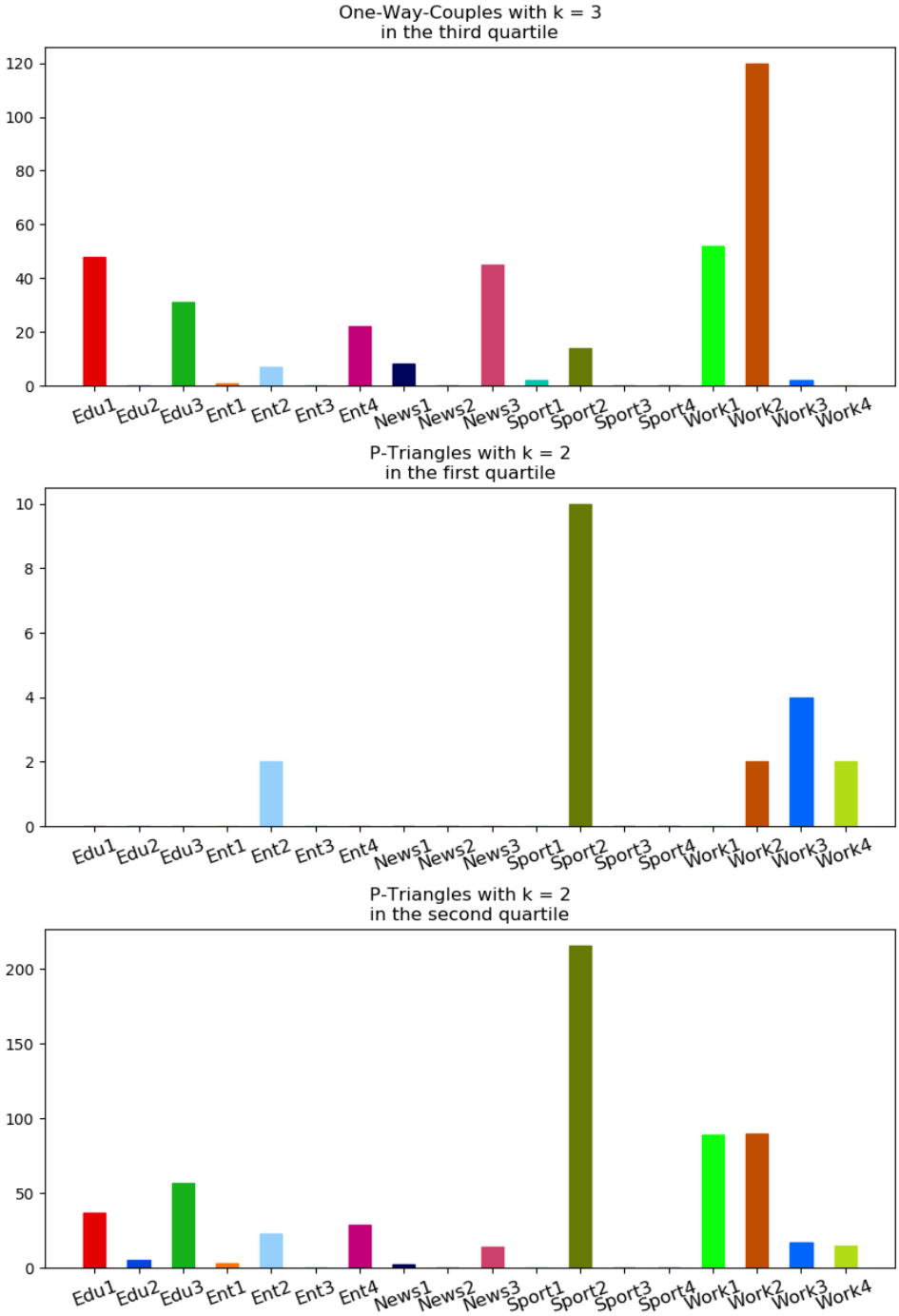

Out-Stars with k = 3
in the first quartile
0
1000
2000
3000
4000
5000
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


Out-Stars with k = 3
in the second quartile
6000
5000
4000
3000
2000
1000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


Out-Stars with k = 3
in the third quartile
8000
6000
4000
2000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


One-Way-Couples with k = 3
in the first quartile
1.0
0.8
0.6
0.4
0.2
0.0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


One-Way-Couples with k = 3
in the second quartile
7
6
5
4
3
2
1
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


One-Way-Couples with k = 3
in the third quartile
120
100
80
60
40
20
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4
P-Triangles with k = 2
in the first quartile
10
8
6
4
2
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4
P-Triangles with k = 2
in the second quartile
200
150
100
50
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

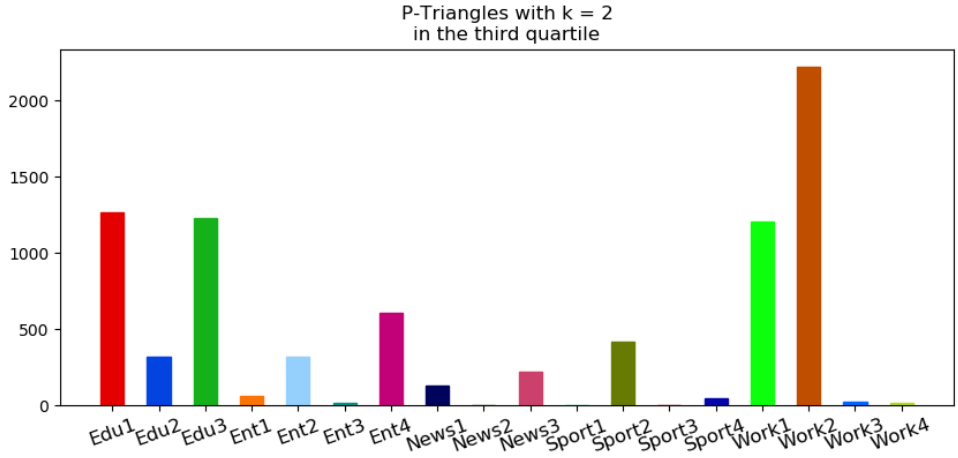
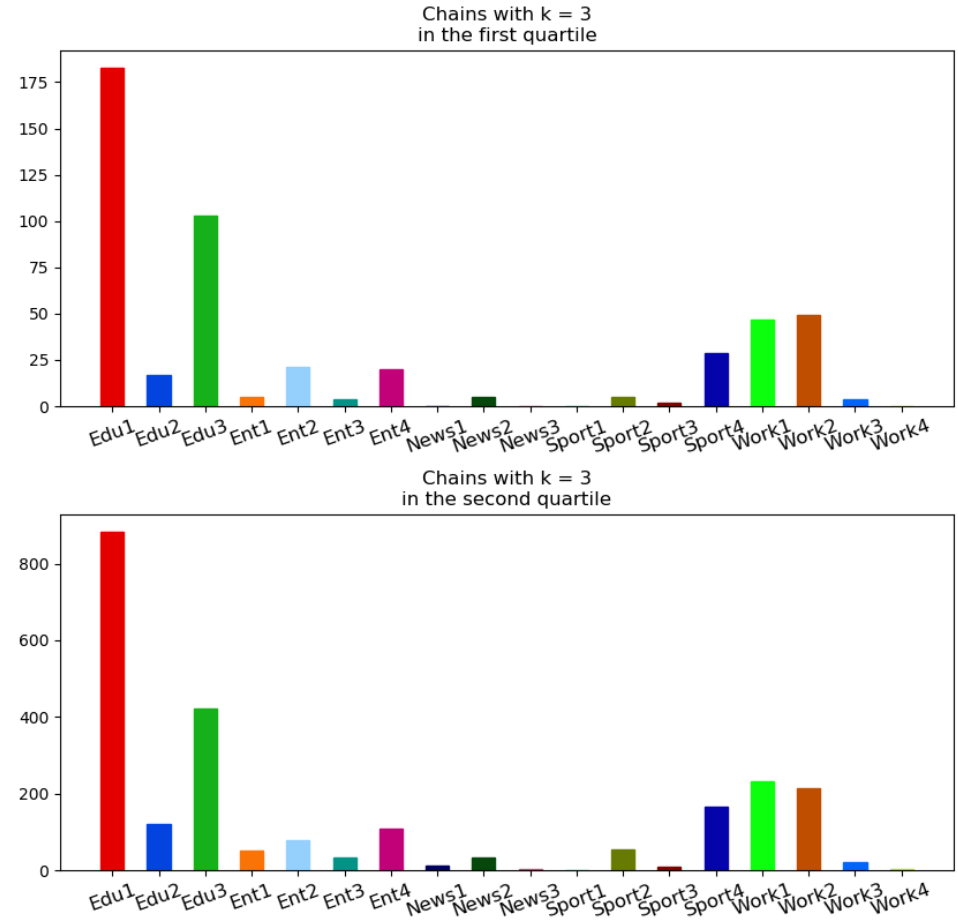
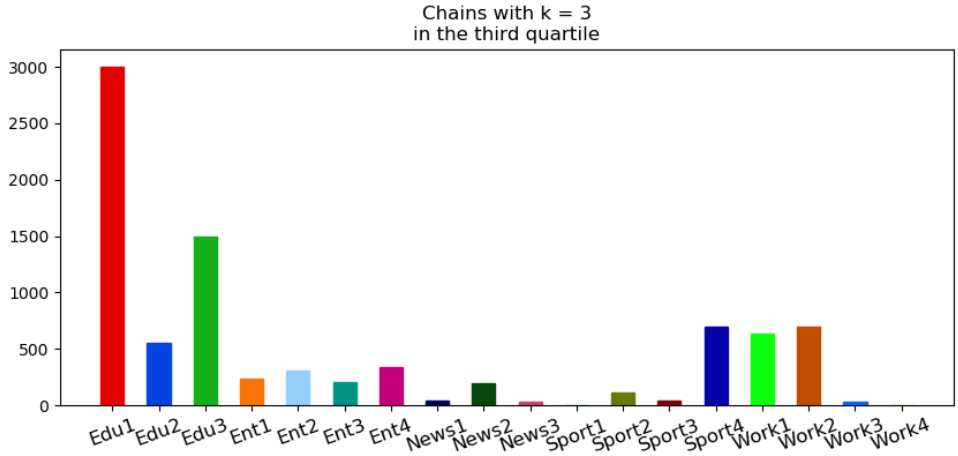
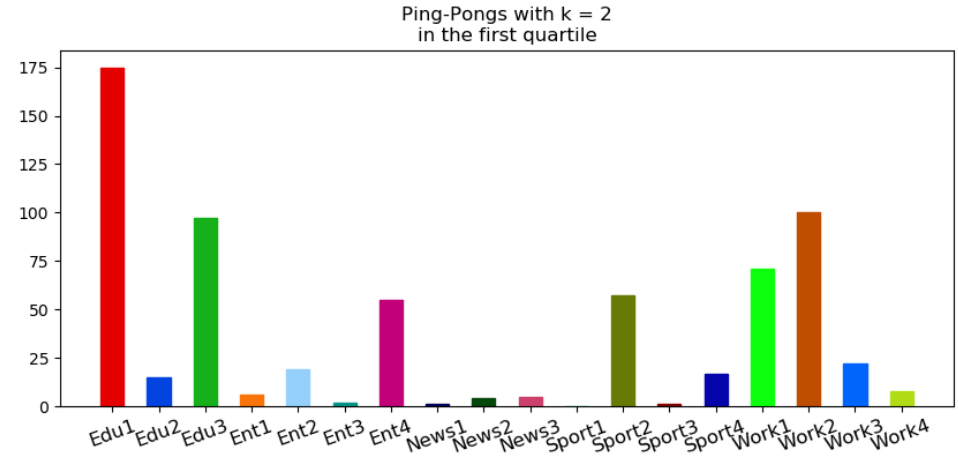


