National Technical University of Athens

School of Electrical and

Computer Engineering

Network Management and

Optimal Design Laboratory

Community Detection and Resource Assignment in Interdependent Systems via Complex Network Analysis

A dissertation submitted for the degree of Doctor of Philosophy

by

Konstantinos Tsitseklis

This research is co-financed by Greece and the European Union (European Social Fund- ESF) through the Operational Programme «Human Resources Development, Education and Lifelong Learning» in the context of the project "Strengthening Human Resources Research Potential via Doctorate Research – 2nd Cycle" (MIS-5000432), implemented by the State Scholarships Foundation (IKY).

July 2021

National Technical University of Athens
School of Electrical and Computer Engineering
Division Of Communication, Electronic and Information Engineering
Network Management and Optimal Design Laboratory

Community Detection and Resource Assignment in Interdependent Systems via Complex Network Analysis

A dissertation submitted for the degree of Doctor of Philosophy

by

Konstantinos Tsitseklis

Advisory Committee: Symeon Papavassiliou, Miltiadis Anagnostou, Ioanna Roussaki

Approved by the seven-member committee on 12th July 2021.

Athens, July 2021

....

Τσιτσεκλής Κωνσταντίνος,

Διδάκτωρ Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών

Acknowledgments

Writing this chapter and reminiscing the years during which I pursued my Ph.D., I feel that there are a lot of people I should thank for their support.

First of all, I would like to thank my advisor, Professor Symeon Papavassiliou, for his guidance and support throughout my Ph.D. studies. I am grateful for his trust in me as well as his valuable input to the problems that arose during my studies. He supported me in my research endeavours, always willing to help and provided the necessary encouragement. Moreover, he gave me the chance to join the NETMODE team and collaborate with many esteemed colleagues.

A special thanks is owed to Associate Professor Vasileios Karyotis. When I was an undergraduate student, Vassilis inspired me to study the field of social network analysis. I thank him for his valuable support during my Ph.D. studies, his guidance, the productive collaboration and the many interesting discussions we had during the past years.

I am also deeply grateful to my good friends and colleagues Grigorios Kakkavas and Margarita Vitoropoulou. In addition to the fruitful collaboration I had with both of them, I feel honored to have met them and consider them as my good friends who were always willing to listen to my problems and provide me with the necessary psychological support. I would also like to thank all the other colleagues in NETMODE laboratory for the precious (and fun) moments we shared the past years.

A special thanks goes to my girlfriend Lena, who was always there for me, believed in me and encouraged me constantly towards pursuing my goal, acquiring the Ph.D. I would also like to thank all my close friends for their support.

From the bottom of my heart, I would like to thank my family, my father Christos, my mother Eleni and my sister Evangelia, for their unconditional love, understanding and support. I owe a lot to my parents for making every sacrifice in order for me to be able to pursue my academic goals and for teaching me valuable life lessons.

for my parents, Christos and Eleni

Abstract

The aim of this thesis is to develop novel approaches for inferring relations and hidden similarities among the actors of complex systems that consist of various types of devices and users, as well as approaches for assigning content to them. In order to accomplish this, the developed methods take into account the interdependency and the various relationships of the multiple types of actors found in these systems. These approaches rely on tools and techniques from the fields of complex networks and social network analysis. Such systems are commonly observed in current interconnected environments, such as Smart Cities, and are expected to become even more prevalent in the future. These systems combine the operation of large infrastructure with the actions and requirements of people that have access to it. For the unobstructed operation of such topologies, network operators need to be able to monitor the generated data, detect possible redundancies, discover similar regions. Moreover, people using such environments need to have fast access to data and learn about relevant applications that keep their perceived quality of experience high. These entities (people, devices, data measurements) being the actors of such complex systems, are related in multiple ways, forming multi-layer complex networks, and highlighting the need to employ proper tools for their analysis.

Aiming to provide a framework for achieving the aforementioned goals, this thesis focuses on the problems of community detection and resource allocation in interconnected and interdependent environments. The identification of important problems observed in these areas and the development of suitable solutions can aid in identifying groups of similar devices operating within the interconnected environment, groups of similar users, and also distinguish the most influential ones in terms of information diffusion.

In particular, in order to deal with the problems of detecting clusters of generated data from the infrastructure and also communities of people in Online Social Networks (OSNs) existing in interdependent and complex systems, a novel community detection algorithm is developed. A new framework is presented for mapping the problem of data clustering to a community detection one. The proposed algorithm manages to discover meaningful clusters of data, outperforming some traditional data clustering approaches in terms of accuracy

and also detect communities in OSNs resulting in high modularity scores. Inspired by the well-known Girvan-Newman (GN) algorithm, it manages to perform many operations faster, leveraging on network embedding in hyperbolic space and by introducing a new approximative network metric for estimating the edge betweenness centrality. Combined with the removal of more edges per iteration instead of a single one as in the case of GN and coupled with a graph database, it marks a more scalable approach than GN for large networks that are oftentimes observed in realistic complex systems. The evaluation process on both synthetic and real data, showcases the benefits of adopting the proposed approach.

The people that use the facilities of such interdependent systems, interact with each other by using OSNs. Focusing on these social relations and studying their interactions can reveal the manners in which information flows throughout the network. The monitoring of the information diffusion across the network arises as one of the most crucial aspects for estimating the possible outcome of seeding sets of users with units of information (recommendations). Considering that each user displays a relevance score towards each available item for recommendation, the problem of assigning recommendations to users is formed as a relevance maximization one. Contrary to other works, in this approach, the tolerance of a user to different levels of recommendations is considered as a major factor for the development of the recommender system for the first time. Complex constraints on the amount of duplicate and distinct recommendations are imposed per user. The maximization problem is proven to be computationally difficult, as it consists of an NP-hard problem with added constraints. In order to overcome this computational obstacle, the problem is divided into two sub-problems treated with greedy algorithms, and their combination produces high relevance scores, while respecting all the imposed constraints.

Aspiring to provide users with fast access to data that increases the users' quality of experience, various schemes for caching at the network edge by utilizing limited memory space in the User Equipment (UE), are examined . Knowledge obtained from recommender systems about each user's preferences can be applied in order to predict future requests. In order to decide the optimal content to cache in each UE, the problem of content placement is formulated as a cache hit maximization one. Algorithms that employ either the full set or only a portion of the set of users as caches, while caching contents either proactively or reactively, are examined and compared in terms of the overall cache hit ratio obtained.

The increased obtained cache-hit ratio proves the benefits of utilizing caching at the UEs instead of just caching at special devices. Also, from these comparisons, the need to take into consideration the probabilities of request for more than one's self in order to design more accurate caching schemes is highlighted. Finally, leveraging users' mobility and by taking into account the impact that recommendations have on users' requests and the ability of dedicated devices and selected users' UEs to cache and offload content, a caching and recommendation scheme is developed. Modeling the perceived user's Quality of Experience (QoE) as a function of the delay experienced by the user for retrieving a requested item and the relevance of the recommended items to her preferences, the problem is formulated as QoE maximization one. Knowing that this problem is NP-hard, a heuristic method is developed and compared to an approximative algorithm showcasing its benefits in terms of balancing the achieved QoE score for each user and the execution time needed, marking it as a computationally feasible approach able to yield results of high QoE.

In the following, the proposed methods are presented, alongside with a discussion on the main contributions of this thesis. Then, each Chapter focuses on one of the aforementioned problems, presenting related work on the field and introducing the developed solutions together with some indicative evaluation that justifies the benefits of their adoption.

Keywords: Community detection, Resource assignment, Complex networks, Information diffusion, Edge caching, Online social networks, Recommender systems

Abstract in Greek

Περίληψη Στα Ελληνικά

Η παρούσα διατριβή εστιάζει στην ανάπτυξη καινοτόμων τεχνικών με σκοπό την ανακάλυψη των σχέσεων και των κρυφών συσχετίσεων μεταξύ των οντοτήτων σύνθετων συστημάτων. Τα συστήματα αυτά αποτελούνται από διάφορους τύπους συσκευών αλλά και ανθρώπους. Ακόμα η διατριβή εστιάζει και στην ανάθεση πόρων στις οντότητες ενός σύνθετου συστήματος. Για να επιτευχθούν αυτοί οι στόχοι, οι προτεινόμενες μέθοδοι λαμβάνον υπόψιν την αλληλεξάρτηση και τις ποικίλες σχέσεις μεταξύ των πολλών διαφορετικών οντοτήτων. Αυτές οι μέθοδοι βασίζονται σε τεχνικές και εργαλεία από τους τομείς της θεωρίας γραφημάτων και της ανάλυσης κοινωνικών δικτύων. Συστήματα σαν αυτά που μελετώνται σε αυτήν τη διατριβή, παρατηρούνται σε σύγχρονα διασυνδεδεμένα περιβάλλοντα όπως αυτά που αποτελούν οι Έξυπνες Πόλεις και αναμένεται να γίνουν ακόμα περισσότερα στο μέλλον. Αυτά τα συστήματα συνδυάζουν τη λειτουργία μεγάλων υποδομών με τις ενέργειες και τις απαιτήσεις των ανθρώπων που αποκτούν πρόσβαση σε αυτές. Για την ανεμπόδιστη λειτουργία τέτοιων τοπολογιών, οι διαχειριστές του δικτύου πρέπει να είναι σε θέση να επιθεωρούν τα δεδομένα που παράγονται, να εντοπίζουν πιθανώς περιττό υλικό και να ανακαλύπτουν παρόμοιες περιοχές. Ακόμα, οι άνθρωποι που χρησιμοποιούν τέτοια περιβάλλοντα χρειάζεται να έχουν γρήγορη πρόσβαση σε δεδομένα αλλά και να έχουν τη δυνατότητα να μαθαίνουν σχετικά με εφαρμογές που θα κρατήσουν την ποιότητα της εμπειρίας που απολαμβάνουν σε υψηλά επίπεδα. Αυτές οι οντότητες (άνθρωποι, συσκευές, μετρήσεις) είναι τα στοιχεία που αποτελούν τα σύνθετα συστήματα και σχετίζονται μεταξύ τους με πολλούς τρόπους, δημιουργώντας πολυ-επίπεδα σύνθετα δίκτυα τα οποία χρειάζονται τα κατάλληλα εργαλεία για την ανάλυσή τους.

Με σκοπό τη δημιουργία ενός πλαισίου μεθόδων που θα ικανοποιεί τους παραπάνω στόχους, αυτή η διατριβή εστιάζει στα προβλήματα της ανίχνευσης κοινοτήτων και της ανάθεσης πόρων σε αλληλοεξαρτώμενα και διασυνδεδεμένα περιβάλλοντα. Η ανακάλυψη σημαντικών προβλημάτων στις περιοχές αυτές και η ανάπτυξη κατάλληλων λύσεων μπορεί να βοηθήσει στην ανακάλυψη ομάδων από παραπλήσιες συσκευές οι οποίες λειτουργούν σε αυτά τα περιβάλλοντα, ομάδων από παρόμοιους χρήστες καθώς και να ξεχωρίσει τους πιο επιδραστικούς από αυτούς από τη σκοπιά της διάχυσης πληροφορίας.

Για τον σκοπό της εξεύρεσης συστάδων από δεδομένα προερχόμενα από την υποδομή του περιβάλλοντος αλλά και της ανακάλυψης κοινοτήτων ατόμων σε Διαδικτυακά Κοινωνικά Δί-

κτυα (ΔΚΔ, Online Social Networks) τα οποία αποτελούν μέρος τέτοιων συστημάτων, παρουσιάζεται αλγόριθμος ανίχνευσης κοινοτήτων που έχει συνδεθεί με κατάλληλη βάση-γράφο για τη λειτουργία του. Ακόμα, παρουσιάζεται ένα νέο πλαίσιο με σκοπό τη μετατροπή ενός προβλήματος εξεύρεσης συστάδων σε πρόβλημα ανίχνευσης κοινοτήτων. Ο προτεινόμενος αλγόριθμος καταφέρνει να εντοπίζει συστάδες από δεδομένα που έχουν νόημα, υπερκεράζοντας σε ακρίβεια παραδοσιακές μεθόδους για τη συσταδοποίηση, καθώς και να ανιχνεύει κοινότητες σε ΔΚΔ που οδηγούν σε υψηλές τιμές αρθρωτότητας. Ο αλγόριθμος αυτός είναι εμπνευσμένος από τον γνωστό αλγόριθμο ανίχνευσης κοινοτήτων των Girvan-Newman (GN) και καταφέρνει να ολοκληρώνει γρηγορότερα αρκετές λειτουργίες βασιζόμενος στην ενσωμάτωση του δικτύου στον υπερβολικό γεωμετρικό χώρο και χρησιμοποιώντας μια προσεγγιστική μετρική για την εκτίμηση της κεντρικότητας ενδιαμεσικότητας ακμής. Σε συνδυασμό με την αφαίρεση ακμών κατά δέσμες, αντί για μοναδικής όπως στον GN και κάνοντας χρήση μιας βάσης δεδομένων-γράφο (graph database), αποτελεί μια πιο βιώσιμη προσέγγιση για μεγάλα δίκτυα από ότι ο GN. Μέσω της αξιολόγησης του αλγορίθμου σε πραγματικά και συνθετικά δεδομένα γίνονται ορατά τα πλεονεκτήματα της προτεινόμενης μεθόδου.

Μελετώντας τις αλληλεπιδράσεις μεταξύ των χρηστών των διασυνδεδεμένων περιβαλλόντων που πραγματοποιούνται με τη διαμεσολάβηση των Κοινωνικών Δικτύων, το ζήτημα της μελέτης της διάδοσης της πληροφορίας εντός του κοινωνικού δικτύου ξεχωρίζει ως ένα από τα πλέον σημαντικά για την εκτίμηση της διάδοσης συστάσεων με αφετηρία ορισμένους κατάλληλα επιλεγμένους χρήστες. Θεωρώντας ότι κάθε χρήστης παρουσιάζει ένα ποσό συνάφειας με κάθε πιθανό αντικείμενο για σύσταση, το πρόβλημα της ανάθεσης συστάσεων μοντελοποιείται ως ένα πρόβλημα μεγιστοποίησης της συνάφειας αυτής. Σε αντίθεση με προηγούμενες δουλειές, για πρώτη φορά, ο σεβασμός της ανοχής του χρήστη σε συστάσεις αποτελεί κομβικό σημείο. Επιβάλλονται σύνθετοι περιορισμοί ανά χρήστη, τόσο ως προς το πλήθος των επαναλαμβανόμενων συστάσεων ανά αντικείμενο όσο και ως προς το πλήθος των διαφορετικών αντικειμένων που μπορούν να προταθούν. Το πρόβλημα αυτό αποδεικνύεται ότι είναι υπολογιστικά δύσκολο, καθώς αποτελείται από ένα πρόβλημα που ανήκει στην κλάση προβλημάτων NP-hard με επιπλέον περιορισμούς. Για να αντιμετωπιστεί αυτή η υπολογιστική δυσκολία, το πρόβλημα χωρίζεται σε δύο υπο-προβλήματα τα οποία επιλύονται με άπληστους αλγορίθμους με τον συνδυασμό τους να επιτυγχάνει υψηλό σκορ συνάφειας, ενώ παράλληλα σέβεται τους περιορισμούς.

Με σκοπό την έγκαιρη λήψη δεδομένων από τους χρήστες, που οδηγεί στην αύξηση της ποιό-

τητας της εμπειρίας (ΠτΕ, Quality of Experience), αναπτύχθηκαν διάφορα σχήματα για την προσωρινή αποθήκευση δεδομένων στα άκρα του δικτύου, τα οποία χρησιμοποιούν περιορισμένο χώρο μνήμης σε συσκευές χρηστών. Η γνώση που αποκομίζεται από τη λειτουργία συστημάτων συστάσεων για τις προτιμήσεις κάθε χρήστη είναι χρήσιμη για την πρόβλεψη της ζήτησης κάθε αντικειμένου. Για να αποφασιστεί η βέλτιστη κατανομή περιεχομένου σε κάθε συσκευή χρήστη επιλύεται ένα πρόβλημα μεγιστοποίησης της ευστοχίας του αποθηκευμένου περιεχομένου. Στο πλαίσιο της διατριβής εξετάζεται διαφορετικό πλήθος συσκευών με δυνατότητα αποθήκευσης αλλά και διαφορετικές πολιτικές ως προς τον χρόνο αποθήκευσης. Εξετάζονται τόσο πρακτικές προκαταβολικής αποθήκευσης (proactive caching) όσο και δυναμικής (reactive caching). Τα αυξημένα ποσοστά καταδεικνύουν τα πλεονεκτήματα της χρήσης χώρου μνήμης από τις συσκευές των χρηστών καθώς και την ανάγκη μια συσκευή να λαμβάνει υπ' όψιν τα πιθανά αιτήματα των γειτονικών της χρηστών. Επιπλέον, εξετάστηκε ο συνδυασμός των συστάσεων που παρέχονται από ένα σύστημα συστάσεων με το ζήτημα της προσωρινής αποθήκευσης σε ορισμένους χρήστες, λαμβάνοντας υπόψιν την κινητικότητα των χρηστών εντός του υπό εξέταση χώρου. Η ΠτΕ θεωρείται συνάρτηση του χρόνου αναμονής του χρήστη και της συνάφειας των συστάσεων που του γίνονται. Το πρόβλημα μεγιστοποίησης της ΠτΕ μοντελοποιείται ως ένα πρόβλημα της κλάσης πολυπλοκότητας NP-hard και προτείνεται ένας άπληστος αλγόριθμος για την επίλυσή του, ο οποίος συγκρίνεται με προσεγγιστικό αλγόριθμο. Κατά τη σύγκριση των μεθόδων αναλύονται τα πλεονεκτήματα του προτεινόμενου αλγορίθμου ως προς τον χρόνο εκτέλεσης αλλά και την ποιότητα της ευρισκόμενης λύσης ως προς τη συνολική παραγόμενη ΠτΕ.

Στη συνέχεια, παρουσιάζονται συνοπτικά οι μεθοδολογίες, υπογραμμίζοντας τη συνεισφορά τους στη διατριβή. Έπειτα, κάθε Κεφάλαιο της διατριβής εστιάζει σε ένα από τα προαναφερθέντα προβλήματα, αναφέρονται συναφείς εργασίες στους αντίστοιχους τομείς και παρουσιάζονται λεπτομερώς οι προτεινόμενες λύσεις μαζί με ενδεικτικά αποτελέσματα που αποδεικνύουν τα πλεονεκτήματα που προκύπτουν από την υιοθέτησή τους.

Λέξεις Κλειδιά: Ανίχνευση κοινοτήτων, Κατανομή πόρων, Σύνθετα δίκτυα, Διάχυση πληροφορίας, Προσωρινή αποθήκευση στα άκρα δικτύου, Κοινωνικά δίκτυα, Συστήματα συστάσεων

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Contributions

Nowadays, a rapid expansion in the number of smart connected devices is observed that is expected to reach even new heights with the adoption of 5G communications worldwide. This next generation in telecommunications technology will make the implementation of the envisioned vast interconnected environments a reality. Moreover, Online Social Networks (OSNs), have established themselves as platforms that mediate the online communications of people, with the more popular ones retaining hundreds of millions of accounts. Their role reaches far beyond the simple message exchanging among friends, to being platforms where retailers advertise products, applications collaborate in order to provide their users with more personalized experiences, and in some cases, the OSNs even act as media companies via the publication of stories from their users. Examining these systems namely OSNs and smart device networks, as complex interdependent systems rather than separately, yields many benefits. This interdependency is evident, since, on the one hand, users of an OSN gain access, via the use of applications, to data generated by smart devices. The devices are also connected and form various kinds of relationships either explicitly (e.g., devices that exchange data) or implicitly (e.g., hidden similarities and correlations among device measurements). On the other hand, the OSNs can provide, through the gathering of data (e.g., location of users) or the opinions of their users, useful insights to the operators of the infrastructure. For example, the number of sensors needed in an area can be estimated by

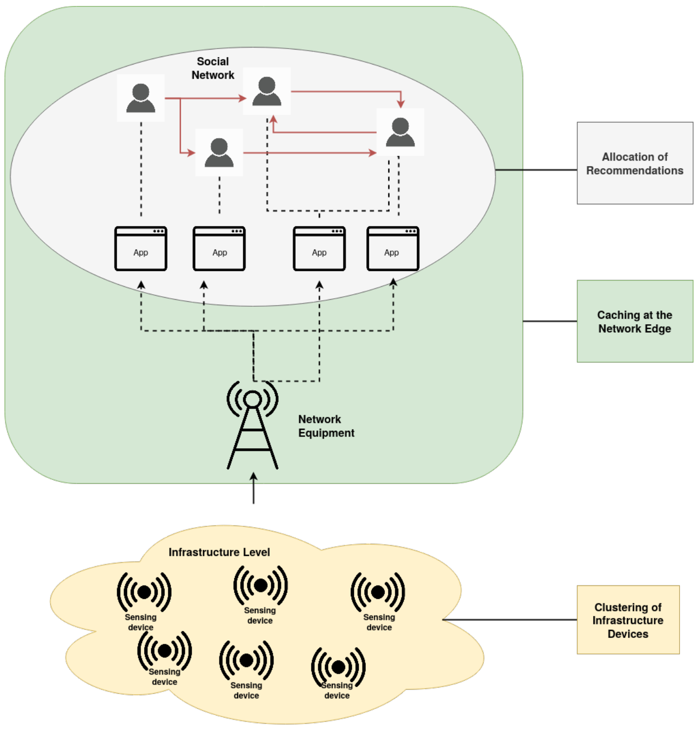

Figure 1.1: Overview of an Interdependent System

the number of people frequenting it. Moreover, these environments generate large scales of data, oftentimes characterized by the term *Big Data*, with predictions revealing further increase in the upcoming years. This vast amount of data poses challenges in its analysis, highlighting the need for developing novel, as well as enhancing traditional approaches in data handling. A common example of these environments is the Smart Cities concept. Smart Cities already exist and are expected to grow as indicated by investment forecasting on Smart City projects worldwide [2]. An overview of the envisioned environment for this thesis and of the employed techniques used for its analysis can be seen in Figure 1.1.