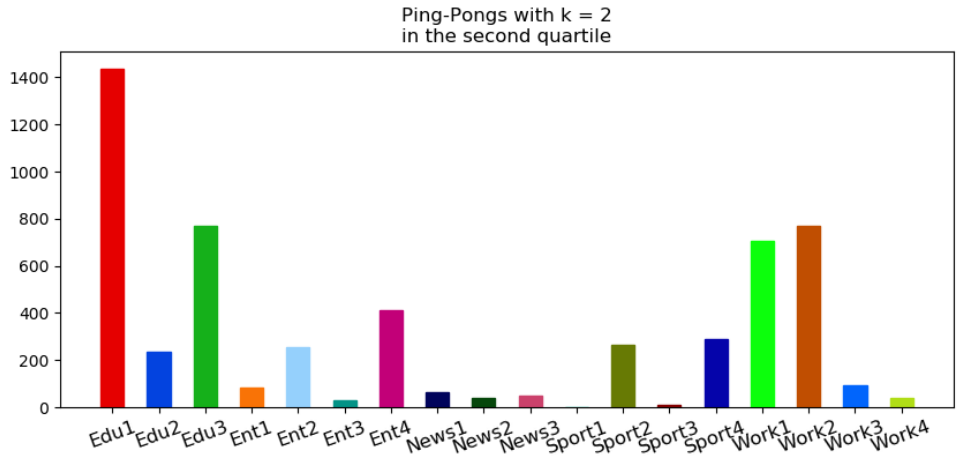
Figure A.5: The quartile-based plots related to the results obtained for our motifs in the Post Graph with the time window sliding approach

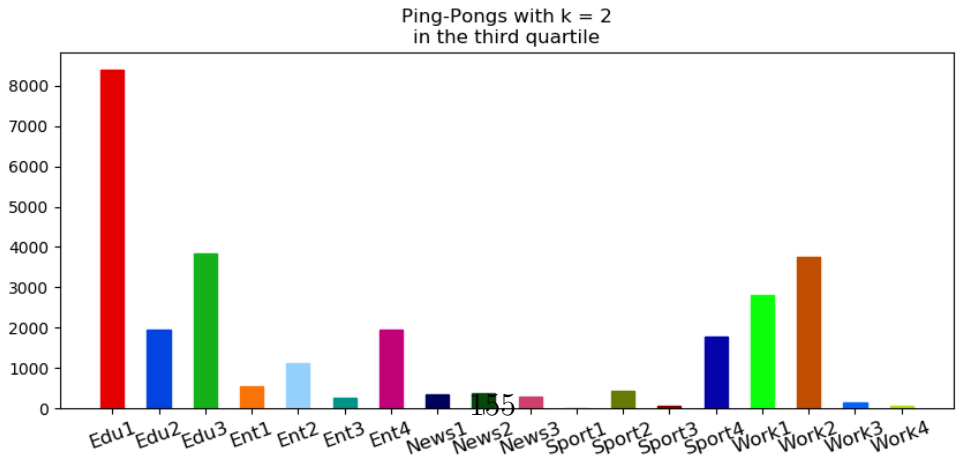
In **Figure A.6** we show the quartile-based diagrams elaborated on the results of the number of motifs found in the Post Graph with the snapshots approach.

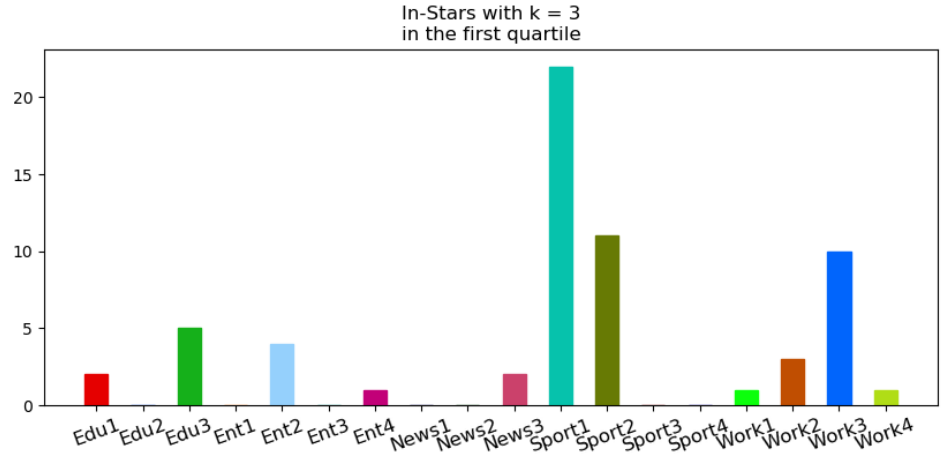


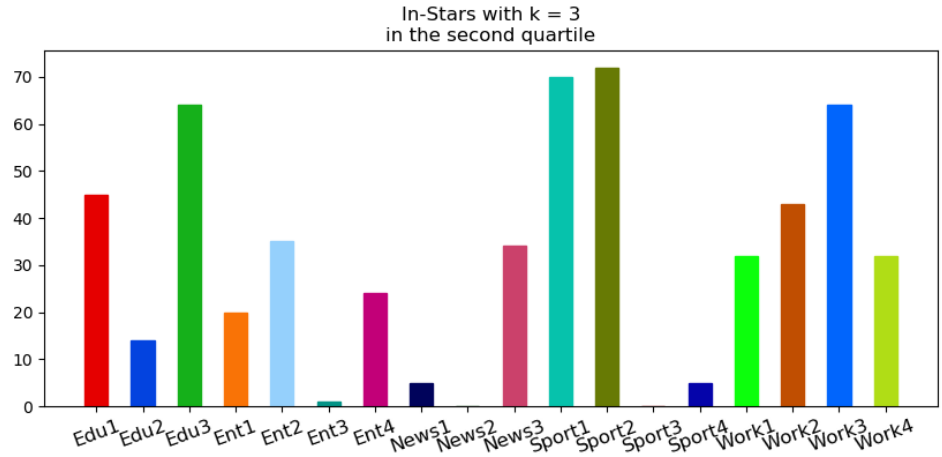

Chains with k = 3
in the third quartile
3000
2500
2000
1500
1000
500
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

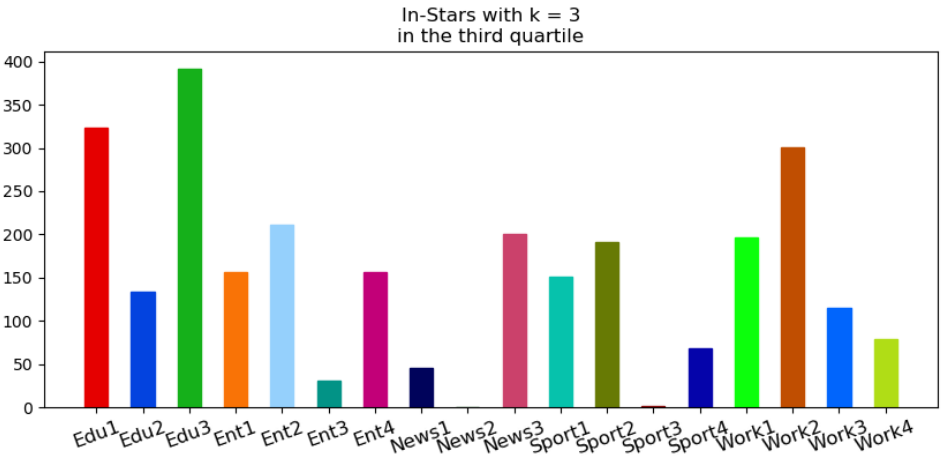

Ping-Pongs with k = 2
in the first quartile
175
150
125
100
75
50
25
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