In this thesis, the focus lies on four areas, which are important in the operation of such large interconnected environments, namely, data clustering, community detection, recommendations and data caching at the network edge. In all of the above, crucial problems are identified and solved by developing tools and methods inspired by works in the field of social network analysis and network science. The aim is to develop approaches that can monitor networks of devices (e.g., sensor networks, smart device networks, etc.) and OSNs, infer hidden, underlying, similarities among entities of the system and leverage on them in order to discover relations that can be of use in prediction and fault tolerance scenarios. Also, by monitoring the information diffusion process in an OSN, influential actors are detected that contribute significantly in the information diffusion in the network in an efficient and unobtrusive manner. Moreover, by utilizing knowledge provided by system entities, such as recommender systems, multiple approaches are designed for suitable Device-to-Device (D2D) caching schemes. These schemes ensure increased cache hit ratios and increased quality of experience for the users by generating recommendation lists that nudge them to request the already cached content. In order to develop such approaches, a well-known divisive algorithm for community detection is modified incorporating aspects of hyperbolic network analysis. In addition, extra constraints that resemble real life situations are incorporated in problems of information diffusion. Because many of the problems addressed in this thesis are proven to be NP-hard, suitable heuristic algorithms that respect each problems' constraints and produce solutions of high quality are developed in order to acquire acceptable solutions in a timely manner for large datasets.

The main contributions of this thesis can be summarized as follows:

- At first, focusing on the infrastructure of an interconnected environment, namely the sensors that measure multiple kinds of data (e.g., weather related, energy consumption, etc.), the work presented in the thesis, aims at detecting groups of sensors that yield similar measurements over time. In order to infer these hidden similarities, the data clustering problem is mapped to a community detection one and solved by incorporating aspects of hyperbolic geometry. Each sensor is considered to measure a multitude of data. Each measurement is considered as a coordinate in a multi-dimensional custom space. In this way, every sensor is treated as a point of this space. Then, the full weighted graph is formed and a proximity graph is constructed by joining a suitable number of minimum spanning trees. This graph is then embedded in the hyperbolic space. Inspired by the well-known Girvan-Newman (GN) divisive community detection algorithm, edges are removed in batches until the network is split into communities that correspond to data clusters. The incorporation of the hyperbolic space speeds up calculations and makes possible the introduction of a novel approximative measure for the edge betweeenness centrality metric. The algorithm is tested on both artificial and real networks and it is proven to be faster than the traditional GN method, able to detect accurately clusters in datasets with known ground truth solutions, while also produce similar and in some cases better partitions in terms of the modularity metric compared to the original approach. Aiming to detect groups of similar people that have access to the sensor measurements via applications and are also part of an OSN, the algorithm is also tested on social networks. From the results, it is proven that it is able to detect groups of people (i.e., communities) that yield high modularity scores while resulting in partitions that are, in many cases, close to the ground truth, if such information is available.

- Moreover, the ability of the developed community detection algorithm to detect clusters in various types of datasets is highlighted via developing a community visualization application for RDF compliant datasets. Marking a step further from operating on spatial data, to the field of semantics, every semantic triplet is considered as a point in a custom space. Suitable distance metrics for every part of the triplet are employed and their weighted average is used for the formation of the proximity graphs. The developed application provides an elegant way for the graph visualization and functionalities allowing to zoom in to communities, or specific nodes. In the evaluation scenarios, the proposed algorithm is able to detect 100% correctly the triplets corresponding in each predicate of the examined dataset.

- In the envisioned interconnected environment, applications that combine information both from the infrastructure as well as the OSNs that the users participate in, need to be properly advertised. The identification of influential users in a network is crucial for any network operator as it provides answers, among other things, to the questions of *what* action (e.g., item, application, video file, etc.) to recommend to *which* user of the network. In order to combine the operation of a recommender system that inserts new information in the network through recommendations and the diffusion of this information among the users, Information Diffusion Aware Recommender Systems (IDARS) are studied. Within the IDARS framework, the problem of maximizing the total relevance of the suggestions to the users is examined alongside complex constraints regarding the number of different recommendations and the number of duplicate recommendations a user can receive without ruining her decision-making process by exposing her to too much information. This work is among the first where the aspect of respecting various users' information capacity thresholds is taken into account. The introduction of these constraints adds extra difficulty in the problem. In order to solve it, it is divided into two sub-problems that need to be solved in sequence, each one concerning the satisfaction of one constraint. The designed solutions, produce acceptable results that are able to achieve a high total relevance score, while guaranteeing that there are no violations of any kind.

- Aiming at increasing the users' quality of experience (QoE) as a function of the time needed for fetching their requested contents, knowledge about recommender systems and caching at the network edge are combined. Exploiting the interdependency among network equipment, smart devices and the social aspects of their users, information from recommender systems that operate in an OSN, is incorporated when designing content allocation schemes in order to infer users' reactions and behavior. Caching at the network edge ensures fast delivery of content to the users by eliminating the need to fetch multiple times the same content from the core network. Inspired by works on caching at the network edge [3, 4, 5], several D2D caching schemes are proposed, aiming to maximize the achieved cache hit ratio. In greater detail, assuming that the content and network operators consist of one entity (or that they closely collaborate), the preferences of users to applications is considered known. User Equipment (UE, i.e., smart devices) is scattered across a cell where a Base Station (BS) is responsible for their communication with the backhaul and the core network. Both the BS and the UEs have limited cache memory space that they can utilize to store information. The problem of allocating contents in cache memories is modeled, depending on the approach, as an appropriate version of the Knapsack Problem. Both proactive and reactive caching policies are developed and compared. Moreover, aiming to combine caching with the process of recommending items to users, the user's QoE is defined as a function of her relevance to the recommendations and the expected delay for fetching the requested content. The problem is formed as QoE maximization one, known to be NP-hard. In order to solve it, a heuristic method is developed which solves the content allocation and assigning of recommendations problems sequentially. The users' mobility patters are taken into consideration and the algorithm is compared to an approximative one, obtaining acceptable QoE scores in a more timely manner.

1.2 Outline

This thesis is organized as follows.

- **Chapter 2** provides concise descriptions of fundamental concepts and metrics from the field of complex network analysis that are essential for understanding the topics presented in this work.

- **Chapter 3** discusses the fields of community detection and data clustering and the importance of these in the analysis of large interconnected environments. In this chapter, a divisive community detection algorithm coupled with a graph database, is introduced that can be also applied for data clustering by a suitable mapping of data points to proximity graphs. Moreover, the incorporation of the developed community detection algorithm in an RDF data visualization application is highlighted.

- **Chapter 4** deals with the issue of information diffusion and information overloading in an OSN. More specifically, a socially-aware recommender system is developed that takes into consideration the users' preferences to the various items provided by a platform. In addition, by inferring the flow of information throughout the network, manages to avoid redundant recommendations that may negatively impact the users' experience.

- **Chapter 5** focuses on the concept of increasing the users' QoE via caching and recommendations at the network edge in order to assist in decongesting the infrastructure's links and spread faster the required information to the users. In greater detail, methods of caching data in the users' devices are discussed and evaluated. Also, a heuristic algorithm that besides selecting the appropriate UEs for caching, also generates recommendation lists is presented. The goal of the algorithm is to nudge the users in requesting some of the already appropriate cached content that will be relevant to their tastes and also reduce the delay needed for obtaining it.

- **Chapter 6** summarizes, in a concise manner, the problems addressed in this thesis, giving the reader a comprehensive overview of the most important conclusions drawn from this study. Then, it proceeds to suggest recommendations for future work that can be carried out as an extension of the work presented in the thesis.

Chapter 2

Network Theory Fundamentals

Nowadays, large interconnected environments such as those observed in Smart Cities, consist of a multitude of actors that communicate, exchange messages and generally form various kinds of relationships. For example, in online social networks (OSNs), common actor types are the people forming profiles in the platforms and the most common relationships are those formed among them, like friendships (e.g., Facebook) or follows (e.g., Twitter). Other common networks observed today, are sensor network topologies. In those, the sensors are the actors and the interactions among them indicate capabilities and relationships such as which sensors can exchange messages, measurements, act as gateways and so on. These kind of networks are called *complex networks*. The term "complex" is used to clarify that these networks evolve over time and can consist of different actor types that are related by a multitude of relationship types [6]. Graph theory is one of the most important fields of mathematics employed in the analysis of this kind of networks. Studying these networks using suitable metrics from graph theory, and more specifically, social network analysis, is crucial for discovering patterns, inferring the information flow among the actors of the system, predicting the network's evolution over time, discovering proving the manner in which it was formed, etc.

In order to deal with the key problems presented in this thesis, namely, those of clustering, community detection, recommendations and caching, complex networks are considered, in all cases, as the ideal form of representation of the relationships among actors. This decision is based on the fact that graphs are among the optimal tools in order to model the complex relationships formed in the large interconnected environments that this thesis focuses on. Even in the case of data clustering, the suggested approach is to map the data points to a complex network by considering the points as nodes of a graph and joining them with edges. Concepts and tools from the field of Social Network Analysis allow the study of the structural properties of the graph. Suitable metrics are employed in order to detect the underlying community structure in a graph as well as identifying the most important nodes and inferring the diffusion of information among the nodes of the network.

In this Chapter, some basic aspects of graph theory that are essential for understanding the discussed concepts, as well as for the development of the algorithms presented later in this work are presented and discussed. More in depth analysis of these topics and relative SNA metrics can be found in [6].

2.1 Elementary Aspects of Graph Theory

The most common notation of a graph is a triplet of the form $G(V, E)$, where G is the symbol of the graph, V is a non-empty set of elements that are named *nodes* and E is a set of tuples of elements in V. The set E contains the *edges* of the graph and an element in E is of the form (u, v) with $u, v \in V$. When an edge exists among two nodes, then, these nodes are called *neighbors*. When a graph is *undirected*, the order of the elements of an edge does not matter, i.e., (u, v) is the same edge as (v, u), whereas in a *directed* graph (u, v) and (v, u) denote two distinct edges. In the latter case, the nodes of an edge (u, v) are called starting point and endpoint respectively. A graph can be *weighted* if there is a countable quantity assigned to each edge. Although there exist weighted graphs in which the weights are assigned to nodes, the focus in this proposal is on the case where the edges are weighted when discussing weighted graphs, unless otherwise stated.

In a graph $G(V, E)$, every node $u \in V$ has a *neighborhood* $N(u)$. The neighborhood consists of all the nodes in V that are neighbors of u. In the case of directed graphs the neighborhood is divided in the *in-neighborhood* of a node $u \in V$, $N_{in}(u) = \{v; (v, u) \in E\}$ and the *out-neighborhood* of u, $N_{out}(u) = \{v; (u, v) \in E\}$. The *degree* of a node u in the undirected case, is the number of its neighbors, $|N(u)|$. In directed graphs, there is the *in-degree* of u, which is the number of nodes in the in-neighborhood of u, $|N_{in}(u)|$, and the *out-degree* of u, which is respectively, $|N_{out}(u)|$. In a weighted graph, the degree of a node is equal to the sum of the weights of the incident edges.

The simplest way in which a graph can be represented is the *adjacency matrix.* This matrix can be denoted as $A = [a_{ij}]_{N \times N}$. A is a square matrix of size $N \times N$, where $N = |V|$.

The elements of A, a_{ij}, are equal to

$$a_{ij} = \begin{cases} 1, & \text{if } (i,j) \in E, \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

There are more ways that a graph can be represented such as by an edge list, a dictionary, or a graph database. Similarly, in the case of a weighted graph, it can be represented by a matrix $W = [w_{ij}]_{N \times N}$, where the elements of W are equal to the weights of the corresponding edge, or zero in the elements that do not correspond to edges in E.

A *path* in G is a sequence of nodes $p = u_1 u_2 ... u_n$, where for any two consecutive nodes u_i, u_j there exists an edge $(i,j) \in E$. The *length* of path p is considered to be the number of edges lying inside it, or, otherwise stated, the number of *hops.* A path of minimum length between two nodes is called the *shortest path.* A path is called *simple* if no node appears more than once in the path. A path in which nodes can appear more than one time contains *cycles.* A cycle is a path in which the starting and ending node are the same. A graph that contains no cycles is called a *tree.* An undirected graph in which there exists a path joining every pair of nodes is called *connected.* In the case of directed graphs, if every pair of nodes is connected by a path, the graph is called *strongly connected.* An edge whose removal leads the graph from being connected to become disconnected is called a *bridge.* In a disconnected graph the maximal groups of nodes for which paths exist among them are called *connected components.*

A subgraph of $G(V,E)$ is a graph $G'(V',E')$ with $V' \subseteq V$ and $E' \subseteq E$. In every graph G there exist subgraphs of the form $T(V,E')$ that contain all the nodes of G and are trees. This graph is called a *spanning tree.* In a weighted graph with weights on the edges, the spanning tree $T(V,E')$ for which it holds

$$\sum_{e \in E'} w_e \leq \sum_{e \in E''} w_e, \qquad \forall T'(V,E''), \tag{2.2}$$

is called the *minimum spanning tree* (MST) of G. The most popular algorithms for discovering the minimum spanning trees are those of Prim [7] and Kruskal [8]. In the following sections some basic metrics of social network analysis are introduced, as well as some basic complex network models.

2.2 Basic SNA metrics

2.2.1 Degree Distribution

As mentioned earlier, each node has a degree, which, in the case of undirected and unweighted graphs, is equal to the number of its neighbors. The degree of each node depends on the manner the graph was formed. For example, there are networks where all nodes have the same degree, while on other cases the neighbors of a node can be dependent on a multitude of criteria like the distance among nodes, node popularity, etc. The nodes in a graph can also be joined either in a deterministic or probabilistic way. In all cases, examining the degree distribution of a network provides useful information about its structure and can aid in revealing the manner in which it was formed. Visualizing the degree distribution of graphs, either as a histogram or cumulatively, is one of the most common ways to identify network types and assists in their analysis. For example, one can easily identify a scale-free network by observing the power-law degree distribution of its nodes, while it is also trivial to identify a regular graph from its Dirac like histogram.

2.2.2 Average Path Length

The average path length is a network-wide metric and it is the average of the length of all the shortest paths that join all the pairs of nodes in the graph. It is a metric that reveals the expected distance between any pair of nodes inside such a graph and an indication of how fast information can flow inside the network. A large value indicates a network in which information can reach quickly the majority of nodes, while a large average path length is characteristic of networks in which the messages travel at a slower pace in the network. In this thesis, the path length is denoted as the number of hops between nodes.

2.2.3 Centrality measures

Among the many metrics that have been defined in order to measure the “importance” of nodes inside the network, the centralities are one of the most widely used. There have been many centrality measures proposed in the bibliography like betweenness centrality [9], closeness centrality [9], eigenvector centrality [9], Katz centrality [10], PageRank centrality [11], etc. In this work though, the degree centrality and the edge betweenness centrality will be more thoroughly explained since they are those that are employed in the following chapters.

The degree centrality is the simplest centrality measure and is equal to the node’s degree. The higher the degree, the more central the node is considered. Although usually it is a good indicator of the node’s importance, it can be misleading in some cases. For example, consider the following network of Figure 2.1. Node 4, despite having the minimum degree centrality in the graph, equal to 2, is a very important node in terms of information diffusion since a fault in its operation would result in the graph becoming disconnected. Evaluating the degree centrality allows for the successful selection of nodes that are directly connected with many others. Oftentimes, a relatively small subset of such nodes is able to cover the whole network (i.e., the nodes of this set and the union of their neighbors equals to the whole network). The detection of sets of nodes with such properties is a vital step in the community detection and also the content caching algorithms described in the following Chapters.

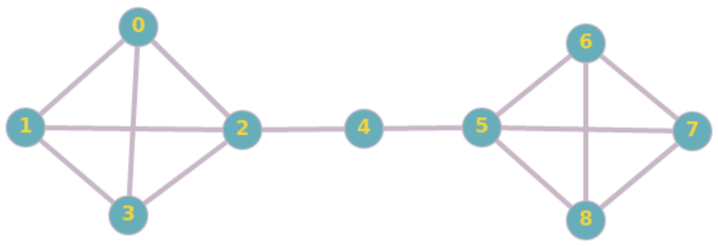

Figure 2.1: Example graph displaying the importance of considering various centrality metrics.

The edge betweenness centrality (EBC) of an edge of the graph is a metric that denotes the percentage of shortest paths among all pairs of nodes that pass through the edge. It is the equivalent of the betweenness centrality of a node. An edge that displays high edge betweenness centrality displays bridge-like properties and its removal is expected to increase the average path length or even deem the network disconnected. Because the computation of every shortest path in the graph is a time consuming procedure, an algorithm has been proposed by Brandes [12] that calculates the EBC of the edges without having to explicitly calculate the paths among all pairs of nodes.

Identifying the most central edges in a graph allows for the detection of groups of nodes that tend to interact more among them than with the rest of the nodes with whom are joined with a few edges that due to their bridge-like status display high EBC scores. The calculation of the edge betweenness centrality is a core component of the Girvan-Newman algorithm [13] as well as the proposed algorithm for community detection that will be presented in Chapter 3.

2.3 Common Types of Complex Network Topologies

In order to classify complex networks as well as to study the evolution of systems over time, many artificial network generation models have been defined. The most common of which are the *Regular graph* in which all the nodes of the graph have the same degree, the *Gilbert random graph* [14], $G(n, p)$, having n nodes and where each possible edge exists (or not) in the graph with probability p, the *Erdos-Renyi random graph* [15], $G(n, M)$, where a graph out of all the possible graphs containing n nodes and M edges is chosen at random. In this section, the graph models that have been employed in the analysis of the proposed framework are described in more detail. Such networks are employed throughout this proposal in order to evaluate the proposed algorithms and study their behavior in different types of networks.

2.3.1 Random Geometric Graphs

A Random Geometric Graph (RGG), $G(n, r)$, is a spatial graph model, meaning that the edges of the graph are dependent on the positions of the nodes in a metric space. An RGG graph is defined by the number of nodes n, and a radius r, and it is structured as follows:

- Select uniformly at random from the unitary space $\Omega \in R^d$ the positions of the n nodes.

- Join all pairs of nodes u, v whose distance is $dist(u, v) < r$.

In sensor networks and telecommunications it is common to simulate positions of sensors or mobile devices with RGGs in the 2-dimensional Euclidean space and the distance between nodes is commonly the Euclidean distance on R^2. However, it is possible to use other metrics, such as the Manhattan distance, which is oftentimes used to model mobility networks [16].

The expected number of neighbors for each node of an RGG graph is equal to $\frac{\pi \cdot r^2}{R^2} \cdot n$. Moreover, it is proven that the average clustering coefficient of RGGs, for large values of n, is $cc_{net} \simeq 0.59$ [17]. This fact indicates relatively strong triadic closure for this kind of graphs.

In this thesis, RGGs are employed as sensor networks for the evaluation of the community detection algorithm described in Chapter 3 as well as graphs representing the locations of users in a telecommunications cell for the purposes of content caching, discussed in Chapter 5.

2.3.2 Small-World Graphs

A small-world network refers to a growing graph whose average path length increases proportionally to the logarithm of the number of network nodes. Its properties lie between those of a regular graph and those of a random graph. This fact results in exhibiting the favorable characteristics of high clustering coefficient and short average path length.

Watts and Strogatz proposed a model in order to build small world graphs in [18]. In this model, initially a regular graph is formed. Then, with a user-defined probability p, an edge is "rewired", meaning that it changes its original endpoint. These edges now form shortcuts that join previously distant nodes, thus decreasing the average path length of the graph, while the original regular graph structure (as long as it is retained in some extent) is responsible for the high clustering coefficient observed in small-world graphs. Of course, as it can be seen in Figure 2.2, as the probability p of rewiring edges increases the graph tends to transform into a completely random graph.

Small-world graphs are employed as models that are able to simulate social relationships in OSNs and are used for the evaluation of the community detection algorithm, while directed small-world networks are used for recommendations allocation in Chapter 4.

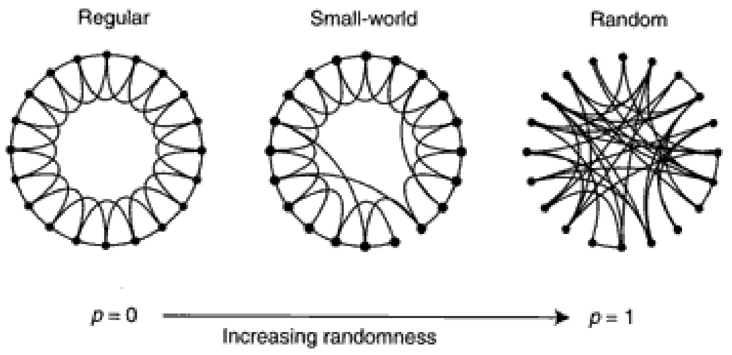

Figure 2.2: Evolution of a graph from regular to random following the Watts-Strogatz model. [1]

2.3.3 Scale-free Graphs

The concept of "scale-free" in this kind of graphs implies the lack of scaling in the degree distribution. The degree distribution of these graphs follows a power-law distribution. Many existing networks can be classified as scale-free, such as biological (e.g., gene interaction networks) and technological networks (i.e, circuit networks, water transportation networks, etc) [19].

The scale-free property displayed by this kind of networks is due to two mechanisms that contribute to the formation of the power-law degree distribution. The first one is the sequential arrival of nodes over each time step. Initially, the graph contains only a small clique of nodes and eventually it reaches the total number of nodes, n. The other is the preferential attachment rule, according to which, a newly added node will prefer to connect to nodes that display high popularity (i.e., have a large degree). This leads to older nodes having largest probabilities to have many neighbors. In this way, just a few nodes display a large degree, while others have relatively low degrees and this explains the power-law degree distribution. These high degree nodes act as hubs and are responsible for the low average path length observed in these networks.

There exist various ways in which preferential attachment can be applied [20]. In this thesis the Barabasi-Albert model (BA) [20] is employed every time Scale-free graphs need to be generated. In greater detail, the BA model operates as follows:

- For every node currently in the graph at time step t, every node u has a degree $k_u(t)$ and a probability to be selected as a neighbor of a new node given by $P(k_u(t)) = \frac{k_u(t)}{\sum_{v \in V} k_v(t)}$.

- At time step $t + 1$ a new node enters the graph and it is joined with m nodes chosen according to the probabilities of the previous step.

A very interesting aspect of scale-free networks is that they display a finite negative curvature [21] revealing their association with the hyperbolic metric space. In [22], the authors propose a model of generating random graphs in the two dimensional hyperbolic plane H^2 of radius R. In this model, nodes are assigned an angular coordinate at random in $[0, 2\pi]$ and a radial coordinate r, following the density $p(r) = e^{r-R}$. It is proven that by joining the nodes that are at a hyperbolic distance less than R, the resulting network is scale-free since its nodes follow a power-law degree distribution. The concept of the underlying hyperbolic geometry of scale-free networks will be further discussed in Chapter 3.

Chapter 3

Efficient Detection of Communities in Large Networks

Large interconnected environments consist of a multitude of components like smart devices (e.g., sensors) and also people as part of OSNs. Such environments exhibit several challenges in their analysis. It is expected, and shown in the following, that in a smart connected environment, there exist similar sensors that yield similar measurements over time. For example, office areas throughout the same city are expected to have similar measurements of temperature. The problem of data clustering of sensor network measurements can be tackled by mapping it into a community detection one.

Community detection is one of the most fundamental problems in complex networks analysis [23]. It is used in order to extract useful information about their structure and the relationships between nodes. As complex networks, either social networks or spatial networks, evolve through time, communities of nodes are formed [24]. A community is a group of nodes that share among them more similar attributes than with nodes belonging in different communities.

Community detection, or network clustering, refers to the algorithmic process of finding groups of nodes that satisfy certain properties. Although the field of studying complex networks is already some decades old, there is no consensus as to what exactly is a community and many researchers define it differently [25]. As a result there exist many methods for community detection. For example, some aim to maximize the internal density of modules [26, 27], while others detect communities based on flows [28]. Moreover, nodes may be part of more than one communities (overlapping communities) [29] or belong to exactly one. It is a fact that there does not exist a single best community detection algorithm and that there will never exist a single general purpose algorithm for all networks and all data types [30]. A consequence of the above, is the existence of multiple different evaluation techniques regarding the evaluation of a network partitioning in communities [31].

In this work, the notion employed for the communities is that, inside a network, nodes are more likely to connect with other nodes belonging in the same community than with nodes belonging in different ones. This approach is similar to what is referred in [25] as maximization of density.

The networks obtained via the mapping process of a data clustering problem to a community detection one are likely to display discernible communities. In particular, it is expected that there exist strongly connected groups of nodes (i.e., corresponding to clusters of data) that are joined together with a few edges displaying bridge-like properties and high Edge Betweenness Centrality (EBC) scores. It is considered that a community detection algorithm that operates by removing such edges, is able to detect communities, and thus, clusters, that are meaningful. The original Girvan-Newman algorithm acts as the inspiration and starting point for developing a novel community detection algorithm that incorporates key ideas of the original approach. However, by using aspects of hyperbolic geometry and also the removal of edges in batches in order to speed up calculations, it consists of a more scalable approach. Aiming to facilitate the operation of the algorithm for large datasets a graph database is also incorporated.

In this Chapter, different techniques for community detection will be discussed. Then, the developed community detection algorithm, HGN, will be presented alongside a detailed evaluation on both synthetic and real data that are obtained through experiments conducted in a real smart city in Spain, Smart Santander. Finally, the ability of the HGN algorithm to detect semantic communities in RDF datasets is highlighted through the demonstration of a graph visualization application.

3.1 Related Works on Community Detection

In the literature there exist multiple techniques in order to partition the network into communities. Although each algorithm addresses uniquely the problem of community detection, the algorithms can be classified in certain broader categories according to the manner in which they partition the graph. In the following, some of the major categories of community detection techniques, alongside with notable works, are presented and discussed.

3.1.1 Hierarchical and Divisive Clustering

In the hierarchical network clustering, the discovery of communities forms a partition where larger clusters, examined in greater detail, reveal smaller clusters of nodes, belonging to lower levels of the hierarchy. The algorithms that are included in this category are very useful in order to discover communities in networks that naturally display a hierarchy among their nodes (e.g., social networks, computer networks, etc). The construction of the community hierarchy is done either bottom-up (agglomerative) or top-down (divisive).

In the agglomerative community detection techniques, initially all nodes are considered as single-node communities. Proper similarity metrics, either on vertex or community level, are defined and communities are joined iteratively as long as their similarity exceeds a certain threshold.

On the other hand, in the divisive node clustering techniques, initially, the whole network is considered as a single community. Then, it is partitioned in smaller communities. This is usually accomplished by removing edges displaying low similarity scores.

For both of the above approaches, the function that calculates the similarity defines the obtained partition. One of the most well-known hierarchical clustering algorithms that uses a structural metric in order to split the network into communities is the *Girvan-Newman* algorithm [13]. Girvan-Newman (GN) is a divisive community detection algorithm. The main idea of this algorithm is that most of the network edges join nodes that belong in the same community and only a small percentage of them connects nodes that belong in different communities and act as “bridges”. To find such central edges, the notion of a network metric called Edge Betweenness Centrality (EBC) is employed. This metric evaluates the percentage of the shortest paths that cross the edge. A high EBC score implies a central edge whose removal may lead to the graph becoming disconnected (i.e., a community is detected). Girvan-Newman removes each time the edge ranking highest in the EBC metric until either the whole network is split into single-node communities or until a pre-specified number of communities has been detected. Because the computation of the EBC metric is rather costly some variations of the traditional GN algorithm have been developed in order for the approach to scale for larger graph sizes. In [32] two variants of the GN algorithm are presented. The first one utilizes the Map-Reduce algorithm on the well-known Hadoop platoform, while the second one uses GraphChi [33] in a vertex-centric approach. Even though the proposed parallel algorithms manage to reduce the execution time compared to the original algorithm, average sized datasets still require significant amount of time, in the order of minutes, to complete their execution. Another divisive approach, similar to GN, is the one presented in [34], where, instead of removing the edge of highest EBC value as in the case of Girvan-Newman, the edge of the highest information centrality is removed. This algorithm is able to find meaningful communities, but suffers from a high time complexity of $O(|E|^3|V|)$. This high complexity marks it as an unsuitable algorithm for modern-day large scale networks but proves the relevance of the influence metric on the formation of communities. Moreover, in [35] the authors propose another variation on the GN algorithm by removing multiple edges at a time based on the line graph of the original network. In greater detail, it removes edges that have the highest betweenness centrality as nodes of the line graph. This algorithm terminates faster than GN and at some cases produces greater modularity scores, although on rather small networks. Capitalizing on the metric of clustering coefficient for every node of the graph, the authors in [36] propose a divisive community algorithm (not hierarchical though), that removes an edge according to a rule based on the clustering coefficient between its endpoints. If the removal of the edge leads to an increase in the clustering coefficient of both nodes, then the edge is an inter-community edge that should be removed. The algorithm keeps removing edges, until no edges can be deleted based on the described criterion. This algorithm produces good modularity scores but as the authors mention it fails to detect meaningful structure in star-graph topologies and its execution is unstable and dependent on the order of the edges' examination. Another divisive algorithm is presented in [37], here the concept of "weak links" is introduced based on the topological attributes of the nodes. In each iteration of the algorithm, the weak links are calculated and removed until no more such links can be detected. The authors claim a better accuracy-efficiency compared to other divisive methods.

3.1.2 Spectral Clustering

Spectral Clustering in networks is performed by using a matrix representation of the graph. This matrix can be weighted and can either be the adjacency matrix of the graph or the Laplacian graph, although there is not a single definition of this matrix [38]. In these techniques the number of clusters (k) is considered known either beforehand or it is estimated via various techniques. The first k eigenvectors with the smallest eigenvalues are obtained and their values are employed as coordinates in a k-dimensional metric space. The eigenvalues of the Laplacian matrix provide information about the connected components and the density of the network. Then, clustering algorithms, like k-means [39], are employed on the points and the obtained clusters correspond to the detected communities.

The simplest method of such a clustering approach is the non-normalized spectral clustering in which the Laplacian is defined as $L = D - W$. D is a diagonal matrix containing the degree of each node and W is the weighted adjacency matrix. Then, the eigenvalues and the eigenvectors of this matrix are calculated. In the normalized version proposed in [40] the Laplacian is again considered to be equal to $L = D - W$, but the eigenvectors and eigenvalues are obtained from the solution of the generalized problem $Lu = \lambda Du$, where u are the eigenvectors and λ the corresponding eigenvalues.

Based on the aforementioned notions, there exist more recent approaches in Spectral Clustering like the one presented in [41]. The authors propose the use of the Bethe-Hessian matrix instead of the Laplacian and claim that the eigenvectors corresponding to negative eigenvalues indicate the number of underlying communities in the graph. In that way, the need for prior knowledge of the number of communities is eliminated.

Spectral clustering methods are typically fast and achieve dimensionality reduction when the nodes of the graph contain multiple attributes. Also, these algorithms can achieve better results on spatial datasets than many traditional clustering algorithms, such as k-means, because they are able to detect clusters of non spherical shape. Despite these facts, Spectral Clustering tends to split networks into communities that vary greatly in terms of their size. Moreover, the optimal selection of the proper similarity matrix, or the Laplacian, is oftentimes cumbersome or beyond the users' knowledge [42].

3.1.3 Modularity Based Methods

Modularity is a metric that evaluates a network partition based on the amount of edges that join nodes belonging in the same community compared to a random distribution of edges among these nodes. The modularity metric that is obtained from a partition of the graph in k communities can be calculated as follows:

$$Q = \frac{1}{2m} \sum_{l=1}^{k} \sum_{i,j \in C_l} (A_{ij} - \frac{d_i d_j}{2m}), \tag{3.1}$$

where m is the total number of edges of the graph, A is the adjacency matrix, d_i, d_j are the degrees of nodes i and j respectively and C_l is the set of nodes belonging in the l-th community. Modularity values range from -1 to 1 and the variation from 0 indicates the difference with a completely random partition.

The target in modularity based methods is the maximization of this metric. The discovery of the optimal partition that results in the maximum modularity is a known NP complete problem [43]. This means that it cannot be solved in a reasonable amount of time, especially for large graphs with many nodes and edges. This fact has led to the development of numerous techniques in order to produce a sub-optimal solution, close to the optimal one. Most modularity based methods are, in fact, hierarchical clustering approaches but they are wider known as *modularity maximization* techniques, and thus, they are presented separately.

The first approach in this direction is the algorithm proposed in the paper of Newman [44], and essentially, is an agglomerative method. Each node is initially considered as a single community and then, the edges are inserted in a way that leads to the greatest increase in the modularity in each step. Then, pairs of communities with existing edges among them are examined as to whether merging them would result in an increase in modularity. If the modularity is increased, then, the communities are merged. The required computations of this algorithm mark it as rather slow approach for this type of community detection algorithms. Its time complexity is $O(|V|^2)$, with $|V|$ being the number of nodes. In [27], the authors discover redundant calculations in the method of Newman and thus reduce the complexity of the algorithm. Another approach widely used is the algorithm of Blondel [26], also known as the *Louvain algorithm.* According to the algorithm, each node starts as a single-node community. Then, iteratively pairs of communities are examined and if their merging leads to greater modularity scores, the communities are merged and replaced by "supernodes". Eventually, the modularity will not be able to increase anymore and the algorithm terminates. This algorithm is very efficient as it has a time complexity of just $O(m)$, with m being the number of edges in the graph.

Aiming to develop a modularity optimization method that can be executed in dynamic graphs, in [45], the authors propose the execution of the Louvain method for the whole graph only for the first snapshot of the network. For the rest of the snapshots it performs a local modularity optimization process only for those communities where changes occur (i.e., nodes/edges are deleted or added) between successive snapshots. This algorithm, named *Dynamic Louvain* was able to obtain communities resulting in similar modularity scores with Louvain but a lot faster, proving that reusing community structure from previous snapshots is a viable solution. In addition to the former method, the authors of [46] also propose a variation on Louvain that can be employed for dynamic networks. In this method, called C-Blondel, the community detection method for a snapshot t is done by executing the Louvain algorithm on a compressed graph. The construction of this graph is based on the previous snapshot $t-1$ and the possible differences between t and $t-1$ in terms of the removal or addition of nodes and edges.

Although modularity is one of the most widely accepted metrics regarding the evaluation of community detection algorithms, it is noted in the literature [47, 48] that many of the modularity maximization algorithms tend to favor larger sizes of communities. In order to address this problem, in [49], the authors propose the employment of a weighted scheme on the edges. The incorporation of this weight is done in order to avoid the development of rather large communities that lead to imbalanced partitions having few very large communities and many small ones. Moreover, another drawback of these methods is that by trying to optimize the modularity metric they cannot always detect the ground truth partition, if it is available. Also, there exist partitions of completely random graphs that display high modularity scores despite the fact that in reality no meaningful groups are formed, as highlighted in [50].

3.1.4 Information Flow based Community Detection

Another approach to detecting communities in graphs is by taking into account the information diffusion inside the network. The study of the information flow can reveal the system's behavior. It is thus expected, that groups of nodes between which information flows quickly can be considered as a community.

A well-known algorithm based on the concept of random walks is the Markov Cluster algorithm (MCL) [28]. The *MCL* algorithm computes the probabilities of random walks inside the graph by employing two operators iteratively, namely inflation and expansion. Because of the underlying cluster structure, even when the exact communities are not known on beforehand, these operators allow for the detection of meaningful communities. Nodes that belong inside each community have large probabilities of reaching one another with random walks of certain length. The algorithm terminates when there are no significant changes in these probabilities for a pre-specified number of consecutive iterations. The MCL algorithm is fast and does not require prior knowledge on the number of communities. Variations of the original algorithm have been widely used to cluster protein networks into families [51, 52]. The authors in [53] propose a parallel implementation of MCL that scales well for networks consisting of million of nodes and billions of edges. The original algorithm cannot succeed in finding communities in reasonable time for such large graphs.

One of the most popular algorithms of this category is *Infomap*, presented in [54]. It is proven that minimizing the description length (e.g., Huffman coding) of a random walker is equivalent to finding community structure in networks. Infomap monitors random walkers in the graph, based on the notion that a random walker is more likely to remain in the same community than changing communities by crossing inter-community edges. Infomap optimizes the map equation [55] aiming to detect communities in the network. During the execution of Infomap, initially each node is considered as a module. Then, each node, in a random order, is assigned to the module that results in the largest decrease in the map equation. The modules discovered in a previous step of the algorithm act as nodes in the next one. The process is repeated until there does not exist a move that leads to a further decrease in the map equation. Infomap is both swift and able to find meaningful communities in very large networks.

Other recent random walk-based community detection methods include *GEMSEC* presented in [56]. In this article, the authors propose an algorithm that embeds the network into the Euclidean space and at the same time performs clustering of the nodes while employing first and second order random walks for node sampling. GEMSEC manages to accomplish greater modularity scores than other similar algorithms on common baseline graphs. In *FluidC* [57], standing for "fluid communities", the number of communities (fluids) is pre-specified by the user. Each fluid begins at a vertex chosen at random. At each step (named superstep) all nodes are examined and assigned to the community that results in the greater increase in density. When for two consecutive iterations the nodes do not change their assigned communities, the algorithm completes its execution. FluidC manages to mitigate the problem that many community detection algorithms face, namely the oftentimes imbalanced sizes of communities, meaning having a giant community surrounded by numerous small ones. However, it fails at achieving optimal solutions when there are few inter community edges.