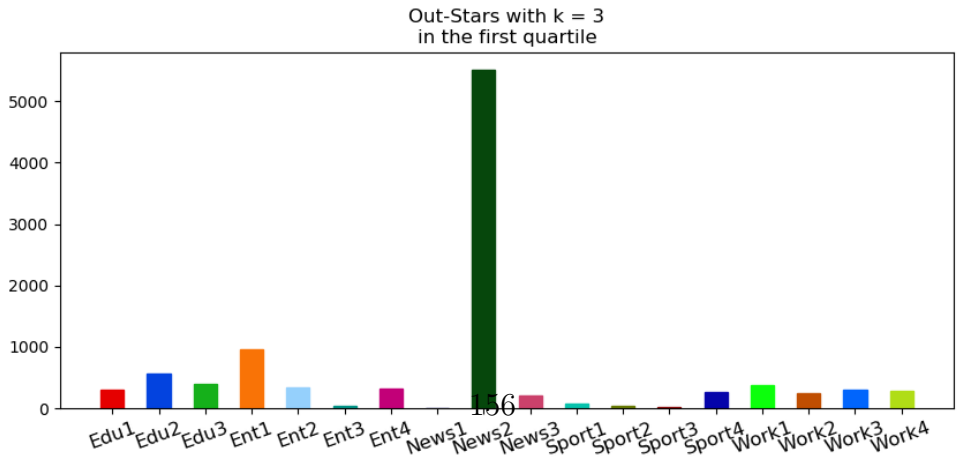

Ping-Pongs with k = 2
in the second quartile
1400
1200
1000
800
600
400
200
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

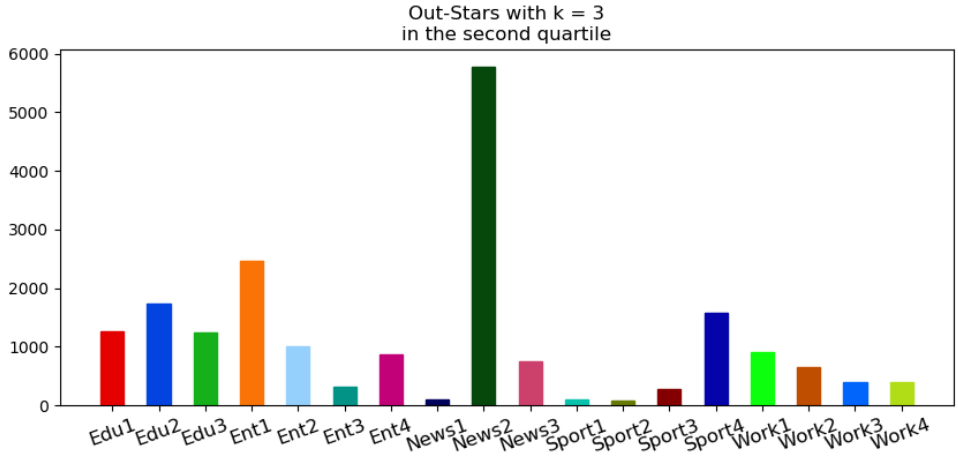

Ping-Pongs with k = 2
in the third quartile
8000
7000
6000
5000
4000
3000
2000
1000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

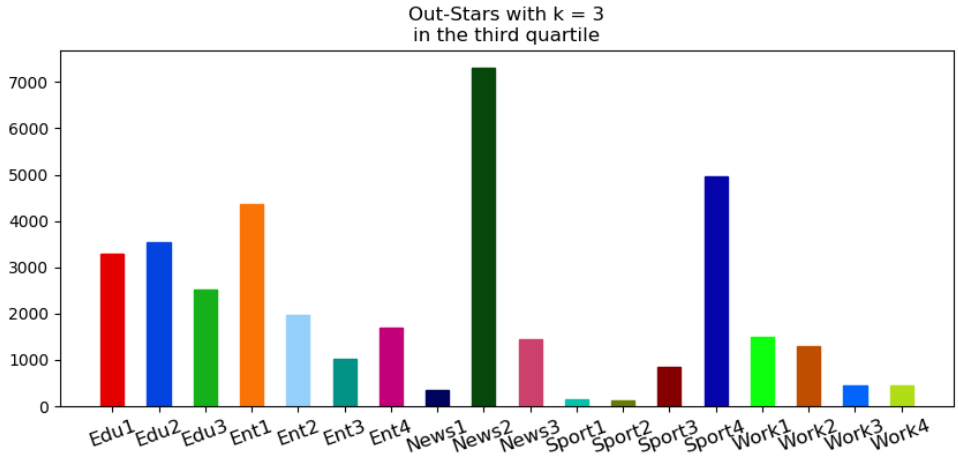

In-Stars with k = 3
in the first quartile
0
5
10
15
20
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


In-Stars with k = 3
in the second quartile
0
10
20
30
40
50
60
70
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


In-Stars with k = 3
in the third quartile
0
50
100
150
200
250
300
350
400
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

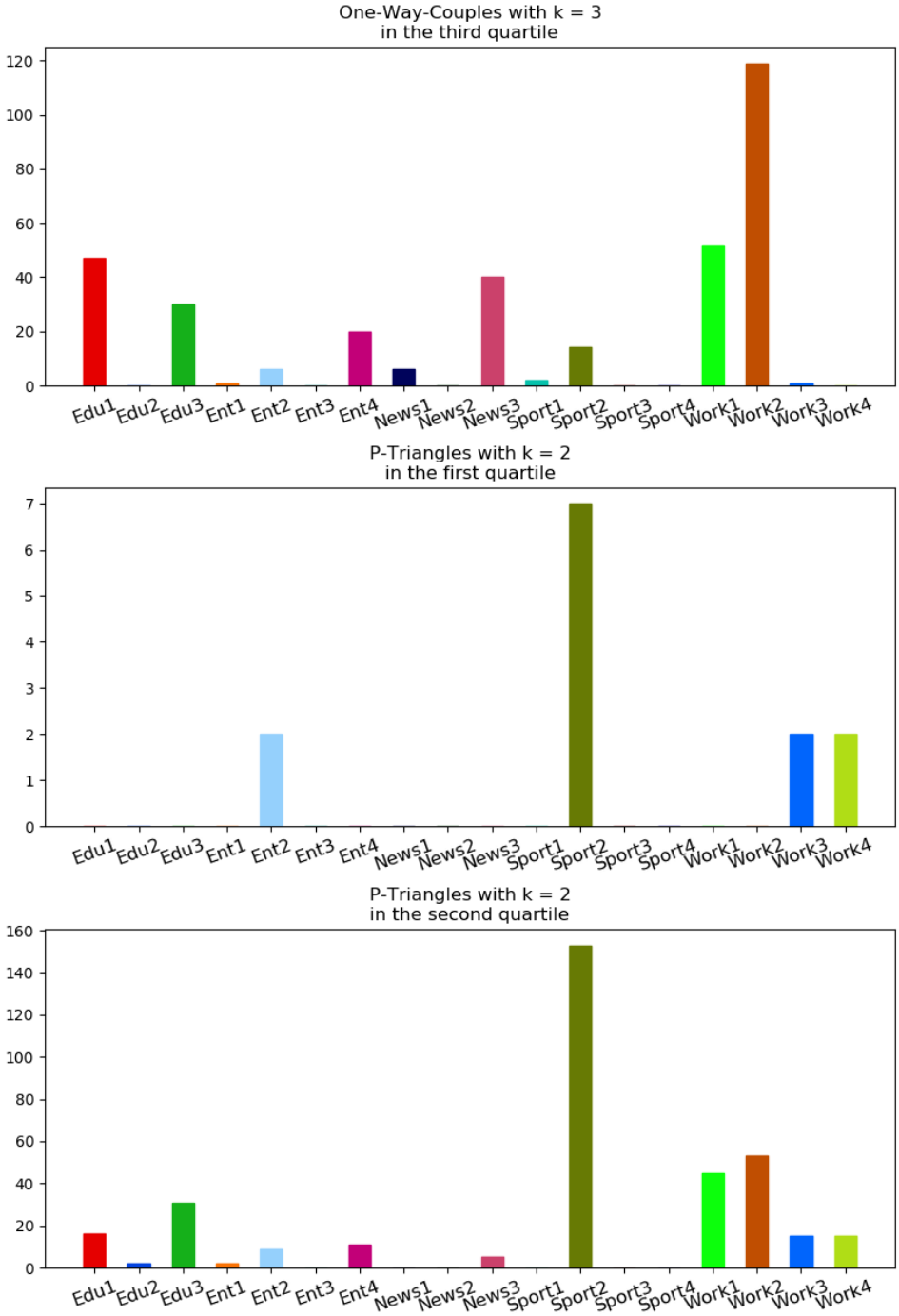

Out-Stars with k = 3
in the first quartile
0
1000
2000
3000
4000
5000
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


Out-Stars with k = 3
in the second quartile
6000
5000
4000
3000
2000
1000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


Out-Stars with k = 3
in the third quartile
7000
6000
5000
4000
3000
2000
1000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


One-Way-Couples with k = 3
in the first quartile
1.0
0.8
0.6
0.4
0.2
0.0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


One-Way-Couples with k = 3
in the second quartile
7
6
5
4
3
2
1
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


One-Way-Couples with k = 3
in the third quartile
120
100
80
60
40
20
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4
P-Triangles with k = 2
in the first quartile
7
6
5
4
3
2
1
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4
P-Triangles with k = 2
in the second quartile
160
140
120
100
80
60
40
20
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