3.1.5 Label Propagation Methods

The main idea behind Label Propagation algorithms is that nodes are assigned labels, exchange that information with their neighbors and possibly alter their own label according to each algorithm's criteria. One of the first such algorithms is *LPA* [58]. In LPA, each node is initially assigned a unique label. At each step of the algorithm, each node chooses whether to modify its label according to the dominant label in its neighborhood, with ties broken randomly. For better results, an asynchronous scheme is preferred where nodes change (or not) their label according to the labels assigned in the previous step. After a number of steps the algorithm converges and nodes stop changing labels. The algorithm has a competent execution time, with $O(|V| + |E|)$ complexity, but it does not guarantee convergence to the same solution for the same input graph. To alleviate this issue, the authors suggest aggregation of the different solutions to produce the final community structure. In order to improve the execution time and quality of the produced graph partition, in [59] an enhanced algorithm, called *SLPA*, is introduced. In this algorithm, nodes are divided in two possible states, namely, active (i.e., likely to change label in the next iteration) and passive. This algorithm needs fewer iterations than the original LPA and produces, in many cases, communities that result in higher modularity scores. The authors in [60] propose a variation of a label propagation technique called *ISLPA* that is able to be executed in networks that display incremental changes over time. Their evaluation proved ISLPA to be about two times faster than SLPA while producing similar modularity scores. The authors in [61] also propose a modification of the traditional LPA algorithm in which each node's influence (reflecting its structural position inside the network) defines the degree that its label will be adopted by its neighbors. Another label propagation method for community detection is the one presented in [62]. There, the authors propose the identification of two types of nodes in the network. The first one is the core node. Around these nodes communities are formed. A core node should have multiple neighbors that are also connected among them. The second type of node is the boundary node that is a node that is connected with nodes belonging to more than one communities. An iterative process is followed until all core and boundary nodes have been identified. Then, the boundary nodes are assigned to a community and the algorithm terminates. While subtly slower than LPA, this algorithm produces more stable communities for multiple executions on the same input. Moreover, in [63], the authors propose a method inspired by the LPA algorithm. Their method, named *EILPA*, capitalizes on first finding a set of influential nodes, and then by performing an extension phase is similar to LPA, discover communities. The difference with the traditional LPA method is that the most influential nodes determine greatly the choice of labels for the other nodes. This method outperforms LPA in terms of the achieved NMI scores with the ground truth on real and synthetic datasets. The authors in [64] propose another variation on LPA called *LPA-S*, which is a parallelized version of LPA in which nodes update synchronously their labels. LPA-S displays similar results with LPA. Moving towards the clustering of nodes in attributed networks, in [65], the *NMLPA* algorithm is proposed. This is an algorithm for overlapping community detection (i.e, a node may belong to more than one communities). In its process it transforms the attributes into weights of edges in a weighted graph. The weights reflect the similarities between the nodes. Then, it proceeds to perform multi-label propagation. This algorithm outperforms other relevant approaches in networks with known ground truth.

The authors in [66] propose *LabelRank*, an algorithm inspired by the MCL approach. Instead of each node having just one label, as was the case in the previous algorithms, LabelRank maintains an entire distribution of labels that define the probability of the node to be assigned each one of them. Across consecutive iterations, nodes exchange information about their distributions and update their own accordingly if there are significant differences with that of their neighbors'. When just a few changes in label distributions across all nodes occur, LabelRank terminates. LabelRank solves the issue of multiple possible partitions for the same graph while being faster than LPA with just $O(|E|)$ complexity.

3.1.6 Leader-based methods

In these methods, leader nodes that are able to influence others in the network are detected. In many of these algorithms, the structural position of a node in the network (e.g., its degree centrality) plays a crucial role in its definition as a leader node. In [67] the authors propose a solution to the community detection problem where each node selects (prefers) another node and they are put together in the same community. Multiple preference metrics have been evaluated as to their resulting accuracy to the ground truth solutions with the Jaccard similarity and number of common neighbors to emerge as the best choices for this algorithm. *LeadersRank*, presented in [68], relies heavily on the notion of leader nodes residing in the network. The nodes in the graph are ordered in decreasing order according to their eigenvector centrality. The most central node is the leader of a community that is formed by including all similar nodes of the leader to it. This process is resumed until all nodes are assigned to communities. In [69] the authors suggest two variations on the LeadersRank algorithm that display improved performance in benchmark networks. *LGIEM* presented in [70], introduces a new node metric based on local and global attributes, named LGI in order to distinguish the k most influential nodes and then follows an expansion strategy to detect communities. This algorithm can detect meaningful communities but the number of them, k, must be known in advance.

In the following section, the developed community detection algorithm will be presented in detail and evaluation results will showcase its strong aspects.

3.2 Hyperbolic Girvan-Newman

In order to efficiently cluster spatial datasets, discover meaningful relationships between devices that form the infrastructure of an interconnected environment and also detect communities in OSNs, a novel community detection algorithm is proposed. The Hyperbolic Girvan-Newman (HGN) algorithm capitalizes on the strong aspects displayed by the original GN algorithm discussed previously. Namely, its ability to detect clusters of high quality in networks where communities are strongly present. Networks generated from spatial datasets, such those that are created by linking similar measurements from sensor networks deployed in a Smart City, are expected to display community structure. HGN differs from GN in the following manners. First of all, it employs an embedding to the hyperbolic metric space which is a geometric space that follows the principles of hyperbolic geometry. Also, it evaluates the centrality of the edges with an approximative metric. Moreover, instead of just removing a single edge at a time, HGN removes edges in batches, until the required number of communities is discovered. In the following subsections the algorithm's operation will be described in greater detail.

3.2.1 Network Embedding

Network embedding refers to the process of assigning coordinates of a metric space to every node of the network. Embedding the network in a low dimensional space can aid in accomplishing many computations more efficiently. The hyperbolic space is proven to be a suitable space for embedding large complex networks. This is due to the fact that such graphs, especially networks that display significant power-law degree distribution, have been conjectured to have an underlying hyperbolic geometry [71]. In the literature there exist multiple embedding algorithms [72, 73, 74, 75, 76], aiming to either assist in understanding the process of the network's evolution or manage to assign coordinates that reflect an underlying property of the graph. Many of these embedding algorithms are also helpful in making certain computations less complex.

The Popularity-Similarity (PS) model [77, 73, 78] is one of the most common models for embedding graphs in the hyperbolic space. Every node is considered to be assigned two polar coordinates in hyperbolic space, with the two-dimensional H^2 space usually being the common choice. The radial coordinate abstracts the node's popularity, with nodes that lie closer to the center of the hyperbolic disk being the most popular ones. The angular coordinate is used to abstract the similarity between nodes. In the PS model the hyperbolic distance between two nodes can be regarded as the probability that they are connected in the network (or that they will be in the future). In [77] it is proven that networks that follow the preferential attachment rule for selecting each new node's neighbors emerge naturally from the PS model under a condition. This condition is that each new node that joins the network is connected with a subset of nodes, so that the product between the popularity and similarity is optimized. Moving towards the opposite direction, a network that displays the characteristics of the preferential attachment model can be embedded in the hyperbolic space by estimating for every node its polar coordinates through a Markov Chain Monte Carlo (MCMC) process.

Extending this work, authors in [73] propose *HyperMap*. HyperMap utilizes the PS model and assigns hyperbolic polar coordinates to every node of the network by sorting the nodes in decreasing order of their degree centrality. Then, the first node is given a zero radial coordinate and a random angular between $[0, 2\pi]$. Following that, each one of the remaining nodes, examined in order of appearance (i.e., ranking in degree centrality), is assigned a radial coordinate that depends on the degree of the node. All the previous nodes recalibrate their assigned radial coordinates. The node is also assigned an angular coordinate, which is the result of a Maximum Likelihood Estimation (MLE) process. HyperMap can be used for link prediction applications as well as greedy routing protocols. In order to speed up the computations required for embedding a network with HyperMap, the authors in [78] propose the estimation of the angular coordinates for a subset of nodes based only on those of their neighbors, without having to solve the corresponding MLE problems. In [72] the authors propose *LaBNE*, aiming at enhancing link prediction techniques. In this algorithm the network is embedded in the hyperbolic space H^2 using the Laplacian matrix L, obtained from $L = D - A$, with D being the degree matrix and A the adjacency matrix. The coordinates of each node are obtained by first embedding the graph in H^2 by solving a linear system and then assigning them radial coordinates that depend on their degree centrality (in a similar fashion as HyperMap) and angular coordinates dependent on the hyperbolic embedding coordinates. Another PS embedding approach is the one developed in [74] where the authors propose *Mercator*. Mercator is a method mixing machine learning and maximum likelihood techniques in order to produce a meaningful embedding regardless of the degree distribution of the graph. It is also based on the Popularity-Similarity (PS) model both on the Euclidean and Hyperbolic space (i.e., S^1/H^2, where S^1 is the one dimensional sphere and H^2 the two-dimensional hyperbolic space). Moreover, hyperbolic embedding can also assist in routing applications.

In [75] a greedy hyperbolic embedding is proposed, where nodes are assigned coordinates in the hyperbolic space by embedding a minimal depth spanning tree of the graph. Such an embedding, guarantees that greedy routing is 100% successful in the embedded network. Moreover, in the same work, an online embedding algorithm is proposed that is capable of adjusting the nodes' coordinates in the events of link failures or new node arrivals.

In HGN, the graph is embedded in the hyperbolic space using the distance preserving *Rigel* embedding algorithm [76]. This embedding algorithm operates as follows. First of all, a small percentage of the nodes, named landmarks, are selected. These landmarks are the nodes with the highest degree, although different strategies for landmark selection are possible and have been utilized in relevant algorithms [79]. The landmark nodes are embedded in the hyperbolic space so that the hyperbolic distance between them is as close as possible to their shortest path length distance in the network. Following this, the rest of the nodes are assigned coordinates such that their distance to a randomly selected subset of the landmarks reflects closely their distance in the network. This is achieved by applying the Simplex method [80]. Among the many models used to simulate and envision the Hyperbolic space [81], Rigel uses the hyperboloid model, where the distance between two points x, y in an n-dimensional hyperbolic space, is given by the following formula:

$$\delta(x,y) = arccosh\left(\sqrt{(1+\sum_{i=1}^{n} {x_i}^2)\cdot(1+\sum_{i=1}^{n} {y_i}^2)} - \sum_{i=1}^{n}(x_i \cdot y_i)\right) \tag{3.2}$$

After the execution of Rigel, the distance between any pair of nodes should be close to the true network distance. Thus, by employing this type of embedding, the problem of calculating the lengths of the shortest paths joining nodes of the network is reduced to a matter of simple mathematical operations, mitigating in this way the required time. A simple BFS algorithm in order to find the shortest path in an unweighted and undirected graph is of complexity $O(|V|+|E|)$, while the arithmetic executions require constant time. This improvement is essential for the operation of the Hyperbolic Edge Betweenness Centrality of each edge, discussed in detail in the following.

3.2.2 Hyperbolic Edge Betweenness Centrality Coupled with Neo4J

As discussed earlier, Girvan-Newman relies heavily in the computation of the EBC metric for every edge of the graph, with edges that display high value, having “bridge” like properties, thus, are likely to join nodes belonging in different communities. In order to identify the most central edges a new metric is introduced, named Hyperbolic Edge Betweenness Centrality (HEBC). To compute the HEBC value of every edge, the coordinates obtained by applying Rigel embedding on the network are used. As in the case of EBC, HEBC quantifies the number of greedy paths that pass through a specific edge over the total number of greedy paths joining the nodes of the network.

In order for the HEBC (and thus the whole HGN algorithm) to become more scalable as the input sizes increase, its execution has been coupled with a graph database as well as a standard relational one. Graph databases [82] are NoSQL solutions that enable the storage, querying and retrieval of large linked datasets, such as networks. The Neo4J graph database [83] is chosen because it is one of the most popular graph databases, used mainly for its scalability, its performance, the variety of available, implemented and optimized, graph algorithms, as well as its developer community that supports new implementations and enhancements.

In Algorithm 1, a pseudocode for the HEBC algorithm is provided. This algorithm returns the $batchSize$ edges that display the highest HEBC, where $batchSize$ is the size of the batch, meaning the maximum number of edges that can be removed in a single step of the HGN algorithm. This algorithm can be executed for large graphs with many edges as it allows the fetching of information in parts from the database.

In line 8 of Algorithm 1, an outer loop sets each node as a potential destination. Inside this loop, in lines 9-19, the nodes are sorted in a non-increasing order according to their hyperbolic distance from the destination node. By doing this, nodes can be examined in the correct order in the following parts. In lines 20-26, the number of greedy paths the source node and the destination node are calculated. A greedy path is a path constructed by moving from a node to a destination choosing each time a node nearest to the final destination [75]. Using greedy forwarding is another novelty of the proposed approach for speeding up the computation. Greedy forwarding using hyperbolic coordinates is known to produce greedy paths with length close to the shortest path length. In the last part of the algorithm, in lines 27-33 the dependencies δ, namely the number of greedy paths towards the destination that pass through the other nodes, are calculated for every node of the graph and so is the HEBC for every edge that appears on a greedy path towards the destination node. Finally, when the loop that started in line 2 terminates, the value of HEBC of every edge has been calculated. The reason that the HEBC values of the edges differ from the nominal EBC values is threefold. First, the number of greedy paths that pass through an edge is not always the same as the number of the actual shortest paths crossing it. Moreover, the greedy paths differ from the shortest paths in one more way. The greedy path from i to node j is not necessarily the same with the path from j to i. To overcome this obstacle the total HEBC value of an edge, i.e., $(i, j) \in E$, is taken as the sum of the HEBC regarding the directions $i \rightarrow j$ and $j \rightarrow i$. Finally, the embedding process is not free of errors in the distances of the nodes. Although the values of the edges concerning the EBC and HEBC metrics differ, the ranking of the edges according to their centrality values, is very close especially for the most central edges. In simpler terms, this means that edges that display a high nominal EBC value, will also display high HEBC values.

Regarding the time complexity of the HEBC algorithm, in line 8 a process is repeated for every node of the input graph $G = (V, E)$. Then, in line 18, the pairwise distances among the destination and the rest of the nodes are calculated, imposing an $O(|V|)$ complexity. The mentioned sorting can be done with an efficient algorithm (e.g., quicksort, mergesort) imposing a complexity factor of $O(|V|log|V|)$ time complexity. In the next parts, the algorithm in fact check every edge of G, thus including another $O(2 \cdot |E|)$ complexity. Finally, the loop that begun in line 8 is of complexity $O(|V|^2 + |V|^2 log|V| + |V| \cdot |E|)$. Because $|V| \cdot |E|$

Algorithm 1: HEBC - Hyperbolic Edge Betweenness Centrality

```
Input  : $src$: number of sources from which to estimate the HEBC of edges, $lcmp$:
         largest connected component, $batchSize$: number of edges in batch,
         $MaxNodes$: maximum number of nodes that can be retrieved at once
         from DB.
Output: $LargestHEBC$, containing top $batchSize$ edges in terms of HEBC
1  # get from database all nodes in component $lcmp$
2  $V = get\_nodes(lcmp)$,   $nodes\_number = len(V)$
3  # get number of edges for component $lcmp$ from database
4  $edges\_number = getEdges(lcmp)$
5  # data structure to store the HEBC value for each edge
6  $HEBC(i,j) = 0$, for every $i, j$ denoting edges in $lcmp$
7  Pick src nodes of set $V$, make set $V'$
8  for $each\ node\ s\ \in\ V'$ do
9      # read from database the coordinates of node s
10     $dst = getCoords(s)$
11     $dist = [0] * nodes\_number$
12     $V'' = V$
13     while $V''$ ! = [] do
14         pick $MaxNodes$ nodes in list $L$, remove from $V''$
15         $node\_coords = getMultipleCoords(L)$
16         for $u\ in\ L$ do
17             $dist[u] = hyperbolic\_distances(dst, node\_coords[u])$
18     Sort all nodes in order of decreasing hyperbolic distance towards the destination
        $s$
19     $Obtain\ S\ as\ S = \{v_1 \leq v_2 \leq ... \leq v_N\}$
20     # $\sigma_s(u)$: the number of greedy paths beginning at node $u$ and ending at node s
21     $\sigma_s = [0] * nodes\_number$
22     $\sigma_s[s] = 1$
23     $N_G(i,s)$ : the greedy neighbors of $i$ in $S$
24     for $i = N{:}1$ do
25         $v = S[i]$
26         for $each\ u_j\ :\ u_i\ \in\ N_G(i,s)$ do
27             $\sigma_s[j] = \sigma_s[j] + \sigma_s[i]$
28     $delta = [0] * nodes\_number$
29     for $i = 1 : N - 1$ do
30         for $each\ u_j \in\ N_G(i,s)$ do
31             $c = \frac{\sigma_s[j]}{\sigma_s[i]} * (1 + delta[i])$;
32             $HEBC(i,j) = HEBC(i,j) + c$;
33             $HEBC(j,i) = HEBC(j,i) + c$;
34             $delta[j] = delta[i] + c$;
35     $LargestHEBC = zeros(batchSize, 2)$
36     Find the edges that correspond to the $batchSize$ largest values of $HEBC$ and fill
        matrix $LargestHEBC$
37     Return $LargestHEBC$
```

is the dominant factor, the HEBC algorithm is of complexity $O(|V||E|)$.

3.2.3 Community Detection

Hyperbolic Girvan-Newman (HGN) is a modification of the original GN algorithm with hypebolic geometry aspects. In HGN, the network is embedded in hyperbolic space with Rigel embedding after proper tuning of the algorithm's parameters. Having an indication on the number of communities, the following process is executed. The HEBC values of all the edges are computed. Instead of removing just a single edge as it is the case in GN, a fixed number of top ranking edges, called the batch, are removed in each iteration. The batch is a rather important parameter and must be tuned according to the number of total edges in the network. As edges are removed two scenarios can occur. Either the batch runs out of edges with no new community being detected or the graph becomes disconnected. In the first case, the graph is re-embedded in the hyperbolic space while in the second case, a new community has been detected and the largest connected component is re-embedded and the process is repeated until the desired number of communities has been found. In the flowchart depicted in Figure 3.1 the steps of HGN are described and the points where the databases are required are highlighted.

An interesting aspect of HGN is its ability to perform well both for clustering spatial datasets and complex networks. To achieve the former, the problem of data clustering needs to be mapped to a problem of detecting communities in a graph. The graphs that are formed by joining data points are called proximity graphs. There are many ways to produce a proximity graph. In the case of HGN, the Disjoint Minimum Spanning Tree (DMST) algorithm [84] is preferred. The complete weighted graph is created by computing the similarity between all pairs of data points (nodes in the graph) and it is given as input to the algorithm. Then, DMST computes the first k (user-specified) MSTs, each time removing the edges that participated in the MST of the previous step (thus the disjoint characteristic). Finally, it joins the nodes with the edges of these k MSTs to produce the proximity graph. DMST is a sound choice because, with a small (of course, dependent on the size of the graph) choice of parameter k, it is possible to get a proximity graph that reflects the relations among the data (i.e., similar observations are linked together). At the same time, the obtained graph has low density that does not impede the execution time

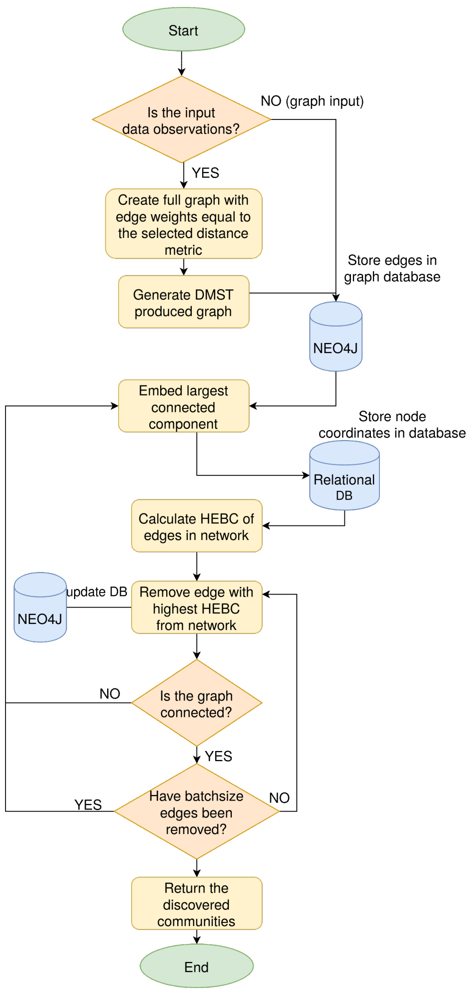
Start
Is the input data observations?
NO (graph input)
YES
Create full graph with edge weights equal to the selected distance metric
Generate DMST produced graph
Store edges in graph database
NEO4J
Embed largest connected component
Store node coordinates in database
Relational DB
Calculate HEBC of edges in network
NEO4J
update DB
Remove edge with highest HEBC from network
NO
Is the graph connected?
YES
YES
Have batchsize edges been removed?
NO
Return the discovered communities
End

Figure 3.1: HGN flowchart.

for the rest of the algorithm. The time complexity of building the full weighted graph by calculating all the pair-wise distances and performing DMST is $O(|V|^2)$.

3.2.4 Evaluation of the HGN Algorithm

In this section, experimental results are presented. These results showcase the strong points and benefits of the HGN algorithm. To begin with, the accuracy of the HEBC metric compared to the nominal EBC values on some common benchmark graphs is presented. Following that, the community detection performance under various options of the *batch* parameter is examined, and more specifically the manner in which it affects the execution time of the HGN algorithm and the quality of the solution in terms of the achieved modularity score. Then, results on the framework are presented and discussed when applied on both real and artificial datasets.

3.2.4.1 Evaluation of the HEBC Computation

In this subsection, the accuracy of computing HEBC metric in scale-free artificial networks and some real social networks obtained from [85] is presented. The main aspects of these graphs are presented in Table 3.1.

Table 3.1: Graphs description

Graph name	**#nodes**	**#edges**
scf1	1000	5788
scf2	1000	5821
karate	34	78
dolphins	62	166
lesmis	77	258
polbooks	105	442

Figure 3.2 presents the percentage of correct prediction of the top-k edges ranked according to EBC and HEBC and shows that notable scores of accuracy are achieved for these types of scale-free and power-law degree topologies. EBC ranking is used as a reference. This figure quantifies the percentage of accuracy achieved by the ranking according to HEBC with respect to the ranking of the top-k edges by EBC. This is expected, because using Rigel embedding for power-law and power-law-like (exponential) networks yields less distortion of the distances in the embedded network, i.e., they are closer to the original ones. The satisfactory results for the social networks examined are justified, since such networks are known to follow a power-law or a power-law like degree distribution [6].

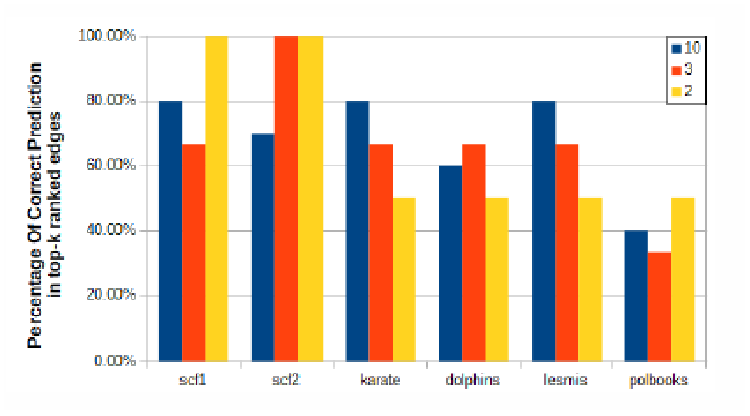

Figure 3.2: Percentage of correct prediction in the top-k ranked edges for the networks presented in Table 3.1. Blue columns denote the accuracy achieved for the top 10 (first of each triplet of bars), red for the top-k= 3 (second of each triplet of bars) and yellow for the top-k = 2 edges (third of each triplet of bars).

3.2.4.2 Sensitivity of HGN regarding the *batch* parameter

As noted earlier, the size of the *batch* is an important parameter to consider when executing the algorithm. In Figure 3.3, in the left axis the time needed for the termination of HGN, when applied on *graph7* of Table 3.5 is displayed, while on the right axis the obtained modularity is depicted. As it can be seen, when the size increases the algorithm tends to terminate faster until it reaches a certain plateau, where, except for minor fluctuations, the execution time reduces at a much lower rate. This is because the crucial edges for community discovery have already been detected and adding more edges to the batch no longer affects the resulting partition. The slight decrease is due to the fact that the graph becomes sparser and some computations are completed sooner. Similar conclusions can be drawn for the modularity metric as well. It is observed that as the size of the batch increases it is possible to remove edges of low edge betweenness score that will lead to the formation of small communities, resulting in low modularity scores. From the figure it is observed that modularity drops and then increases again, but as a general rule, it can be deduced that any batch size larger than 500 (for this graph) may impact negatively the community detection process, since the process becomes unstable afterwards. To conclude, the appropriate selection of the *batch* parameter plays a crucial role in the performance of the algorithm and a balance between execution time and quality of the partition must be achieved.

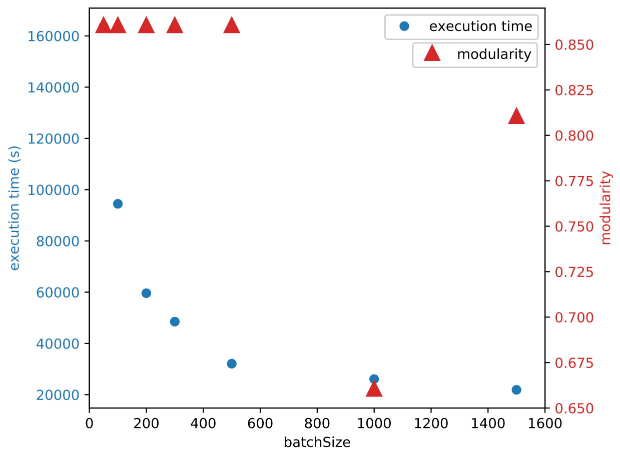

Figure 3.3: Performance of HGN under different sizes of batch.

3.2.4.3 Evaluation on Real Datasets

In order to demonstrate the advantages of the suggested framework, the algorithm was executed with input three real-world datasets. The first one is the Western States Power Grid of the United States[1]. The second is a network representing friendship between users of the online social network Facebook[2]. Finally, the network named "web_indochina" represents the links between web-pages as compiled by a web-crawler[3]. Some basic attributes of these networks are summarized in Table 3.2.

Table 3.2: Characteristics of real networks

Graph name	# nodes	# edges	avg. cc	# comms.
power_grid	4941	6594	0.08	43
facebook	4039	88234	0.61	10
web_indochina	11354	47606	0.71	10

The results demonstrated in Table 3.3 show that the proposed model is able to detect communities of high quality and, additionally, faster than the original GN algorithm when the network is dense, which is the case in the Facebook and web_indochina graphs. This marks a significant improvement in the scalability of this kind of community detection algorithms. In all cases, the NMI score between the two compared methods is high enough, meaning that a relatively similar clustering is achieved with either of the two methods. It

[1]http://networkrepository.com/power-US-Grid.php
[2]https://snap.stanford.edu/data/ego-Facebook.html
[3]http://networkrepository.com/web-indochina-2004-all.php

is safe to assume that the proposed algorithm is a competent alternative for this kind of networks because it can be employed to save time and still be able to identify meaningful communities of high modularity, especially as the size of the graph increases.

Table 3.3: Results on real networks

Graph name	**HGN mod**	**HGN time**	**GN mod**	**GN time**	**NMI (GN,HGN)**
power_grid	0.79	29912	0.93	8939.35	0.45
facebook	0.59	18999.12	0.7	108875.01	0.84
web_indochina	0.60	24781.67	0.7	35434.89	0.84

3.2.4.4 Spatial Artificial Datasets

In order to examine the behavior of the integrated system, the performance of the algorithm is evaluated on datasets with visually verifiable clusters and graphs with known communities.

Aiming to demonstrate the feasibility of the algorithm to detect clusters of spatial data, the employed datasets consist of points in the two dimensional Euclidean plane so that it is easy to validate the accuracy of the clustering. Three indicative examples will be presented.

These datasets were transformed into graphs by applying the DMST algorithm discussed earlier, with the number of disjoint MSTs set equal to 5. The HGN algorithm is executed setting the required number of communities. The datasets employed are described briefly in Table 3.4.

Table 3.4: Spatial graphs description

Graph name	**number of nodes**	**number of edges**	**communities**
noisy moons	1000	4995	2
kernel	1000	4995	2
blobs	2000	9995	4

Due to the nature of the datasets, it is easy to verify whether the partitioning of the graph into communities results to a correct clustering of the nodes or not. In order to showcase the advantages of the HGN algorithm when applied to the clustering of spatial datasets the output of the HGN algorithm is compared to that of two other data clustering techniques, as well as the well-known community detection algorithm Infomap in which the graph produced from the DMST algorithm is given as input, as in the case of the proposed HGN algorithm. The first data clustering algorithm is the well known k-means algorithm and the other is the algorithm SC-SGRF presented in [86]. This algorithm performs subspace clustering by considering a network formed by constructing multiple K-nearest neighbors affinity graphs, each at a different random subspace. These affinity matrices are fused together and then by means of spectral clustering data are clustered in the required number of clusters. In Figure 3.4, it can be observed that the proposed framework is able to detect correctly the clusters in all cases. On the other hand, k-means is able to achieve 100% accuracy only in the case of “blobs” dataset where the clusters are spherical, SC-SGRF fails to detect 100% the correct clusters in all cases. Finally, Infomap cannot detect correctly the communities in the last dataset (noisymoons) as it classifies a few nodes in the wrong community. This proves the ability of the algorithm to detect clusters of arbitrary shape, as long as the embedding parameters of Rigel and the size of the batch are tuned appropriately. These results are a strong indicator that the algorithm is suitable for clustering data that are described by a multitude of attributes if they are considered as nodes having coordinates in a suitable space. The integration of database management systems to the original algorithm is crucial for the scalability, enabling the analysis of large scale datasets.

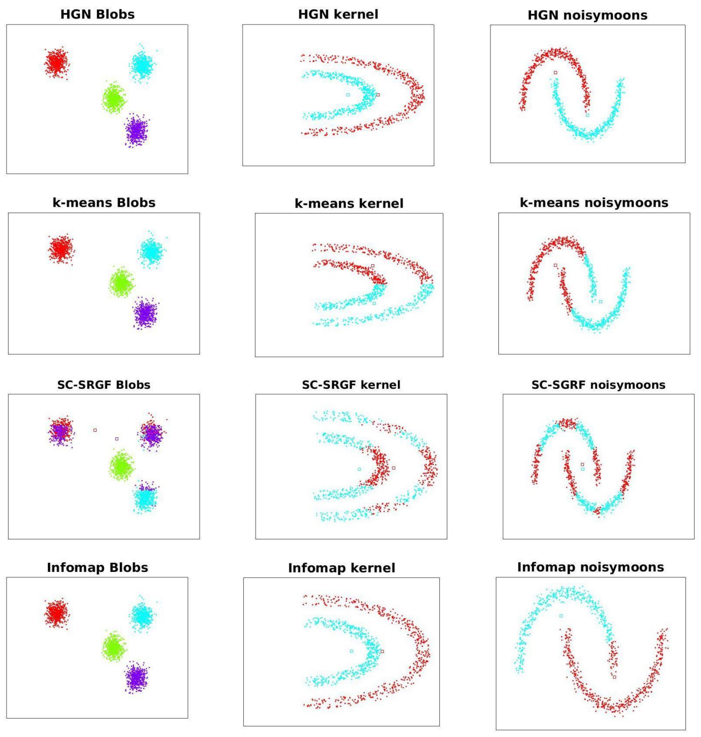

Figure 3.4: Visualization of HGN, k-means, SC-SRGF and Infomap produced data clusters.

3.2.4.5 Synthetic Graphs

The proposed framework can be employed in order to detect communities in graphs displaying relatively tightly knit communities (e.g., being able to achieve modularity scores over 0.3). In order to test the algorithm, benchmark graphs that were constructed following the procedure described in the LFR benchmarking algorithm [87]. The graphs that this method generates are considered ideal for testing hierarchical community detection algorithms that operate similarly to Girvan-Newman. For this kind of graphs, the communities are also given as outputs, so the partitioning of a community detection algorithm can be evaluated. In Table 3.5 some characteristic metrics of the employed graphs are presented.

Table 3.5: Artificial graphs description

Graph name	**#nodes**	**# edges**	**avg. degree**	**communities**
graph1	1000	4812	9.62	3
graph2	1000	5933	11.87	3
graph3	1500	7966	10.62	4
graph4	2000	19823	19.82	6
graph5	2500	31509	25.21	6
graph6	3000	44755	29.84	6
graph7	3500	122334	34.95	12

The proposed HGN algorithm is compared with other community detection algorithms, namely Attractor, presented in [88], GEMSEC, FluidC, and Infomap. Attractor exploits the interactions among the users and the influence that each node effects to others in order to identify communities of arbitrary sizes denoting groups of users that interact more closely than others. In order to compare these four algorithms the NMI score that each one achieved is compared to the ground truth of the LFR benchmark networks of Table 3.5. From these results, presented in Table 3.6 it can be seen that in all examined cases the HGN framework achieved an NMI score equal to 1, meaning that the partition of nodes in each network was identical to the one produced by the LFR algorithm, outperforming the Attractor-discovered communities as well as the GEMSEC and FluidC ones in terms of accuracy. Infomap is able to recover 100% the ground truth communities but it could not recover 100% correctly the spatial datasets of the previous section. These results verify that the HGN algorithm can detect with high accuracy and also achieve large modularity scores when applied to graphs that present strong community structure, with a large number of edges lying inside the same community and few to act as “bridges” that connect the different communities.

Table 3.6: NMI with ground truth comparison of HGN and the other community detection algorithms

Graph	NMI Attractor	NMI GEMSEC	NMI FluidC	NMI Infomap	NMI HGN
graph1	0.01	0.02	0.01	1	1
graph2	0.20	0.58	1	1	1
graph3	0.22	0.98	1	1	1
graph4	0.32	1	0.91	1	1
graph5	0.31	0.90	0.91	1	1
graph6	0.31	1	0.91	1	1
graph7	0.62	1	0.96	1	1

Another interesting aspect of HGN is the execution time required for its successful termination in comparison with the regular Girvan-Newman implementation. From Table 3.7 it is obvious that in the case of graphs with a relatively small number of edges the simple approach yields the same result faster than HGN, but as the networks grow larger and denser with more edges, HGN outperforms traditional GN to the point of requiring even less than 10% of the time for the termination, including the time required for the initial construction of the database.

Table 3.7: Results of HGN algorithm on artificial graphs

Graph	batch size	HGN modul.	Total HGN ex. time (s)	GN ex. time (s)	DB create time(s)
graph1	10	0.64	205.90	12.17	49.85
graph2	10	0.65	1033.72	475.07	116.9
graph3	10	0.72	1972.62	1159.73	133.60
graph4	20	0.82	6548.61	6483.88	480.69
graph5	20	0.83	15612.72	18495.73	826.71
graph6	20	0.83	32406.30	45156.91	1292.54
graph7	500	0.86	32095.16	502647.79	2233.74

3.2.4.6 Evaluation of the Framework on a Real Smart City Topology

The HGN framework was also tested in the Smart-Santander smart city [89], one of the largest modern smart cities, located in the town of Santander in Spain, where hundreds of sensors are deployed all over it. These sensors are able to measure a multitude of resources such as environmental measurements (e.g., temperature, humidity, particle concentration), garbage management, electric power consumption, etc. During the experimentation, environmental resources were polled with varying time granularity. The purpose of the conducted experiments were to show that HGN is capable of detecting meaningful clusters of diverse sensory data, obtained from operational smart city topologies, at realistic scales, as well as yield a framework for energy reduction in sensor networks.

To develop this framework, first there was the need to determine in an efficient manner which sampling instances can be omitted in a specific set of measurements defined by a sampling rate, thus conserving the associated energy for all involved sensors, and secondly, identify the sensors that exhibit practically identical behavior in the data clusters and use them either for monitoring load balancing or measurement prediction. In both directions, energy savings are gained by determining additional idle periods for sensors, based on the obtained data clusterings provided by the developed analytics approach.

To achieve the aforementioned goals, data from the Smart Santander smart-city, obtained from the FIESTA-IoT platform [90], a federation of testbeds, allowing access to real data measurements are used. The datasets are obtained at different sampling rates, e.g., 1 sample/5min, 1 sample/20min, etc. During the experimentation phase, measurements were collected from various FIESTA-IoT resources and of different types, e.g., temperature, relative humidity, humidity, illuminance, sensor battery levels, NO^2, NO, CO^2, soil temperature, soil moisture, traffic intensity, etc., for different time periods ranging from a few hours to four consecutive days. Various such intervals were examined in order to assess the received data and compile suitable datasets. The number of employed resources ranged from a few sensors to approximately 450 IoT devices in total, excluding devices that returned non-numeric values. The compiled datasets were cleaned properly from any 'NaN' values present. The datasets are clustered using the HGN algorithm in order to produce clusters (communities) corresponding to specific sampling instances.

Furthermore, the HGN framework allows for the analysis of the datasets with the goal of identifying which nodes of the data graph (corresponding to multi-dimensional/co-located data) remain invariant within clusters. The intuition behind this, is that as multi-dimensional data evolve over time, the corresponding communities-clusters of data will change as well. If however some of these data remain constantly together (i.e., clustered together) in the same communities, this indicates the existence of a hidden correlation among them. The proposed approach allows discovering such seemingly hidden correlations. The set of nodes remaining invariant within clusters can be used in various capacities, e.g., energy reduction as was the original goal, measurement prediction, etc. For the case of reducing the required monitoring energy across nodes, the invariant nodes and especially those that are relatively close in terms of GPS location can be used interchangeably as active monitors, while the rest are set to idle operation, thus conserving operational cycles and the associated energy among them. The potential measurement outcome of some invariant nodes set to idle mode can be predicted by that of their neighboring invariant ones. The rest of nodes of each cluster that exhibit non-invariant membership to the cluster cannot be cast in idle mode, as their measurements are needed to maintain the same level of monitoring accuracy as that of the whole sensor network. A similar approach can be taken in a fault-tolerance application. Whenever one of the invariant nodes faces a fault, its value can be estimating using the values of the rest of invariant nodes. An example is presented in Table 3.8, where the stable groups of nodes are identified. These nodes remain clustered together across all the sampling periods with interval 5 minutes. It is, thus, safe to assume that if for example node 10, belonging in group 4, suffers a failure, its measurement could be approximated by the other measurements generated by the nodes in the same group.

Table 3.8: Stable multidimensional nodes per cluster in 1 sample/5min. series

Group 1	**Group 2**	**Group 3**	**Group 4**	**Group 5**	**Group 6**	**Group 7**	**Group 8**	**Group 9**
3	5	8	10	18	33	36	43	69
37	6	47	15	75	35	44	55	71
41	39	50	17	-	-	-	-	-
45	46	68	20	-	-	-	-	-
57	48	72	22	-	-	-	-	-
63	51	79	29	-	-	-	-	-
67	56	80	53	-	-	-	-	-
77	58	-	82	-	-	-	-	-
-	59	-	-	-	-	-	-	-
-	74	-	-	-	-	-	-	-

Moreover, the corresponding proximity graphs were compiled for each one of the datasets and HGN was compared in terms of obtained modularity with the original GN algorithm and a modularity maximization algorithm. The following Figures 3.5-3.8 display the obtained modularity scores for each sample. By examining each barplot, it can be seen that in all cases HGN produces results that are close to the maximum modularity and in many occasions surpassing in this metric the clusterings produced by Girvan-Newman.

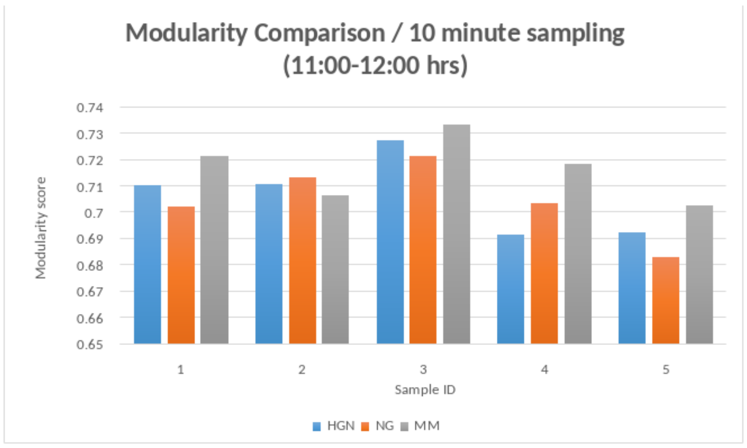

Figure 3.5: Comparison of data clustering approaches with respect to modularity for 1 sample/10min.

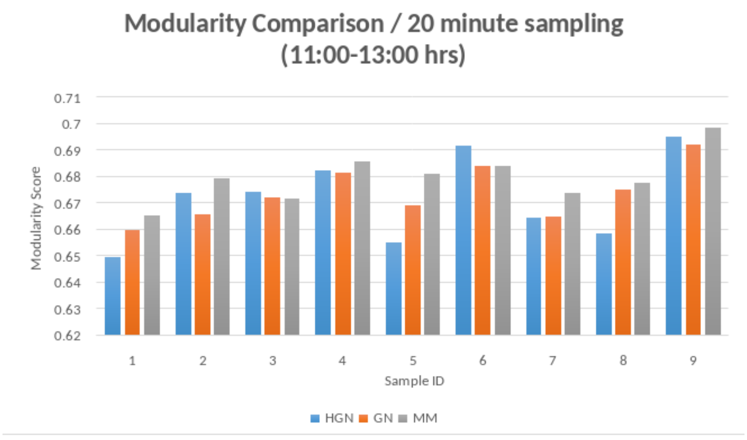

Figure 3.6: Comparison of data clustering approaches with respect to modularity for 1 sample/20min.