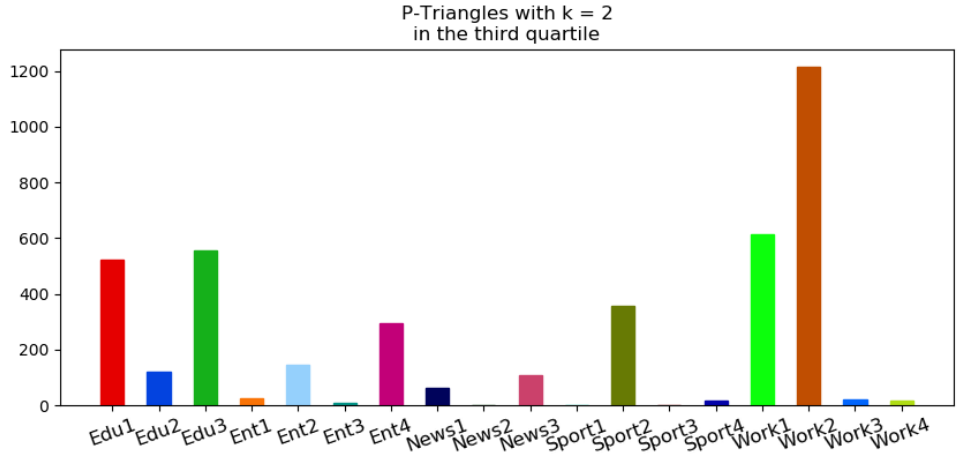
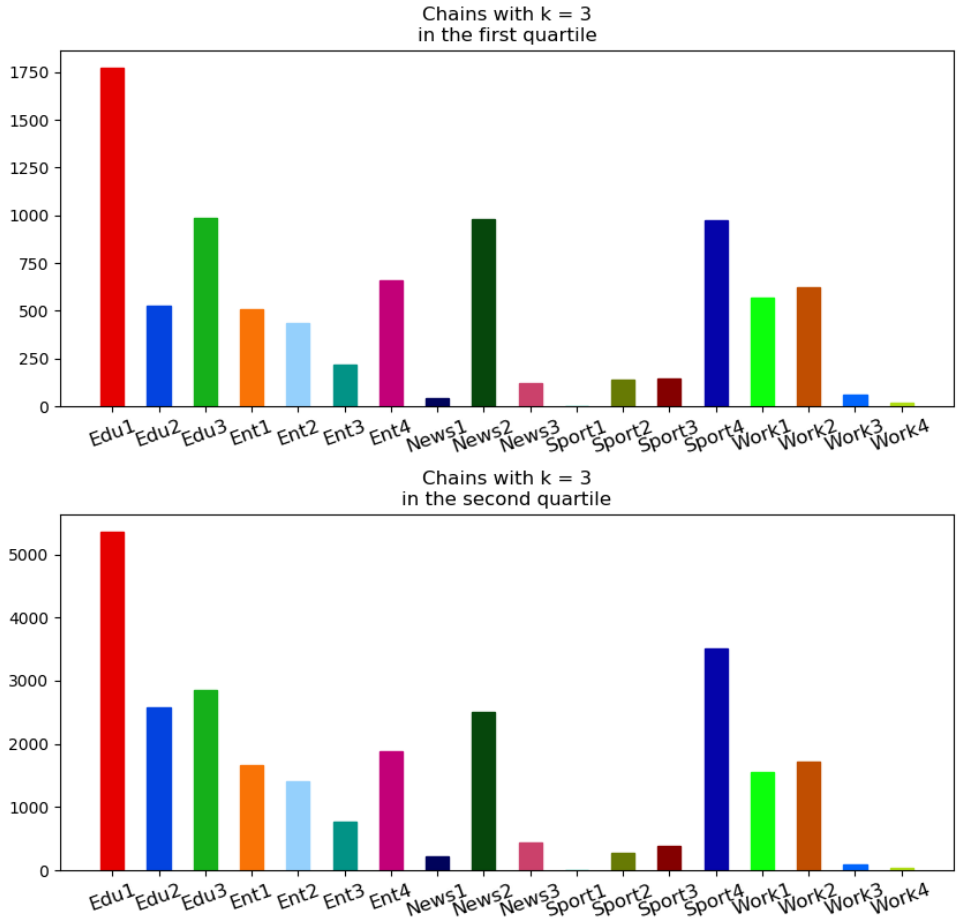
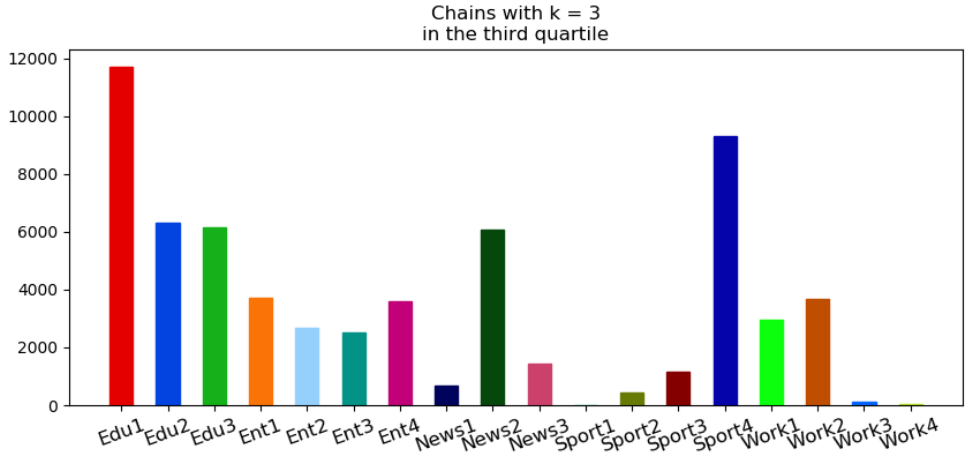
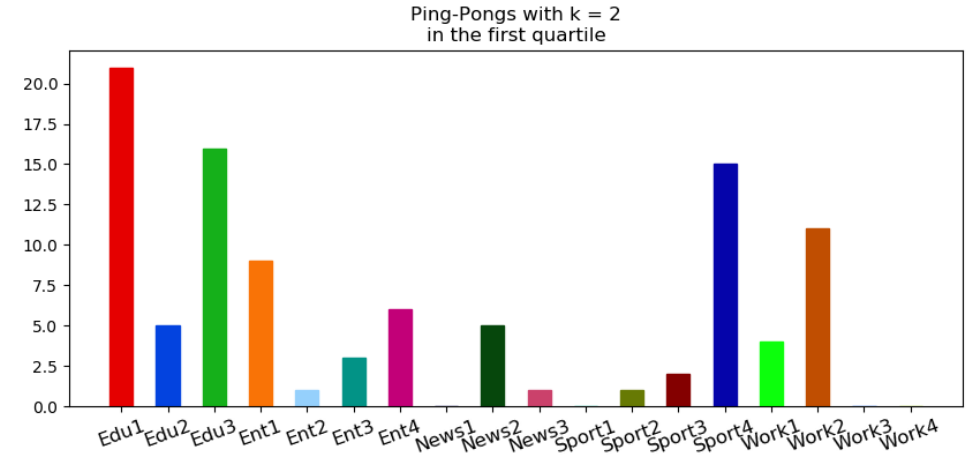


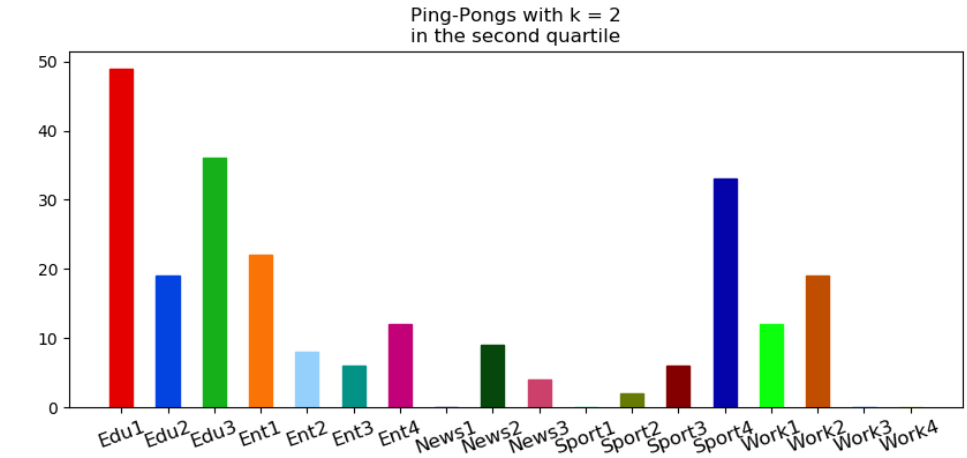
Figure A.6: The quartile-based plots related to the results obtained for our motifs in the Post Graph with the snapshots approach

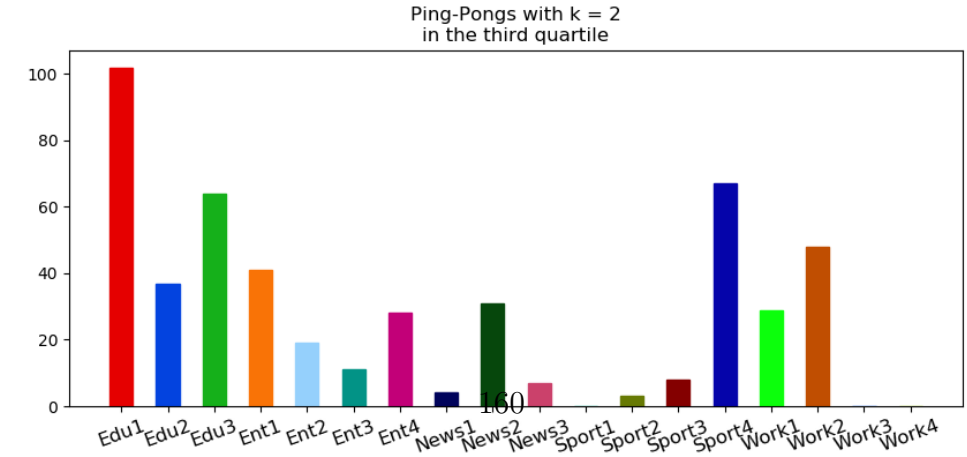
In **Figure A.7** we show the group-based diagrams elaborated on the results of the number of motifs found in the User Graph with the time window sliding approach.