From the results presented in the previous sections the usefulness of the proposed HGN algorithm is justified. It is a faster hierarchical community detection method than the original GN algorithm, coupled with databases that ensure scaling in larger graphs. Moreover, HGN can be employed both over spatial datasets such as those generated by sensor network measurements and provide useful indications that can be applied in fault-tolerance and energy saving scenarios in real smart cities.

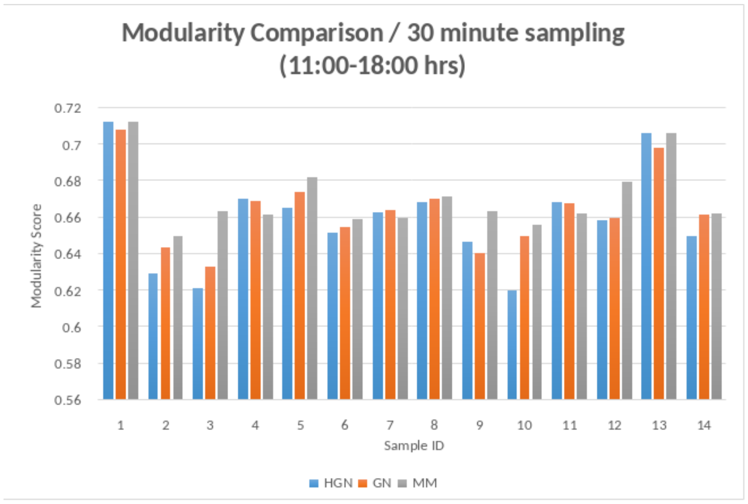

Figure 3.7: Comparison of data clustering approaches with respect to modularity for 1 sample/30min.

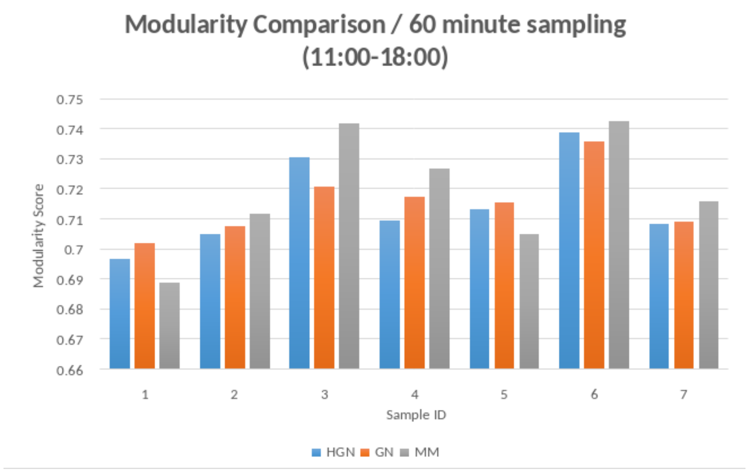

Figure 3.8: Comparison of data clustering approaches with respect to modularity for 1 sample/60min.

3.3 Employing HGN for Semantic Communities Visualization

In this Section, an interesting application of the HGN algorithm, aiming at visualizing semantic datasets that comply to the Resource Description Framework (RDF) is presented. The extended use of the RDF model has made available many datasets from heterogeneous sources that are of interest to a wide audience. Their exploration, however, is a highly demanding task requiring extensive training and knowledge of the SPARQL language, so various techniques for easing their exploration by plain users are developed [91]. In order to build an application that enables the exploration of such datasets from users without extensive knowledge on RDF or SPARQL, the HGN algorithm is applied in order to detect semantically similar communities. Then, the detected communities are visualized and presented within a user-friendly User Interface (UI) accompanied by a useful toolbox of functionalities.

In order for the proposed system to detect groups of similar triplets, each RDF triplet is treated as a point in a custom 3-dimensional space, with each one of the three dimensions corresponding to the *subject, predicate, object* parts of the RDF triplets. Then, a weighted graph is formed by employing suitable distance metrics alongside each dimension for each pair of triplets. In order to ensure the efficiency and scalability of the algorithm for very large datasets, the information is stored and indexed in a graph database where each node has three properties.

In order to calculate semantically meaningful weights the following procedure is followed. For the *subject* and *predicate* parts of the triplets, measures like the *edit distance* are employed in order to measure how similar pairs of them are in terms of lexical similarity. For the *object* part of RDF triplets, the proper distance metric depends on its data type and could be any metric applied to categorical (e.g., edit distance, Jaccard) or numerical data (e.g., euclidean distance). Thus, for each pair of triplets, three distances are calculated and then averaged. The averaging process can be assigned weights if more importance is required to be placed over an attribute.

After the full weighted graph is computed, by applying the DMST algorithm, a proximity graph is formed. This graph is the input of the HGN algorithm. Beginning by pre-processing the RDF triplets one can estimate the required number of communities. For example, if it is required to split the dataset into communities according to the predicate of the RDF, parsing all the RDF triplets allows for the computation of the number of different predicates.

In Figure 3.9 the way in which the proposed algorithm detects the communities on a small RDF dataset can be seen. The figure shows six nodes (i.e., RDF triplets) that relate to the ages and salaries of three people, as well as the complete graph created by following the process described above. The distances for each node pair are calculated as follows:

the euclidean distance is used for the numerical data age and $salary$ and the edit distance is used for the rest. For example, the distance between the triplets (:Alice, :age, :25) and (:Bob, :age, :27) is computed as follows. First, the distance $d('Alice', 'Bob') = 5$, then $d('age', 'age') = 0$ and finally the distance $d(25, 27) = 2$. From that it follows that the distance between these two triplets is $\frac{5+0+2}{3} = 2.3$ (with equal weights).

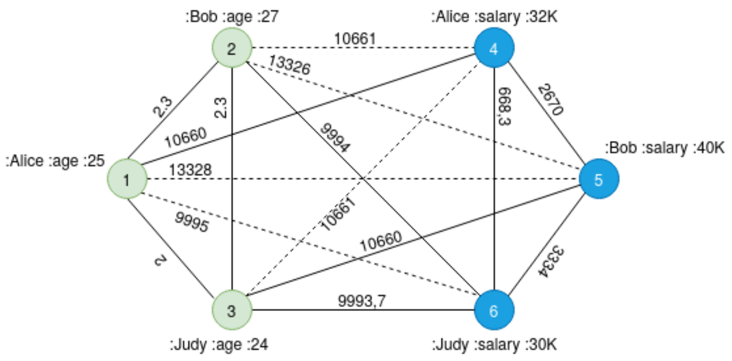

Figure 3.9: Weighted graph of RDF triplets.

Afterwards, the full graph is given as input to the DMST algorithm, which removes the edges shown as dashed lines, and then, the core part of the HGN community detection algorithm follows, which results in two communities denoted by the two different node colors. It is obvious that the algorithm discovers a community that corresponds to the ages of the people and a community that contains information about their salaries. It is important to note that a different averaging technique could result in a different community detection as more importance could be given to one attribute over another. As an example, if it was desired for the analysis to focus more on people's names, and thus possibly discover clusters containing information about the attributes of a person (e.g., salary, age, etc.), a weighted average with larger weight on the distance between names could have been employed.

The system is built around a server-client architecture that takes as input an RDF dataset, process it using the HGN algorithm, stores the communities into a Neo4j graph database together with the relevant attributes about every node (i.e., name, age, occupation, etc). The processed information is presented to the user through a visualization interface that employs the Neovis graph visualization algorithm [92]. Neovis is a dynamic, browser based visualization library based on vis.js. It is designed so it can be readily integrated with data from Neo4j databases.

The application utilizes two abstraction levels in order to present the semantically similar communities. The high level is a hypergraph with nodes representing communities and the edges representing the existence of edges joining nodes belonging in different communities, denoting in this way, the interconnections between the detected communities, as well as a brief summary regarding the context of each community. The second layer presents as a graph all the triplets belonging in each community.

In order to allow the user to explore the input dataset and the detected communities a series of functionalities are provided through the interface. Employing the indexing and querying capabilities of the graph database, the functionalities are provided in real-time.

Moreover, the system, enables the user to view the information within the communities using multiple filtering and aggregation criteria. For example the user can isolate a specific node type as well as nodes with a given node degree. To further support the exploration of the dataset, the user can locate information through keyword search. A term is matched against node labels and the result is presented as an appropriate subgraph containing all the matched nodes.

In order to present the developed interface an RDF dataset containing information about scientific publications made from the LAAS-CNRF group [93] from 2003 to 2006 is employed. The dataset was cleaned of any incomplete entries and the three most prevalent predicates were selected in order to produce the full weighted graph. Those were the *has-author*, *has-title*, and *has-date* predicates. Then, the distances between pairs of triplets were computed by applying the edit distance for all types of data. For the averaging purposes, the most weight was assigned to the predicate part since it is desirable to detect communities corresponding to each one of these predicates. Following this process, applying DMST and requiring the union of 5 MSTs, the final graph is produced and given as input to the HGN algorithm requiring the detection of three communities.

Figure 3.10 shows the interface of the system, the dataset selector, the functionality toolkit and the graph visualization. The graph panel depicts three hypernodes corresponding to the three detected communities. It should be noted that the detection was 100% accurate.

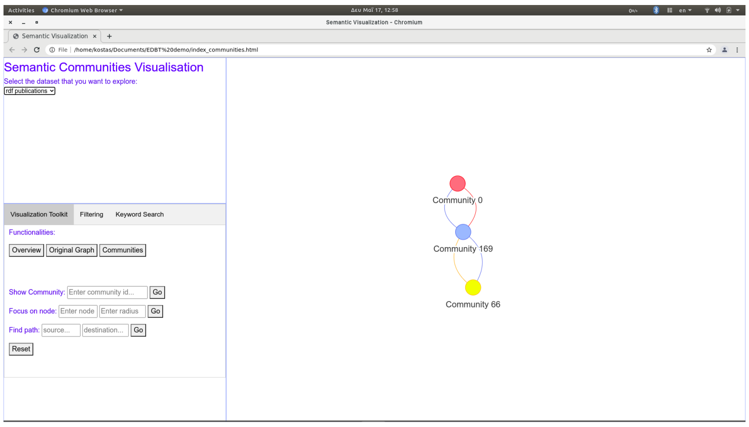

Figure 3.10: Semantic communities visualization overview.

The graph, with each community assigned a different color, can be seen in Figure 3.11. Also, in this figure the set of the offered functionalities can be seen in the lower left part of the screen. These allow for a more in-depth exploration of the dataset.

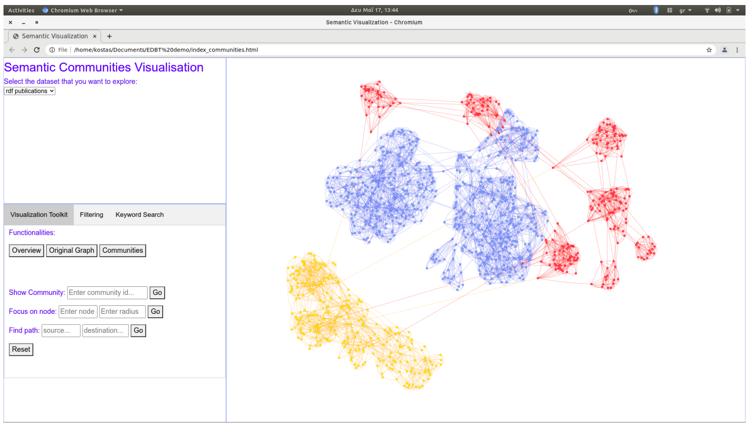

Figure 3.11: Detected communities of the publications RDF dataset.

For example, a user may choose to focus on a publication and see all the available relevant information about it. This functionality can be seen in Figure 3.12, where, through the search mechanism employed, the relevant RDF triplets for the publication titled “An empirical investigation of simulated annealing applied to property-oriented testing” were retrieved and displayed as nodes. From this screen, the user of the application can find out the authors of the paper as well as the publication date.

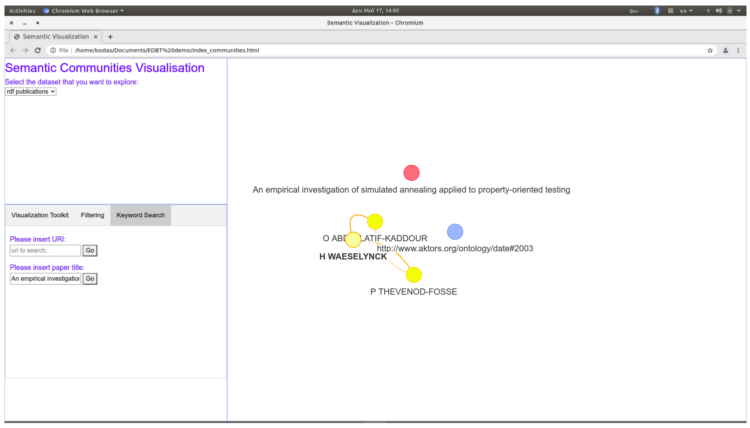
Activities
Chromium Web Browser
Δευ Μαΐ 17, 14:05
Semantic Visualization - Chromium
Semantic Visualization
File | /home/kostas/Documents/EDBT%20demo/index_communities.html
Semantic Communities Visualisation
Select the dataset that you want to explore:
rdf publications
Visualization Toolkit
Filtering
Keyword Search
Please insert URI:
uri to search..
Go
Please insert paper title:
An empirical investigation
Go
An empirical investigation of simulated annealing applied to property-oriented testing
O ABDELLATIF-KADDOUR
http://www.aktors.org/ontology/date#2003
H WAESELYNCK
P THEVENOD-FOSSE

Figure 3.12: Retrieval of information for a single publication.

In this Chapter, a framework for discovering clusters of data as well as communities in networks, enhanced with the appropriate databases, was examined and its strong aspects were discussed. Detecting communities is one of the basic techniques employed for a network's analysis. It is not though capable, on its own, of generating a complete overview of a complex system, providing all the necessary information about its actors, since several other aspects, such as the diffusion of information as well as the possible communications between the actors of the system must be studied. The following chapters deal with relevant problems in these fields.

Chapter 4

Allocating Recommendations in OSNs

In Chapter 3, the organization of nodes in communities for many different types of networks was examined. Knowledge about the existing groups in an OSN is a method that can aid in the network's analysis and also in inferring which users are similar. The analysis of users interactions though cannot always reveal if the users are indeed similar or if one user was influenced by the other [94]. Therefore, studying the diffusion of information in such a network allows for gaining further insights on the role of each user. These insights are crucial for estimating the effect of recommendations in the network. Recommendations are ever present in today's networks and their importance is highlighted in the following.

In the large interconnected environments studied in this thesis, OSNs are expected to play a crucial role [95, 96, 97]. They are the platforms employed by users for a variety of purposes (e.g., messaging, getting informed about news and events, etc.). These users, that live in such environments, can also gain access to measurements, information and data made available by infrastructure or smart devices, via the use of suitable applications (e.g., weather apps, smart parking, etc.) or even crucial Covid-19 information [98]. It is evident that the advertisement of these applications is necessary in order for the users to become aware of their purpose and significance. A Recommender System (RecSys) operating in the OSN platform can be utilized for this task.

The process of information diffusion among users of an OSN is an important factor that should be taken into account when examining the allocation of recommendations to users. Recommending the appropriate items (e.g., applications, events, video files, etc.), to a carefully selected subset of users can increase their adoption [99]. The interactions of the users reveal certain degrees of influence among them. A user that is influenced in a significant degree by another is likely to adopt this user's choices (i.e., follow the user). Knowledge about the amount of influence a user can exert over another is quintessential when studying a diffusion process in the network [100, 101]. Moreover, users, through their past activity in the OSN reveal their relevance towards the certain types of items that are available for recommendations. Recommender systems that take into account the information diffusion in a network belong to special category called Information Diffusion Aware Recommender Systems (IDARS) [102, 99, 103].

Such systems are able to determine the flow of information about the suggested items throughout the underlying OSN [99]. This feature allows increasing the diversity of recommended items by avoiding redundant recommendations of items that users will eventually learn about through the activity of their peers in the OSN, i.e., items that will become available in any case through likes, shares, retweets, etc. Moreover, IDARS are based on the assumption that users prefer the recommendations of items from their friends rather than recommendations generated from a RecSys [104].

Even though these methodologies reduce significantly the amount of duplicate recommendations, information overloading is still possible, leading a user to not paying attention to the advertised products [105], refrain from any activity or even lose interest for the OSN platform [106]. One such scenario of overloading is when a user is recommended of multiple different products that may lead to confusion about choosing the one most suited to his/her needs [99]. Additionally, multiple recommendations of the same item to the same user are prone to lead to frustration.

Studying the problem of information diffusion in social networks, the focus is on discovering which recommendations to which users will result in the maximum total relevance score obtained in the network. At the same time strict constraints are imposed, that guarantee that users will not be overwhelmed by a large amount of information that may ruin their decision process. The goal is to design methods that will aid in the diffusion of applications in a large interconnected environment, like a Smart City, with the majority of recommendations being “implicit”, i.e., recommendations generated from an influential user rather than “explicit”, i.e., generated by a RecSys. Also, these recommendations should be as unobtrusive a s possible aiming at actually assisting the user in getting advantage of the benefits of living in a Smart City, rather than spam her with repetitive recommendations.

In this Chapter, several related works that have inspired the development of the proposed algorithms will be described, followed by an in depth description of the proposed recommendations framework, denoted as Socially Constrained Recommendations (ScoRe). Also, indicative results on synthetic datasets, that showcase its suitability will be provided.

4.1 Documenting the Need to Avoid Information Overload

One of the most popular categories of recommender systems are Content-based RecSys. This kind of systems propose to the users products that are similar to them. In order to achieve this task, the systems maintain databases that contain descriptions of the items, as well as information about each user’s preferences. In that way, they are able to recommend products similar to their tastes. Item descriptions can be materialized using Linked Open Data in order to produce richer descriptions. Also, information obtained by multi-layered graphs joining users, items, and their properties, can aid in designing more accurate recommendations as it can be seen in [107]. The most common way used to encode textual item descriptions is the TF-IDF or other latent representations [108], that aim to identify important words in the descriptions. In addition to these methods, there also exist embedding techniques like Meta-Prod2Vec described in detail in [109]. In this paper, item descriptions are mapped to an appropriate multi-dimensional space aiming at placing relevant products in close proximity in order to produce clusters. The inclusion of metadata in this algorithm contributes in more accurate recommendations than previous techniques that did not include them. Although Content-based systems are able to recommend relevant items, they are not without limitations [110]. First of all, Content-based RecSys suffer from the cold start problem. If the user does not input enough attributes to the platform, or has not yet rated other items, then, it is not possible to receive accurate recommendations. The same is true for an item not described appropriately. Moreover, in most cases, users explicitly rate only a few items, making the process of inferring their similarity to the vast amount of unrated items a challenging task.

Another popular approach for designing recommender systems are the Collaborative-filtering (CF) RecSys. In these RecSys, the system learns to recommend products to each user based on the ratings provided by users with similar tastes. A common approach in CF RecSys is based on the similarity vectors between users and items. The goal is to determine for each user the most similar users to her, and then, produce recommendations based on these users' preferences. CF is oftentimes used for recommending friends in OSNs, music, movies, etc. [111]. For example, in [112], a CF recommender system is developed in order to recommend friendships in OSNs like Twitter. The proposed technique recommends potential new friendships based on the already formed relationships. In order to avoid the phenomenon of creating a few users of very large degree, similarity measures are employed. In this paper, the distinction of directed OSNs (e.g., Twitter) regarding the propagation power, in contrast to OSNs with reciprocal relationships, is highlighted. Although CF filtering RecSys are widely used, they too suffer from problems like cold start in the case of a user joining the system and not providing enough information in order for similarities with other users to be established. Moreover, both CF and Content-based RecSys fail to take into account the information diffusion that occurs in an OSN due to the users interactions. Also, the fact that each user has a unique tolerance on the amount of information she can receive during a short period of time is not considered. There is a number of other recommendation techniques like knowledge-based [113, 114], location-based [115, 116], demographic [117], etc., as well as hybrid RecSys that combine two or more techniques in order to achieve more precise recommendations [118]. Limitations on these traditional approaches are discussed in detail in [119].

Recommender systems are also widely used in Smart Cities. In [120] recommendation techniques are proposed aiming to recommend to the administrators of the Smart City actions that will increase the range of activities that are monitored by the city's infrastructure. The authors in [111] deal with the issue of designing a CF recommender system that can operate in a Smart City while respecting the users' privacy. The authors consider a set of users and a set of items that are available for recommendation. In order to achieve their goal of protecting the information that users have entered in the system, a directed trust graph is used with edge weights denoting the amount of trust between pairs of users. This directed trust graph can be considered similar to the influence graph employed in the proposed method, described in the following Section. Also, the survey presented in [121], overviews the many different types of recommender systems that are currently employed in such large topologies, thus highlighting the importance of the RecSys in the efficient operation of a Smart City.

In a highly interconnected environment with many interdependent actors, OSNs play a key role in the spread of information [122, 6]. A recommender system that can employ both knowledge of the users' relevance to the available items, using any of the techniques used in CF or Content-based RecSys and also combine it with knowledge about the social aspects of each user can be very beneficial both for the owners of the platform and the advertised retailers. In an OSN, a user that receives recommendations for an item (e.g., product, event, etc.) may choose to pass along this information to its neighbors, or followers in a directed OSN, provoking their reaction to the recommended item. RecSys that utilize information gained from OSNs are called Social Recommender Systems. In [123], FLOWER is presented. This recommender system captures local and global correlations between people of an OSN in order to suggest potential links. The inclusion of local correlations associated with triangular motifs improves significantly the performance of the algorithm compared with other methods that utilized only global features. The authors in [124] propose a query personalization algorithm in order to supply users of an OSN with more personalized responses to their queries. In order to achieve this task the authors consider, among other factors, the influence that other users may exert on the user that posted the SQL query. It is claimed that a user is more probable to respond to recommendations originating from friends in the OSN which are highly influential to this user. This approach is also followed in the proposed framework SCoRe where information is considered to travel along paths of users that influence one another in significant degrees.

The modern sizes of OSNs coupled with the enormous variety of available items leads to an exponential growth of the information that travels across the networks. On the other hand, the cognitive capacity of each user remains limited. This inevitably results in information overload for the users [125]. In this paper, the authors show that the information processing behavior relies on the rate at which they receive information. They also state that novel information exposure at users that receive information at high rates tends to be less effective. Information overload, at individual level, can ruin the users' decision making [126], and eventually compromise the users' engagement to the platform. The paradoxes that arise in social network are reviewed in [127]. There, an analogy is drawn between the Braess' paradox [128], where adding more options to the users of an OSN may lead to worse overall performance, while removing options may lead to better results. The conclusions drawn from these works highlight the need to take seriously into account the toll of information overload both at the user and also at the network level. In [129] the authors propose a RecSys that provides people with sequential recommendations for points of interest in a city. Taking into account the possibility of information overload, it imposes constraints so that users do not get overloaded with information by trying to avoid duplicate recommendations to each user. Despite this fact though, this method does not take into account the possible interactions and information sharing among the users.

Lately, Social Recommender Systems tackle this problem by filtering huge collections of items (i.e., friends, products, events), while leveraging behavioral properties (e.g., homophily, trust, influence) and the interplay between the entities of a social platform [118]. From this point of view, the authors in [130] propose an information flow driven recommendation scheme that uses time and topic related adoption patterns of the users, categorizing the former to early and late adopters, in order to predict the extent of an item's diffusion in the network. Estimating accurately the potential adopters is pivotal to the selection of the seed users from the Social Recommender System. Incorporating information about users' locations, in [131], the authors study the diffusion process among users of an OSN. They consider that a user u spreads a new piece of information only if it originates from a user v that has greater influence on u than u has on v and if also this influence is higher than a predefined threshold. A CF filtering approach enriched with information diffusion is proposed in [102] that produces good results in directed networks.

The work on ScoRe, presented later on, is more closely related to [99]. In this paper, the authors propose DifRec, an algorithm providing recommendations to users of a social network with follower-followee relationships, e.g., Twitter. DifRec takes advantage of the fact that information shared by a user will eventually spread across nodes that follow her, dismissing the need to recommend explicitly an item to a user that will get the information anyway. Although, DifRec marks a step towards addressing the problem of information overload [125], there still remain aspects of information diffusion not considered. Namely, the difference in tolerance displayed by each user towards either repetitive advertisements of the same item or towards the number of different items at a specific time. These are the key issues addressed in this work where the items' allocation takes into consideration its anticipated diffusion in the network while being constrained on users' capacity to both duplicate and distinct items.

4.2 Socially Constrained Recommendations - ScoRe

The approach presented in this thesis differs from others in the field of IDARS. First of all, it considers a directed network of users that influence one another at different rates for every distinct item available. This seems a rather realistic assumption, since a user may trust a certain user for one item category (e.g., sports) and easily adopt choices for this topic but may not display the same trust for other items (e.g., music). A lot of effort has been put in order to make the recommendations originating from the RecSys as unobtrusive as possible, respecting user boundaries on the received information and at the same time manage to recommend highly relevant items to the users. For this task, two new types of constraints are introduced. A constraint on the amount of duplicate recommendations as well as a constraint on the number of distinct item recommendations. In order to achieve the aforementioned goals, the problem of selecting the seed users for explicit recommendations has been modeled as a relevance score maximization one, that is subject to the aforementioned constraints. Due to its computationally expensive nature, it was divided into two sub-problems that must be solved in sequence. For each one of these problems, two different methodologies have been proposed. In the following, the process is explained in detail.

4.2.1 System Model for SCoRe

Information sources in an OSN can be classified into external and internal [132]. In this work, the RecSys is considered part of the OSN, exploiting all publicly available user information in the OSN. Within the framework of the OSN, the RecSys is considered as the external source of information, whereas the users of the OSN that initiate the diffusion of items in the network are the internal sources of information. In particular, user u is characterized as an internal source of information if any user received from u a piece of information. The term *explicit recommendation* will be used for endorsement by the external source and the term *implicit recommendations* for endorsement by the internal sources of the network. In addition, it is assumed that users actively engage with the recommended items through actions visible to their neighbors, e.g., sharing a post, commenting, retweeting, etc.

4.2.1.1 Multilayer User-Items Graph

The relationship between internal and external sources of information in an OSN is modeled with a two-layer, weighted and labeled directed graph $G = (\mathbf{V}, \mathbf{E}, \mathbf{W}, \mathbf{c}, \mathbf{C})$, as presented in Figure 4.1. This multi-layer graph is indicative of the interdependent nature of the problem of recommendation allocation, where $\mathbf{V} = \{U, I\}$ is the set of nodes which consists of the set of N active users (user space) $U = \{u_j\}, j = 1, ...N$, and the set of M items (item space) $I = \{i_k\}, k = 1, ..., M$. The set of edges $\mathbf{E} = \{E_U, E_{IU}, E_I\}$ consists of:

- $E_U = \{(u, v); u, v \in U\}$, which is the set of directional edges between users based on their social relations (who influences whom in the network),
- $E_{IU} = \{(i, u); i \in I, u \in U\}$, which is the set of undirectional edges between items-users and
- $E_I = \{(i_j, i_k); i_j, i_k \in I, i_j \neq i_k\}$, which is the set of undirectional edges between items.

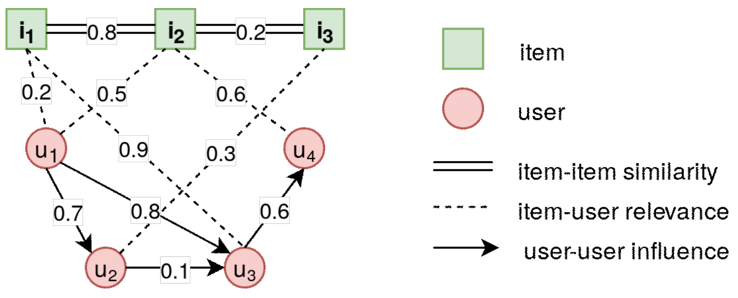

Figure 4.1: Example of IDARS graph. An items-users network with $I = \{i_1, i_2, i_3\}$, $U = \{u_1, u_2, u_3, u_4\}$ and their corresponding edge-weights.

The set of weights is defined as $\mathbf{W} = \{\mathbf{W}_U, \mathbf{W}_{IU}, \mathbf{W}_I\}$ with $\mathbf{W}_{IU}$ being an $|I| \times |U|$ matrix of item-user relevance, with $w_{IU} : E_{IU} \to [0, 1]$ denoting the item-user relevance. A large value implies high relevance of the item to the user. $\mathbf{W}_I$ is an $|I| \times |I|$ matrix of items' similarity with $w_I : E_I \to [0,1]$, which is pre-calculated with a similarity measure (cosine or Pearson coefficient, co-view metric, etc.) that can be employed on the publicly available information of the OSN. Finally, $\mathbf{W}_U$ is a $|U| \times |U|$ matrix of influence probabilities between users with $w_U : E_U \to [0,1]$, which is determined by users' past activity in the network and it is defined in detail in the following subsection.

In the vast environments examined, with large OSNs being formed, consisting of many people, it is expected that there exist users that exhibit diverse amounts of resilience to information overload [133]. In order to reflect this fact in the proposed model, $\mathbf{c}$ is defined to be a vector of size $|U|$ with $c(u) : U \to \mathbb{N}$, denoting the capacity of user u to distinct items. This is the cumulative number of different items that can be recommended to a user during a single time slot. Matrix $\mathbf{C} = (C_{i,u})_{|I|\times|U|}$, with $C(i,u)$: $I \times U \to \mathbb{N}$, models the capacity of user u for duplicates of item i. An item is characterized as duplicate of item i, if it is an exact copy of the latter. In this work, the terms duplicate and copy are used interchangeably. A user receives copies of an item if she is recommended of that item by more than one sources. Consider user v, in Figure 4.2, who receives information from users a, d, u, f and g and is not yet exposed to item i. If users a, d and g are assigned of item i by the RecSys, then, v is assumed to receive item i and two duplicates of item i.

The explicit recommendation of an item i to a user u results in an information diffusion process over the nodes of G. The ReliableSet algorithm presented in [99] is extended, to model the information diffusion and infer the influence graph for each item, by taking into consideration the influence $\mathbf{W}_U$ between users and the item-user relevance $\mathbf{W}_{IU}$. The edges of the influence graph of item i can be interpreted as indicators of which user receives implicit recommendations of item i from which other user [134].

The computation of the reliable set is a Monte Carlo approach, which, given a probabilistic graph G_P, a node of the graph and a threshold value $\eta \in [0,1]$, samples a set of T deterministic graphs and returns the reliable set of the node. This means, all the nodes that are reachable from this specific node in a fraction of graph instances that is larger than ηT. For item i, the probabilistic graph $G_P^{(i)} = (U, E_U, p^{(i)})$ is defined in order to compute the reliable sets of nodes for this item. The computation of the probabilistic graph's edge probabilities is based on the fact that it is not possible to have the same influence graph for all items: a high influence score in a pair of users in the initial graph does not imply that users' preference is similar for all items. Thus, the probability graph should be item-dependent. So, instead of computing the edge probabilities of $G_P^{(i)}$ based only on the influence between users, as in [99], the item-to-user relevance w_{IU}, which results in item-determined reliable sets is also considered. The edge probabilities are:

$$p^{(i)}(u, v) = w_{IU}(i, v) \cdot w_U(u, v), \quad \forall (u, v) \in E_U. \tag{4.1}$$

By executing the algorithm for all the users of the network, the family of the reliable sets for item i is computed, $\{RS^{(i)}(u),\ u \in G\}$, i.e., the set of users that can be influenced by each user u for a specific item i in the system graph. If item i is recommended to node u from the RecSys, the nodes in its reliable set $RS^{(i)}(u)$ will be implicitly recommended of item i through u. These sets will determine the influence graph of item i, $G_F^{(i)} = (U, E_F^{(i)})$, which is a deterministic directed graph with $E_F^{(i)} = \{(u, v) : u \in U, v \in RS^{(i)}(u)\}$.

By knowing the previous interactions between users for each item, the measure of influence between them can be extracted. The users that previously recommended item i to user u are denoted by $V_{in}^{(i)}(u)$, thus $V_{in}(u) = \bigcup_{i \in I} V_{in}^{(i)}(u)$ is the set of users from whom user u previously received implicit recommendations. The set of implicit recommendations that user u was successfully subjected to and originated from v, is $r_{im}(v, u)$. In this way, the set of all implicit recommendations that user u reacted to during a predetermined time window can be defined as $R_{im}(u) = \bigcup_{v \in V_{in}(u)} r_{im}(v, u)$. Then, the influence of node v to node u is given by:

$$w_U(v, u) = \frac{|r_{im}(v, u)|}{|R_{im}(u)|}. \tag{4.2}$$

In the following, the matrix $\mathbf{W}_U$ is considered to be known in advance, supposing that there exist historical data about the behavior of users to recommendations from their peers.

4.3 Modeling Relevance Maximization Problem with Multiple Constraints

The proposed RecSys should allocate items to users in order to maximize their total relevance to items, while ensuring that the information capacity of each user (via explicit and implicit recommendations) is not exceeded. It should be noted that maximizing the total relevance of the network does not imply achieving maximum relevance for each user as well. In the following, first the two types of violations that can occur by not respecting the imposed constraints are presented. Then, the sub-problems are presented formally, and finally, the algorithms that compose SCoRe are described.

Without loss of generality, it is assumed that all users have the same resilience to information overload, thus $c(u) = c$ and $C(i, u) = C \leq max(deg_{in}(u))$, $\forall u \in U$, $\forall i \in I$, where deg_{in} is the in-degree of nodes in the influence graph. The set of implicit recommendations to user u is denoted by $R_{im}(u) \subset I$ and the set of explicit recommendations to user u is denoted by $R_e(u) \subset I$. Below, examples of capacity violations that would emerge in the model proposed in [99] are given.

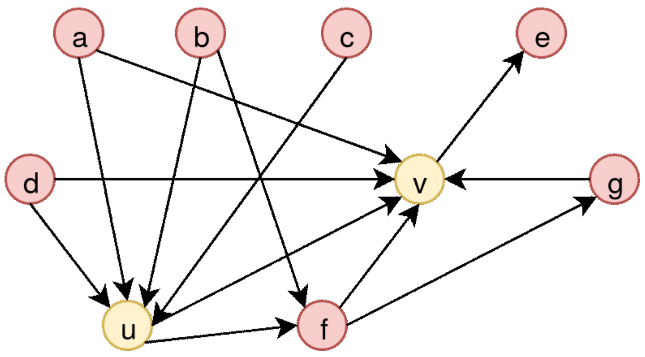

Figure 4.2: Example of capacity violations. A network consisting of nodes u, v, their one-hop neighbors and the edges between them in the influence graph, which is the same for all the recommended items under consideration.

Example 1 - Violation of a user's capacity for distinct items, c.

Let there be users u, v of the network shown in Figure 4.2 with $G_F^{(i)} = (U, E_F^{(i)})$ for all the items i under consideration. The nodes u, v exhibit the same information capacity, $c(u) = c(v) = 5$, and have already received implicit recommendations $R_{im}(u), R_{im}(v)$ from b

and f, g correspondingly, with $|R_{im}(u)| = 1$ and $|R_{im}(v)| = 2$. The RecSys, with respect to the nodes' information capacity, may suggest to user u at most $|R_e(u)| = 5 - 1 = 4$ items and to user v at most $|R_e(v)| = 5 - 2 = 3$ items. If the items i_1, i_2, i_3, i_4 are recommended to u, they will reach v through sharing, which leads to $|R_{im}(v)| = 2 + 4 = 6 > c(v) = 5$. While user u benefits from all the suggested items, the capacity of user v is exceeded, thus the decision-making of v is ruined.

Example 2 - Violation of a user's capacity for the duplicates of a specific item, C.

Let there be users u, v of the network of Figure 4.2, as described above, with capacity for item i_1, $C(i_1, u) = C(i_1, v) = 3$. If the non-neighboring nodes a, b, c, d, g are all recommended of item i_1 from the RecSys, then $|R_{im}(v)| = 3$, since item i_1 will be received from a, d, g, and $|R_{im}(u)| = 4 > C(i_1, u) = 3$, since i_1 will be received from a, b, c, d.

The main objective of this work is to assign items to users in such a way that the explicit recommendations combined with the information flow in the network (implicit recommendations) will result in no violations, both in terms of distinct items and duplicates, while maximizing the total relevance score. This problem is of significant computational difficulty, so it is infeasible to solve it in reasonable amount of time for large graphs and many available items. In order to tackle it, it is chosen to partition the problem into two sub-problems that need to be solved successively. This division of the problem into two "simpler" ones, allows for the development of heuristic algorithms. The combination of these algorithms, produces recommendations of items to users that on the one hand, achieve a high relevance score, while on the other manage to respect the constraints imposed on duplicates and distinct items, as shown in Section 4.4.

4.3.1 Subproblem 1: Find a set of users with high relevance score for an item, while capacity for duplicates is not violated

In this subsection, the problem of finding the set of users that maximizes the total relevance for a specific item i, while satisfying user capacity constraints on duplicate recommendations, is tackled. This problem is equivalent to finding the Maximum Weighted Independent Set (IS) of the graph $G_F^{(i)}$ that satisfies the user constraints. Specifically, the recommendation of item i to user u is defined as follows:

$$x_{u,u}^{(i)} = \begin{cases} 1, & \text{if } u \text{ is explicitly recommended of } i, \\ 0, & \text{if } u \text{ is not explicitly recommended of } i, \end{cases} \tag{4.3}$$

and for $(v,u) \in E_F^{(i)}$:

$$x_{v,u}^{(i)} = \begin{cases} 1, & \text{if } u \text{ is recommended of } i \text{ from } v, \\ 0, & \text{if } u \text{ is not recommended of } i \text{ from } v. \end{cases} \tag{4.4}$$

The matrix $\mathbf{X}^{(i)} = [x_{u,v}^{(i)};\ u,v \in U]$, is defined, which is a $|U| \times |U|$ matrix that models the assignment of item i to users. Then, given the influence graph $G_F^{(i)} = (U, E_F^{(i)})$, the matrix of item-user relevance $\mathbf{W}_{IU}$, and the vector of users' capacity for duplicates, $\mathbf{c}$, the problem of finding the subset of users that maximizes the relevance of item i, is formulated as follows:

$$\mathbb{X}^{(i)} = \underset{\mathbf{X}^{(i)}}{\text{argmax}} \sum_{u \in U} [w_{IU}(i,u) \cdot x_{u,u}^{(i)} + \sum_{v \in RS^{(i)}(u)} w_{IU}(i,v) \cdot x_{u,v}^{(i)}]$$

subject to:

$$x_{u,u}^{(i)} + x_{v,v}^{(i)} \leq 1, \forall v \in RS^{(i)}(u), \tag{4.5}$$

$$x_{u,u}^{(i)} + x_{v,u}^{(i)} \leq 1, \forall u \in RS^{(i)}(v), \tag{4.6}$$

$$x_{u,u}^{(i)} = x_{u,v}^{(i)}, \forall v \in RS^{(i)}(u), \tag{4.7}$$

$$\sum_{u \in U} x_{u,v}^{(i)} \leq C(i,v), \forall v \in RS^{(i)}(u). \tag{4.8}$$

Constraint (4.5) ensures that item i will not be explicitly recommended to neighboring nodes in $G_F^{(i)}$, because that would be unnecessary since only one explicit recommendation would be enough. Constraint (4.6) ensures that a user that has not been explicitly recommended of item i, cannot recommend it to its neighbors in $G_F^{(i)}$. Constraint (4.7) ensures that if an item is explicitly recommended to a user u then it will be implicitly recommended to all the nodes in its reliable set. Finally, constraint (4.8) establishes that the capacity of each user for the duplicates of item i is not violated.