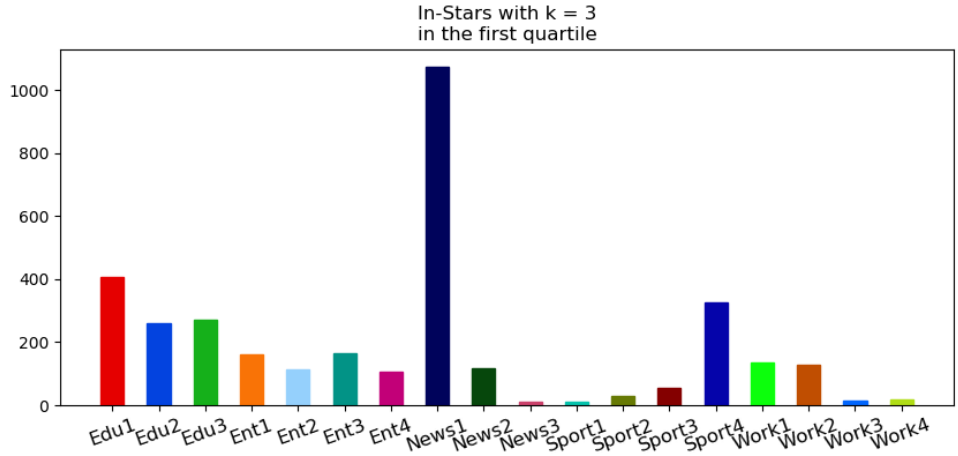


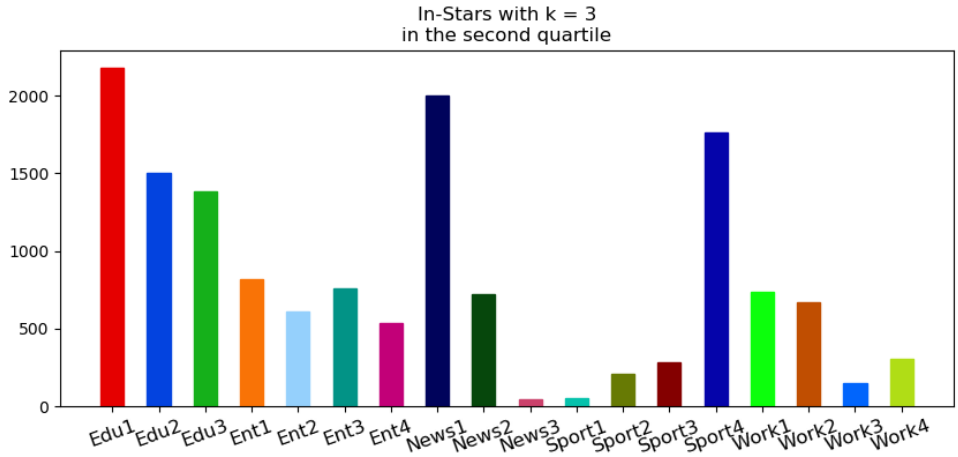

Chains with k = 3
in the third quartile
12000
10000
8000
6000
4000
2000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

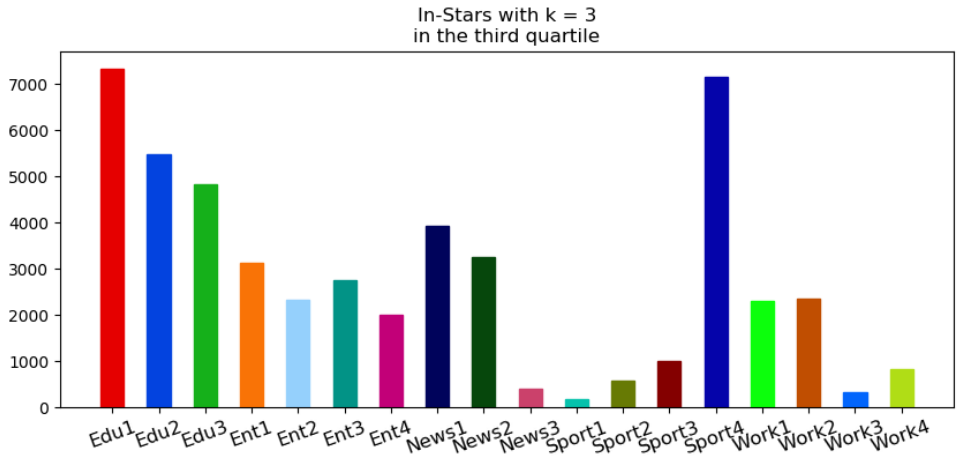

Ping-Pongs with k = 2
in the first quartile
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

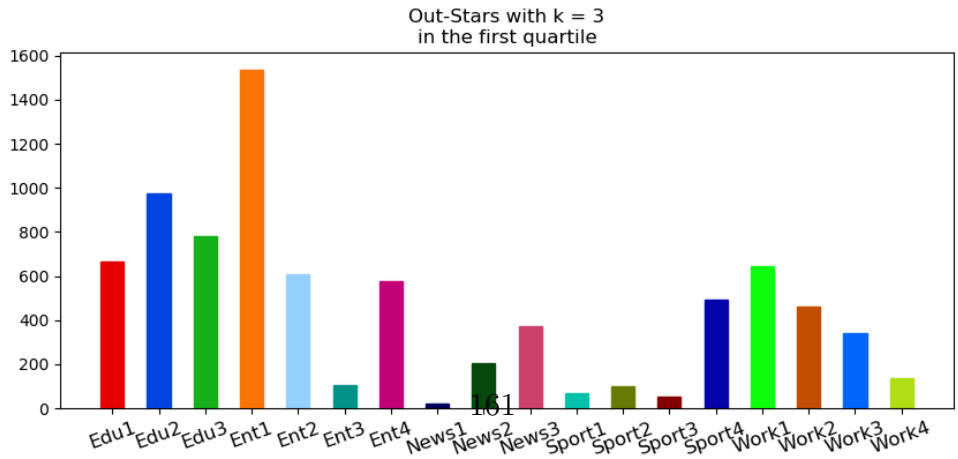

Ping-Pongs with k = 2
in the second quartile
50
40
30
20
10
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

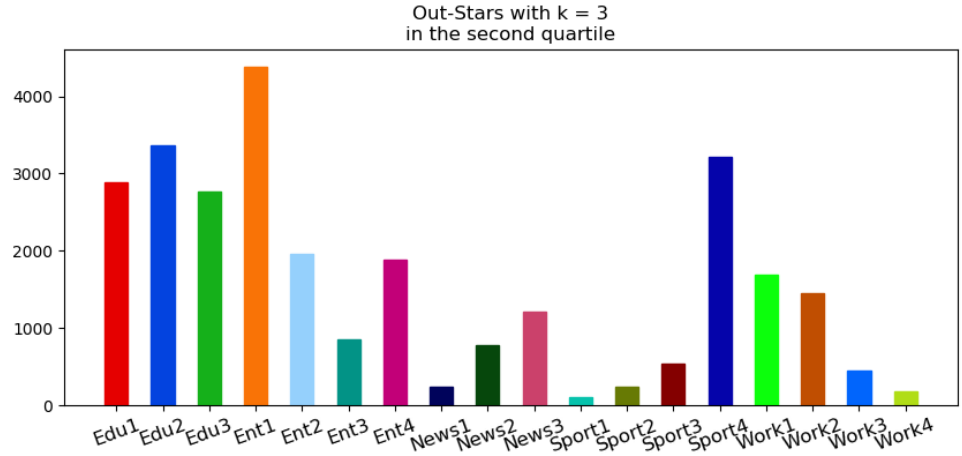

Ping-Pongs with k = 2
in the third quartile
100
80
60
40
20
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

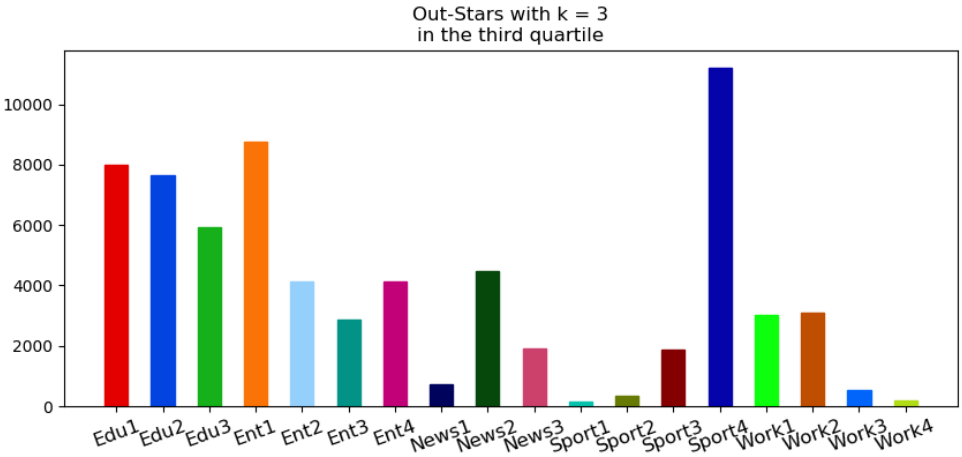

In-Stars with k = 3
in the first quartile
0
200
400
600
800
1000
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

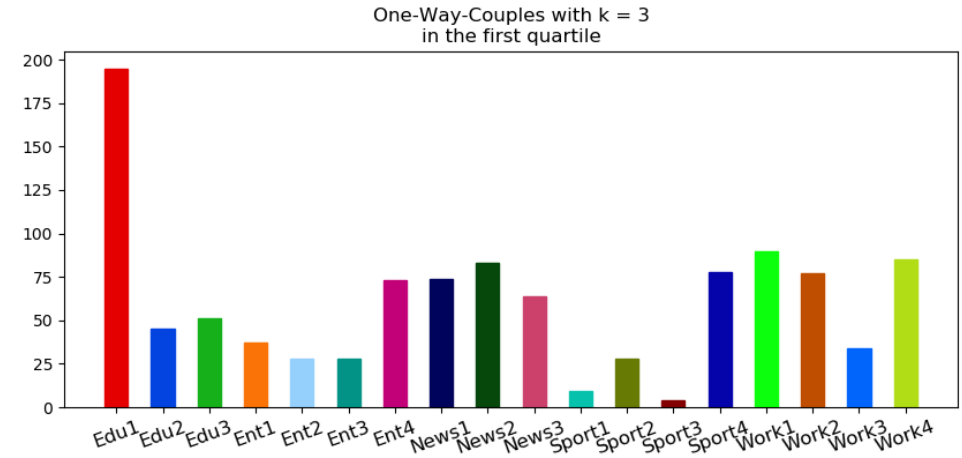

In-Stars with k = 3
in the second quartile
0
500
1000
1500
2000
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


In-Stars with k = 3
in the third quartile
0
1000
2000
3000
4000
5000
6000
7000
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


Out-Stars with k = 3
in the first quartile
0
200
400
600
800
1000
1200
1400
1600
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


Out-Stars with k = 3
in the second quartile
4000
3000
2000
1000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


Out-Stars with k = 3
in the third quartile
10000
8000
6000
4000
2000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


One-Way-Couples with k = 3
in the first quartile
200
175
150
125
100
75
50
25
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

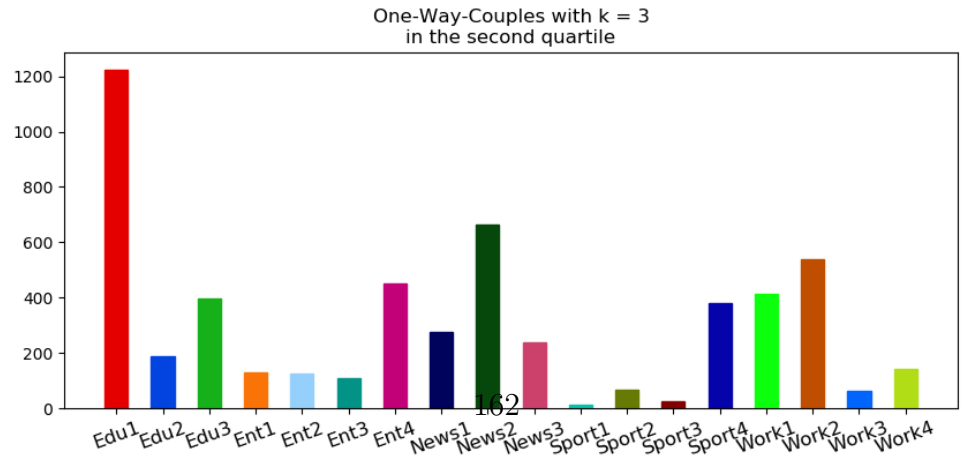
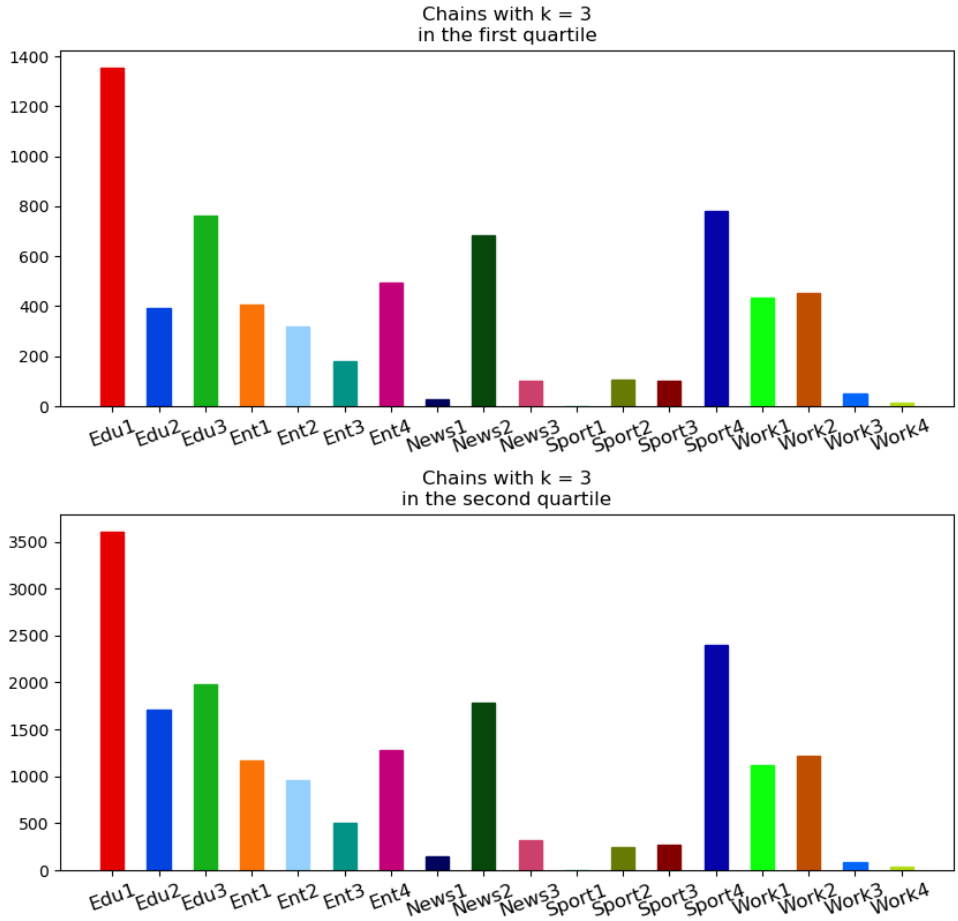
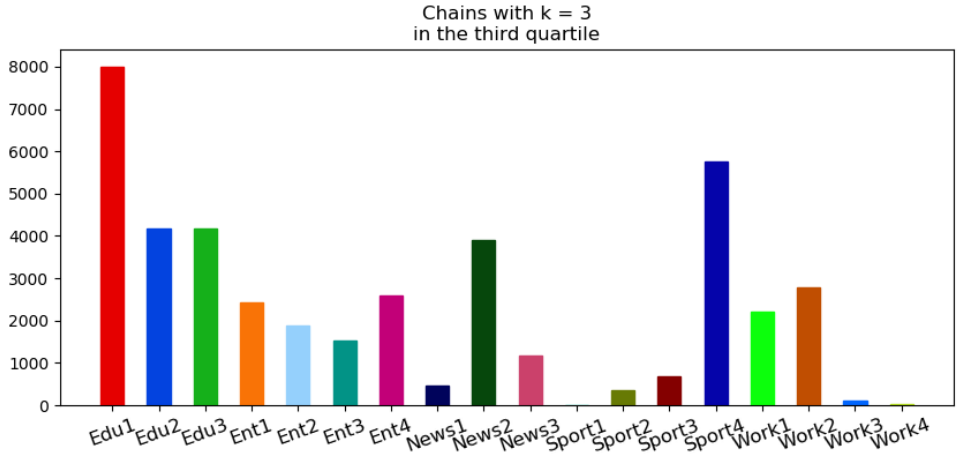
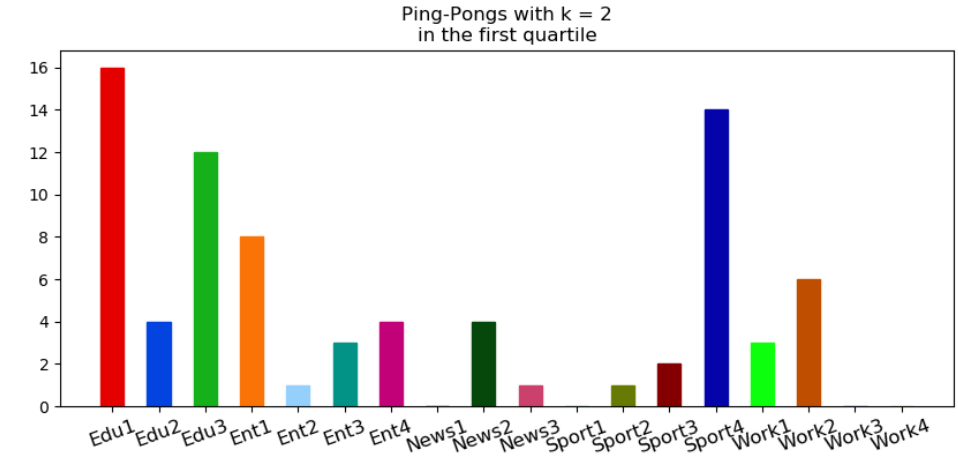

One-Way-Couples with k = 3
in the second quartile
1200
1000
800
600
400
200
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

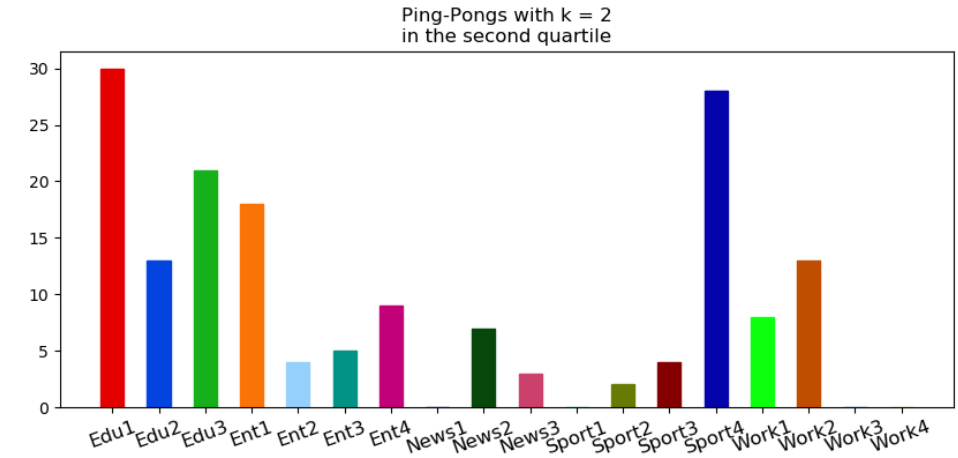
Figure A.7: The quartile-based plots related to the results obtained for our motifs in the User Graph with the time window sliding approach

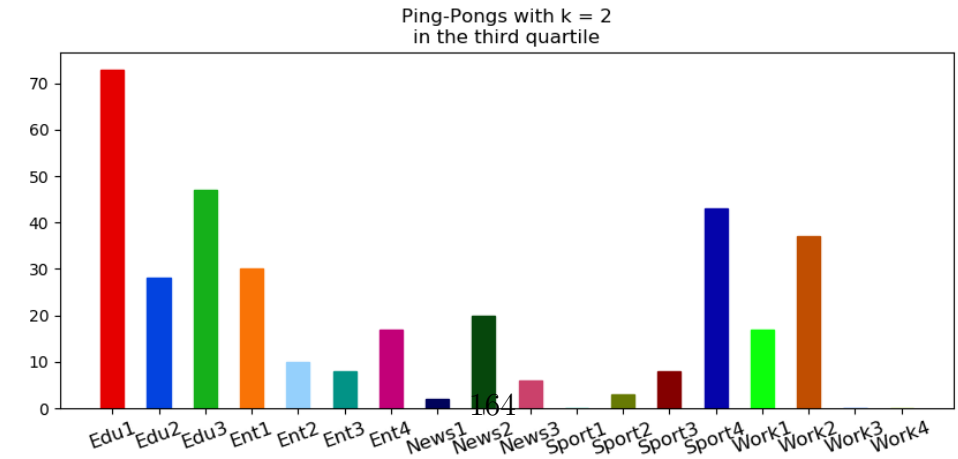
Finally, **Figure A.8** contains the quartile-based plots relating to the number of motifs we found in the User Graph using the snapshots approach.

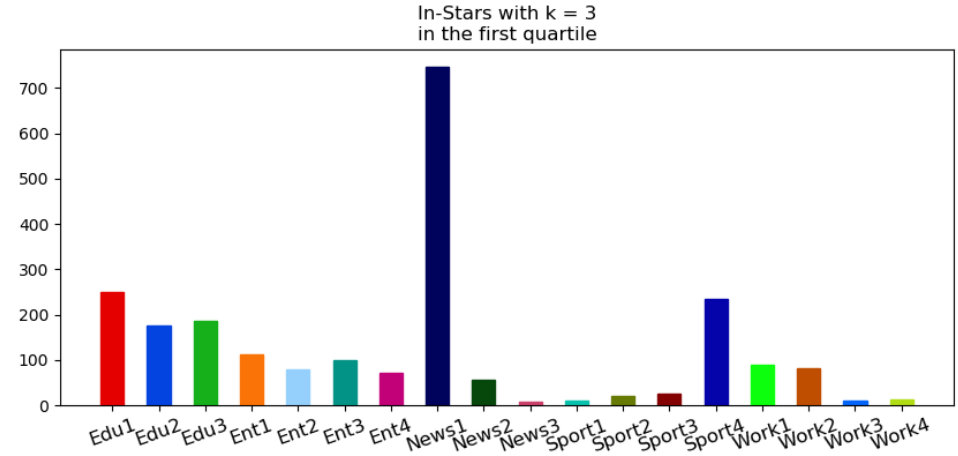


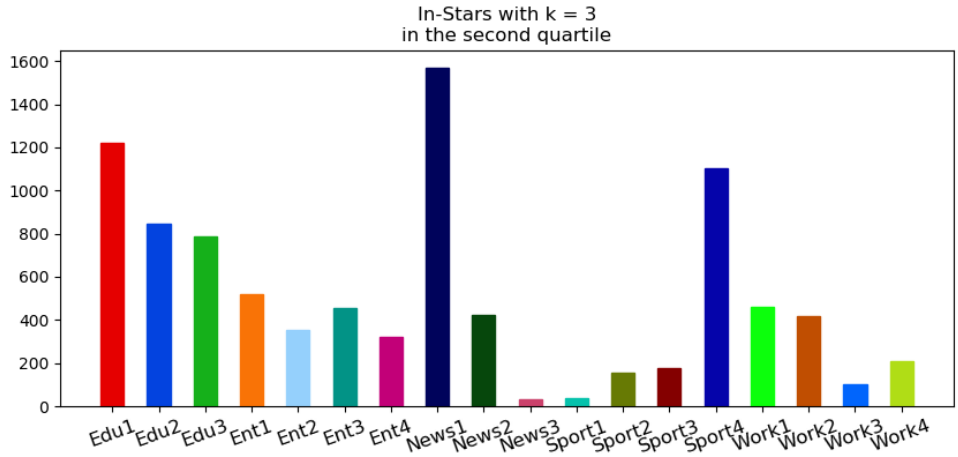

Chains with k = 3
in the third quartile
8000
7000
6000
5000
4000
3000
2000
1000
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

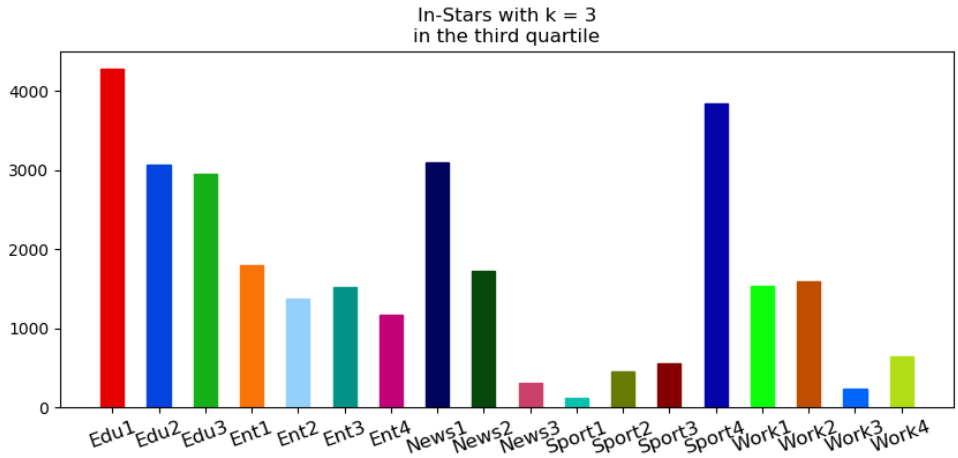

Ping-Pongs with k = 2
in the first quartile
16
14
12
10
8
6
4
2
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

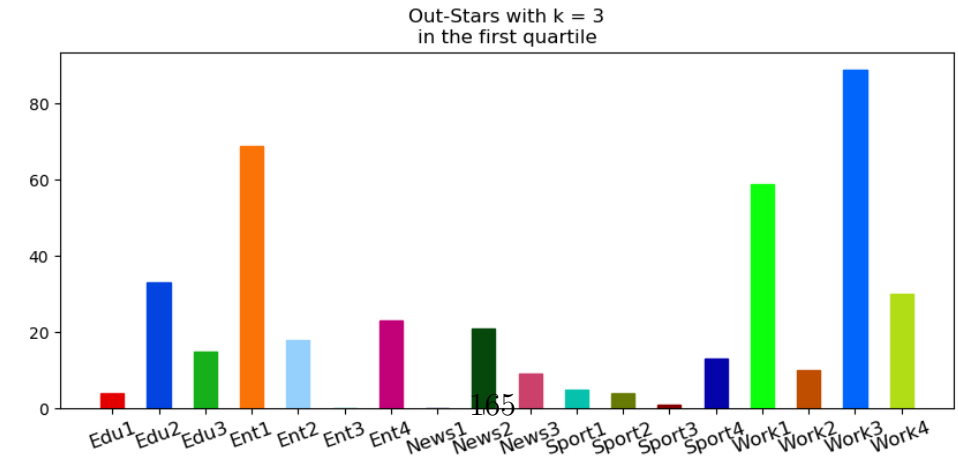

Ping-Pongs with k = 2
in the second quartile
30
25
20
15
10
5
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4

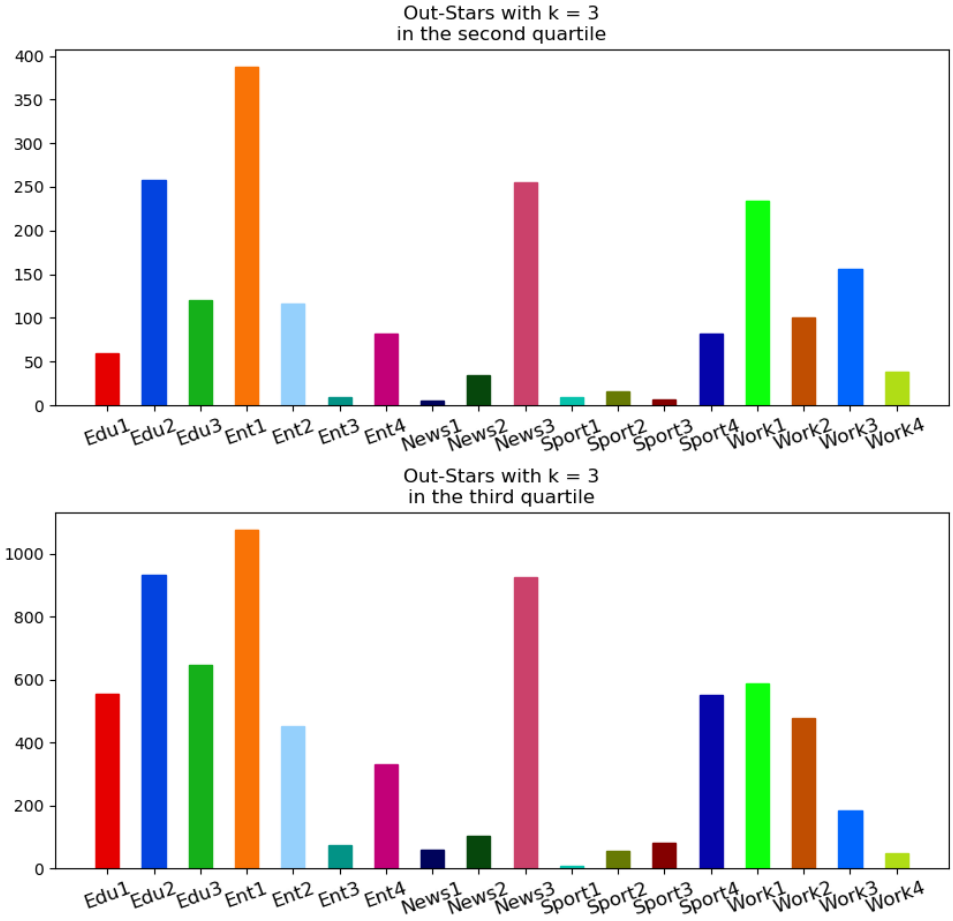

Ping-Pongs with k = 2
in the third quartile
70
60
50
40
30
20
10
0
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


In-Stars with k = 3
in the first quartile
0
100
200
300
400
500
600
700
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


In-Stars with k = 3
in the second quartile
0
200
400
600
800
1000
1200
1400
1600
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


In-Stars with k = 3
in the third quartile
0
1000
2000
3000
4000
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


Out-Stars with k = 3
in the first quartile
0
20
40
60
80
Edu1 Edu2 Edu3 Ent1 Ent2 Ent3 Ent4 News1 News2 News3 Sport1 Sport2 Sport3 Sport4 Work1 Work2 Work3 Work4


Out-Stars with k = 3
in the second quartile
400
350
300
250
200
150
100
50
0
Edu1
Edu2
Edu3
Ent1
Ent2
Ent3
Ent4
News1
News2
News3
Sport1
Sport2
Sport3
Sport4
Work1
Work2
Work3
Work4
Out-Stars with k = 3
in the third quartile
1000
800
600
400
200
0
Edu1
Edu2
Edu3
Ent1
Ent2
Ent3
Ent4
News1
News2
News3
Sport1
Sport2
Sport3
Sport4
Work1
Work2
Work3
Work4

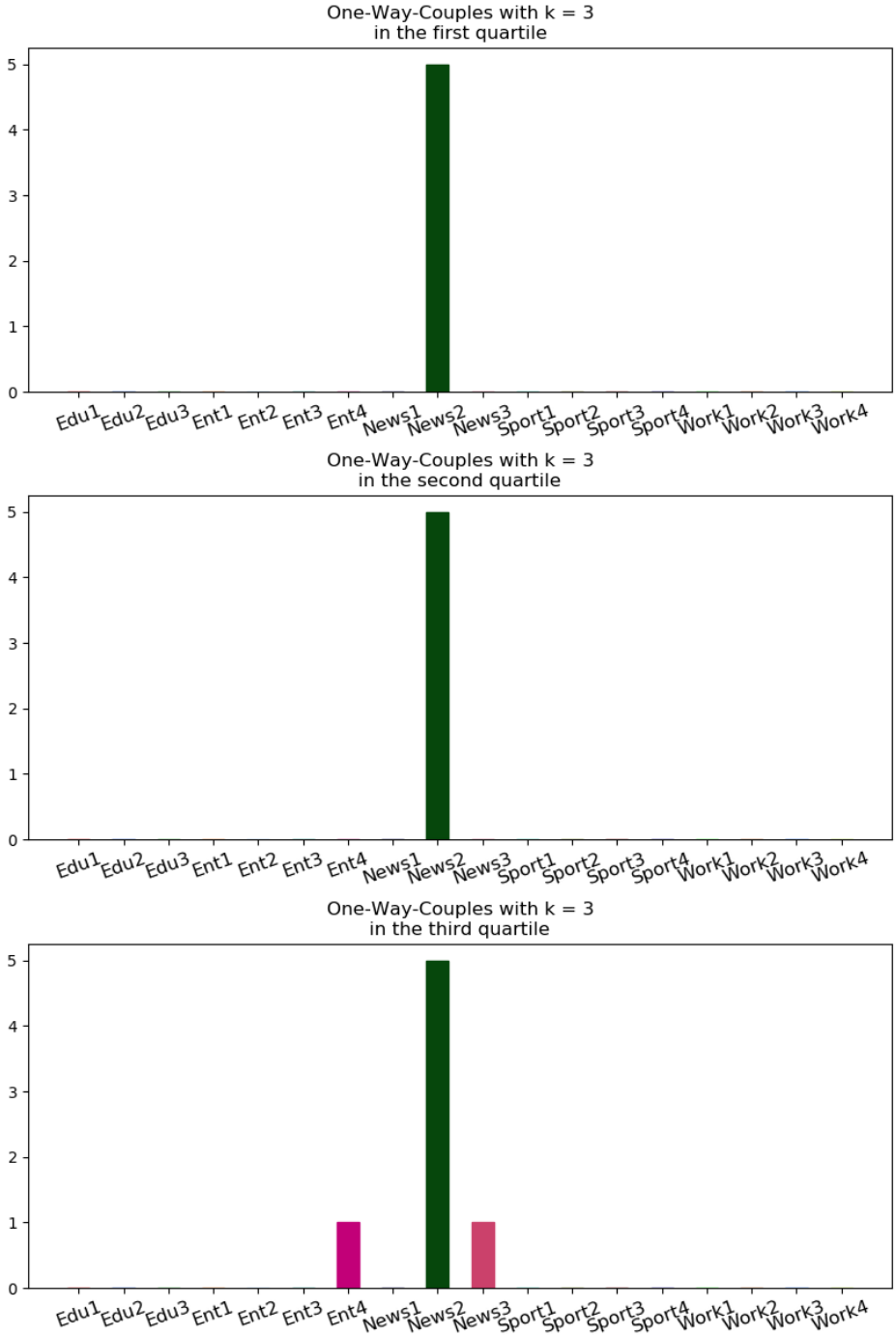


Figure A.8: The quartile-based plots related to the results obtained for our motifs in the User Graph with the time window sliding approach

Bibliography

[1] Lauri Kovanen, Kimmo Kaski, János Kertész, and Jari Saramäki. Temporal motifs reveal homophily, gender-specific patterns, and group talk in call sequences. *Proceedings of the National Academy of Sciences*, 110(45):18070–18075, 2013.

[2] Ashwin Paranjape, Austin R Benson, and Jure Leskovec. Motifs in temporal networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 601–610. ACM, 2017.

[3] Qiankun Zhao, Yuan Tian, Qi He, Nuria Oliver, Ruoming Jin, and Wang-Chien Lee. Communication motifs: a tool to characterize social communications. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1645–1648. ACM, 2010.

[4] Lauri Kovanen, Márton Karsai, Kimmo Kaski, János Kertész, and Jari Saramäki. Temporal motifs in time-dependent networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(11):P11005, 2011.

[5] Andrew Mellor. The temporal event graph. *Journal of Complex Networks*, 6(4):639–659, 2017.

[6] Matthieu Latapy, Tiphaine Viard, and Clémence Magnien. Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining*, 8(1):61, 2018.

[7] Zhao Zhao, Maleq Khan, VS Anil Kumar, and Madhav V Marathe. Subgraph enumeration in large social contact networks using parallel color coding and streaming. In *2010 39th International Conference on Parallel Processing*, pages 594–603. IEEE, 2010.

[8] Saket Gurukar, Sayan Ranu, and Balaraman Ravindran. Commit: A scalable approach to mining communication motifs from dynamic networks. In *Proceed-*

ings of the 2015 ACM SIGMOD International Conference on Management of Data, pages 475–489. ACM, 2015.

[9] Mohammed Elseidy, Ehab Abdelhamid, Spiros Skiadopoulos, and Panos Kalnis. Grami: Frequent subgraph and pattern mining in a single large graph. *Proceedings of the VLDB Endowment*, 7(7):517–528, 2014.

[10] Lotte Romijn, Breanndán Ó Nualláin, and Leen Torenvliet. Discovering motifs in real-world social networks. In *International Conference on Current Trends in Theory and Practice of Informatics*, pages 463–474. Springer, 2015.

[11] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995.

[12] Mira Gonen and Yuval Shavitt. Approximating the number of network motifs. *Internet Mathematics*, 6(3):349–372, 2009.