Consider the case of a graph $H^{(i)} = (U, E_F^{(i)}, W_H^{(i)})$, containing the nodes and edges of the influence graph $G_F^{(i)}$ with weights on the nodes equal to $W_H^{(i)}(u) = W_{IU}(i,u) + \sum_{v \in RS^{(i)}(u)} W_{IU}(i,v)$. Solving Subproblem 1 subject only to the first constraint is equivalent to finding a Maximum Weighted Independent Set (MWIS) in graph $H^{(i)}$. Finding a MWIS in a graph is a known NP-hard problem as an extension of the Maximum Independent Set problem with weights on each node of the graph [99]. Thus, Subproblem 1 subject only to constraint (4.5) is NP-hard. The addition of the additional constraints (4.6, 4.7) that monitor the implicit recommendations and of the last constraint (4.8) on the capacity for duplicate recommendations, implies that the produced Independent Set for item i needs to be the the MWIS with the extra property that every node $u \notin IS^{(i)}$, has at most $C(i,u)$ of its neighbors in $IS^{(i)}$. This extra difficulty leads to the formation of heuristic approaches for solving Subproblem 1.

4.3.1.1 GRIS - A Greedy Centralized Algorithm

Due to the computationally difficult nature of the above problem, at first, a greedy algorithm is proposed, in order to cope with the newly added constraint on the number of repetitions of the same item per user. For item i, the algorithm operates iteratively, until an Independent Set of graph $G_F^{(i)}$ is formed. In each loop, the contribution to the total relevance score (RSc) of each node in the candidate list U', which initially contains all the nodes (users) of $G_F^{(i)}$, is (re)calculated so that the relevance score of the nodes that have been excluded as candidates to join the IS is not taken into account for the rest of the process. This marks a different approach from [99] in which the impact to the total score is calculated only once. In the next step of the algorithm, the nodes in U' are sorted using the criterion introduced in [99]. By dividing the relevance score achieved by recommending the item to each user u by $1 + deg_{out}(u)$, where $deg_{out}(u)$ denotes the out-degree of node u, due to the sorting of the users it is feasible to distinguish those that achieve a high relevance score while influencing a relatively small number of users. This kind of users are preferred as candidates for the IS due to two reasons. First of all, their inclusion leads to a high relevance score in aggregate. Also, because they influence relatively few users, their inclusion in the IS results in the exclusion of few users (i.e., their neighbors), allowing more users to be eventually included in the IS, thus more users may receive the item under consideration. Then, for each node in the sorted list it is examined if adding it to the IS is allowed. A node is allowed to be added in the IS if: a) it is not a neighbor (successor or predecessor) of any node in the IS, and b) its successor nodes (followers) do not get overloaded by its addition (because they will "see" the item one more time). If this is not the case, the node is discarded from the candidate list and not added to the IS. Otherwise, the node joins the IS and all its followers and followees are removed from the candidate list. Also, vector $\mathbf{c}'$, which is initially a vector of zeros of size $|U|$, is updated in each iteration in order to keep track of how many times each user gets an implicit recommendation of the item. The steps of the algorithm are provided in Algorithm 2.

Algorithm 2: GRIS - A heuristic for computing a Weighted Independent Set of the influence graph $G_F^{(i)}$ under user capacity constraints for item i.

```
Input  : Influence Graph for item $i$, $G_F^{(i)} = (U, E_F^{(i)})$, $\mathbf{C}^{(i)}$ which is the $i^{th}$ row of
         matrix $\mathbf{C}$, that is, the users' capacity to item $i$, $\mathbf{W}_{IU}^{(i)}$ which is the $i^{th}$ row
         of matrix $\mathbf{W}_{IU}$, that is, the users' relevance to item $i$.
Output: $IS^{(i)}$ for item $i$ of total relevance score $W(IS^{(i)})$
1  Initialization: Independent Set $IS^{(i)} = \emptyset$, Set of nodes recommended of item $i$,
     $S = \emptyset$, $U' \leftarrow U$, $\mathbf{c}'$ is a vector of zeros of magnitude $|U|$.
2  while $U' \neq \emptyset$ do
3      for each $u \in U'$ do
4          Compute the contribution of relevance of node $u$ to the network:
               $$W^{(i)}[u] = \sum_{\substack{v \in N_{out}^{(i)}[u] \\ v \notin S}} w_{IU}(i, v),$$
           where $N_{out}^{(i)}[u]$ is the closed out-neighborhood of $u$ (the set of the
           out-neighbors of node $u$, including node $u$), in the influence graph of item $i$.
5      Sort nodes $u \in U'$ in decreasing order of $\frac{W^{(i)}[u]}{1+n(u)}$, where $n(u) = |N_{out}^{(i)}(u) \setminus S|$, and
        $N_{out}^{(i)}(u)$ is the open out-neighborhood of node $u$ (the set of the out-neighbors of
        node $u$, excluding $u$) in $G_F^{(i)}$.
6      Examine node $u$, which is the first node in the ordered list
7      if $IS^{(i)} \cup \{u\}$ is an Independent Set for $G_F^{(i)}$ and $c'(v) < C^{(i)}(v), \forall v \in N_{out}^{(i)}[u]$
        then
8          $IS^{(i)} \leftarrow IS^{(i)} \cup \{u\}$
9          $W(IS^{(i)}) = W(IS^{(i)}) + W^{(i)}[u]$
10         $c'(v) = c'(v) + 1, \forall v \in N_{out}^{(i)}[u]$
11         $U' \leftarrow U' \setminus N^{(i)}[u]$, where $N^{(i)}[u] = N_{out}^{(i)}[u] \cup N_{in}^{(i)}[u]$ is the closed
             neighborhood of $u$ in $G_F^{(i)}$.
12         $S \leftarrow S \cup N_{out}^{(i)}[u]$
13     else if $IS^{(i)} \cup \{u\}$ is an Independent Set for $G_F^{(i)}$ and $\exists v \in N_{out}^{(i)}[u]$:
        $c'(v) = C^{(i)}(v)$ then
14         $U' \leftarrow U' \setminus N_{in}^{(i)}[v]$, $\{v \in N_{out}^{(i)}[u] : c'(v) = C^{(i)}(v)\}$
15     else
16         $U' \leftarrow U' \setminus \{u\}$
```

4.3.1.2 GAIS, A Genetic Algorithm

Apart from the greedy approach described above, a genetic algorithm was also developed and tested in terms of the achieved relevance score. This algorithm, named GAIS, is displayed in Algorithm 3. The nature of the problem, regarding finding which users should be recommended of an item, leads to the intuition that the solution can be modeled as a chromosome and that a genetic algorithm with a suitable objective function can be an effective approach in order to search the state space of the problem across the generations for a solution.

Algorithm 3: GAIS- A Genetic Algorithm for computing a Weighted Independent Set of the influence graph $G_F^{(i)}$ under user capacity constraints for item i.

```
Input  : Influence Graph for item $i$, $G_F^{(i)} = (U, E^{(i)})$, Vector of users' capacity to
         item $i$, $\mathbf{C}^{(i)}$, Vector of users' relevance to item $i$, $\mathbf{W}_{IU}^{(i)}$ and a chromosome
         mapping $g : b_{j,t} \rightarrow B_{j,t} \subset U$
Output: An Independent Set $IS^{(i)}$ for item $i$ of total relevance score $W(IS^{(i)})$ which
         is the sum of relevance scores of the nodes in the $IS^{(i)}$ and their
         out-neighbors in $G_F^{(i)}$
1  Initialization: $t := 0$, Population $P_0 = \{b_{1,0}, ..., b_{n,0}\}$
2  while $t \leq 1000$ do
3      %(proportional selection)
4      for $j = 1 : n$ do
5          $x := Random[0, 1]$
6          $k = 1$
7          while $k < m$ and $x > \frac{\sum_{l=1}^{k} f(b_{l,t})}{\sum_{l=1}^{n} f(b_{l,t})}$ do
8              $k := k + 1$
9          $b_{j,t+1} = b_{j,t}$
10     %(one-point crossover)
11     for $j = 1 : n - 1$ do
12         if $Random[0, 1] \leq p_C$ then
13             $pos := Random\{1, ...|V| - 1\}$
14             for $k = pos + 1 : |V|$ do
15                 $swap(b_{j,t+1}, b_{j+1,t+1})$
16     %(mutation)
17     for $j = 1 : n$ do
18         for $k = 1 : |V|$ do
19             if $Random[0, 1] \leq p_M$ then
20                 invert $b_{j,t+1}[k]$
21     $t = t + 1$
22 $index = \text{argmax}_z f(b_{z,t})$
23 $IS^{(i)} = g(b_{index,t}) = B_{index,t}$
```

For item i, each chromosome $b_{j,t}$ of the population at generation t, $P_t = \{b_{1,t}, ..., b_{n,t}\}$, represents a set of nodes $B_{j,t} \subset U$ as a possible solution to the problem. It is defined as a vector of zeros and ones of size $|U|$ with

$$b_{j,t}(u) = \begin{cases} 1, & \forall u \in B_{j,t}, \\ 0, & \forall u \notin B_{j,t}. \end{cases} \tag{4.9}$$

Chromosomes are assigned a fitness score according to the function:

$$f(b_{j,t}) = \frac{\sum\limits_{u \in N_{out}^{(i)}[B_{j,t}]} w_{IU}(i,u)}{1 + \mu \times |\text{violations in } b_{j,t}|}, \tag{4.10}$$

where $N_{out}^{(i)}[B_{j,t}] = \bigcup\limits_{v \in B_{j,t}} \{v \cup \{ \bigcup\limits_{(v,u) \in E_F^{(i)}} u\}\}$ is the union of out-neighbors of nodes in $B_{j,t}$ including the nodes of $B_{j,t}$.

Two types of violations are encountered in chromosome $c_{j,t}$:

- The nodes in $B_{j,t}$ do not form an Independent Set, i.e., $\exists\ u, v \in B_{j,t}$ such that $(u,v) \in E_F^{(i)}$. The number of violations of this type in chromosome $b_{j,t}$ is equal to the number of the encountered pairs (u,v).

- The recommendation of item i to the nodes of $B_{j,t}$ exceeds the capacity for this item of at least one node in $N_{out}^{(i)}[B_{j,t}]$. For example, in chromosome $b_{j,t}$, node $u \in N_{out}^{(i)}[B_{j,t}]$, receives d recommendations of item i, while its capacity is $C(i,u) < d$. The number of violations for node u equals to $d - C(i,u)$.

The fitness function is chosen so that it favors chromosomes that have no violations. The constant μ defines the magnitude of the penalty to be assigned to chromosomes that violate the problem's constraints. A small value encourages chromosomes that display just a few violations to pass to the next generation. These chomosomes, although displaying violations, eventually, through the evolution mechanism contribute to the formation of acceptable high quality chromosomes. On the contrary, a large value of μ decreases significantly the chance that unacceptable solutions pass on to the next generation. However, this may lead to a worse evolution process in the long-term because acceptable solutions of high fitness scores are less likely to be obtained by simply combining initially acceptable ones, meaning that the search space is greatly reduced.

GRIS performs well in terms of execution time but suffers from the fact that it cannot always produce the optimal solution (i.e., an allocation of items to users that results in the maximum possible total relevance score).

To give an intuitive example, consider the graph displayed in Figure 4.3. In this simple influence graph of item i, GRIS will output user 0 as a candidate for its recommendation, resulting in a total score of $1 + 6 + 4 = 11$. A closer examination of the graph though, would reveal that the optimal solution is given if i is recommended to users 1, 2, 3 and 4, yielding a total score of $1 + 3 + 1 + 6 + 4 = 15$ (each user contributes to the total relevance score once). It is observed that, in such cases, GAIS can find a better solution, but it suffers from a high execution time when many generations are required for larger graphs.

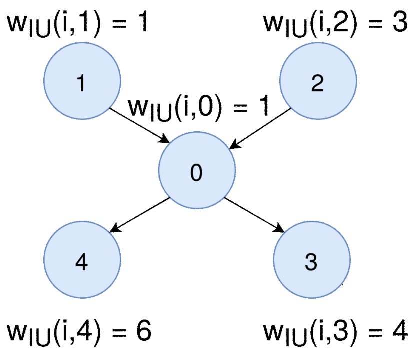

Figure 4.3: GRIS sub-optimality example, assuming no capacity constraints exist.

4.3.2 Subproblem 2: Allocate items to users of high relevance while their capacity for distinct items is not violated

By solving Subproblem 1 for every item of the set I, thus, having the family $\{\mathbb{X}^{(i)}\}_{i \in I}$, the family $\{\mathbf{Y}^{(i)}\}_{i \in I}$ is inferred. $\mathbf{Y}^{(i)} = (y^{(i)}_{i,u})$ is a $|I| \times |U|$ matrix of implicit and explicit recommendations of item i given by the assignment matrix $\mathbb{X}^{(i)}$. The elements, except for the ones in row i, are all zero. The information about the type of recommendation (implicit or explicit) is lost in $\mathbf{Y}^{(i)}$. Each element $y^{(i)}_{i,u}$ is defined:

$$y^{(i)}_{i,u} = \begin{cases} 1, & \text{if } \exists\, v \in U : x^{(i)}_{v,u} = 1, \\ 0, & \text{if } x^{(i)}_{v,u} = 0,\ \forall v \in U, \end{cases} \tag{4.11}$$

and

$$y^{(i)}_{j,u} = 0, \quad \forall j \neq i \in I. \tag{4.12}$$

Given the family $\{\mathbf{Y}^{(i)}\}_{i \in I}$, the vector of users' capacity for distinct items $\mathbf{c} = (c(u);\ u \in U)$ and the rows of matrix $\mathbf{W}_{IU}$, with its i^{th} row to be denoted by $\mathbf{W}^{(i)}_{IU}$, the problem of finding the subset of items $T \in I$ with $\mathbf{Y} = \sum_{i \in T} \mathbf{Y}^{(i)}$ that maximizes the total relevance in the network, is formulated as shown in Subproblem 2:

$$\mathbb{Y} = \underset{\mathbf{Y}}{\operatorname{argmax}} \sum_{i \in I} \mathbf{Y}^{(i)} \cdot \mathbf{W}^{(i)\intercal}_{IU},$$

subject to:

$$\sum_{i \in I} y^{(i)}_{i,u} \leq c(u), \quad \forall u \in U. \tag{4.13}$$

Both GRIS and GAIS address Subproblem 1, which is the problem of maximum relevance assignment for one item without duplicate violations in the network. By solving Subproblem 1 for item i, a set of candidate non-neighboring users who may be recommended of i is determined. Any recommendation of i, either to the whole Independent Set produced as a solution of Subproblem 1 or to any of its subsets, fulfills the above terms. The sets of assignments resulted by solving Suproblem 1 for every item, will be the input of Subproblem 2. The elements of these sets will be combined and filtered with respect to users' capacity for distinct items in order to conclude to a subset that achieves high total relevance score.

Subproblem 2 is addressed by two greedy approaches, an item-centric (RecIt) and a user-centric (RecUs). As mentioned before, due to the propagation of information, candidate user u from $IS^{(i)}$ determines a cover set, that is a set of users to whom item i will be implicitly recommended through u, defined in algorithm GRIS as $N^{(i)}_{out}(u)$. This is denoted by $COV^{(i)}(u) = \{v : \ v \in RS^{(i)}(u)\}$, which is the open cover set of node $u \in IS^{(i)}$. The closed cover set of $u \in IS^{(i)}$ is defined as $COV^{(i)}[u] = \{v : \ v \in RS^{(i)}(u) \cup \{u\}\}$. Thus, for item i, the set $IS^{(i)}$, which is computed by GRIS and GAIS, determines the family of cover sets $COV^{(i)} = \{COV^{(i)}(u)\}_{u \in IS^{(i)}}$.

In algorithm RecIt, presented in Algorithm 4, the families $\{COV^{(i)}\}_{i \in I}$ are sorted in a decreasing order of the relevance scores of their corresponding $IS^{(i)}$, which means that all sets that belong in $COV^{(i)}$ are assigned of the same score. The rationale is that popular item i is given priority for recommendation to all available users determined by $COV^{(i)}$, even to those who have a small contribution to its total relevance score. In order to tackle this issue and increase the diversity of recommended items in the network, the algorithm RecUs is employed which sorts the sets of users belonging to the union of the families $COV^{(i)}$, $\forall i \in I$, denoted by $COV = \{COV^{(i)}(u), i \in I, u \in IS^{(i)}\}$ in a decreasing order of their relevance scores. In that way, items are recommended to subsets of highly relevant available users, allowing the recommendation of less popular items, known as "long-tail items", in the network. A pseudocode for this algorithm is given in Figure 5. In both algorithms, the sets $COV^{(i)}(u), i \in I, u \in IS^{(i)}$ are assigned of item i if the capacity of its users is not exceeded.

In the following section, some indicative results of SCoRe for artificial topologies resembling real social networks are presented. First, the behavior of DifRec is examined that showcases the necessity of the proposed framework for addressing the problem of relevant recommendation of items to users, while respecting strict constraints. Then, the evaluation of the different approaches for the components of SCoRe follows.

Algorithm 4: RecIt - An item-centric heuristic for recommendations of items of high relevance under user capacity constraints. Assignments are based on the relevance scores of the Independent Sets of the family of ISs.

```
Input  : Family of Influence Graphs, $\{G_F^{(i)}\}_{i \in I}$, Vector of users' capacity for
         distinct items, $\mathbf{c}$, Vector of relevance scores of the Independent Sets
         $(W(IS^{(i)}))_{i \in I}$, family of sets of users who received item $i$ from user $u$, $u \in$
         $IS^{(i)}$, $COV^{(i)} = \{COV^{(i)}(u)\}_{u \in IS^{(i)}}$
Output: A matrix $\mathbf{M} = (m_{iu})_{|I| \times |U|}$ of zeros and ones of user-item recommendations.
1  Initialization: $m_{iu} = 0$, $\forall i \in I$, $\forall u \in U$, $U' \leftarrow U$, $I' \leftarrow I$, $\mathbf{c}'$ is a vector of zeros of size
   $|U|$.
2  Sort $IS^{(i)}$ in decreasing order of $W(IS^{(i)})$, $i \in I'$.
3  while $U' \neq \emptyset$ and $I' \neq \emptyset$ do
4      Examine sets in the order sorted above
5      for $u \in IS^{(i)}$ do
6          if $c'(v) < c(v)$, $\forall v \in \{COV^{(i)}[u]\}$ then
7              if $m_{iv} == 0$ then
8                  $c'(v) = c'(v) + 1$
9              $m_{iv} = 1$
10         else
11             $U' \leftarrow U' \setminus \{v \in \{COV^{(i)}[u]\} : c'(v) \geq c(v)\}$
12     $I' \leftarrow I' \setminus i$
```

Algorithm 5: RecUs - A heuristic for recommendations of items of high relevance under user capacity constraints. Assignments are based on the relevance scores of the cover sets of the family COV.

```
Input  : Family of Influence Graphs, $\{G_F^{(i)}\}_{i \in I}$, Vector of users' capacity for
         different items, $\mathbf{c}$, family of the open cover sets of users for all items,
         $COV = \{COV^{(i)}(u)\}_{u \in IS^{(i)}}^{i \in I}$ and a matrix of its corresponding relevance
         scores $[W(COV^{(i)}[u])]_{|I| \times |U|}$
Output: A matrix $\mathbf{M} = (m_{iu})_{|I| \times |U|}$ of zeros and ones of user-item recommendations
1 Initialization: $m_{iu} = 0$, $\forall i \in I$, $\forall u \in U$, $\mathbf{c'}$ is a vector of zeros of size $|U|$,
    $COV' \leftarrow COV$.
2 Sort sets $COV^{(i)}(u)$ in decreasing order of $W(COV^{(i)}[u])$, $\forall u \in IS^{(i)}$, $\forall i \in I$
3 while $COV' \neq \emptyset$ do
4     Examine sets in the order sorted above
5     if $c'(v) < c(v)$, $\forall v \in \{COV^{(i)}[u]\}$ then
6         if $m_{iv} == 0$ then
7             $c'(v) = c'(v) + 1$
8         $m_{iv} = 1$, $\forall v \in COV^{(i)}(u)$
9     $COV' \leftarrow COV' \setminus COV^{(i)}(u)$
```

4.4 Evaluation of SCoRe

In this Section, results will be presented along with a discussion showcasing the strong aspects of utilizing the proposed model for recommending items in an OSN by taking into account the information diffusion between its users as well as complex constraints on information capacities. From the results displayed below, it is evident that by using only a few explicit recommendations from the RecSys, all users of the network are recommended of highly relevant items, while their capacity constraints are not violated. Thus, employing this framework in a Smart City scenario, would be beneficial both for the retailers that would see their applications gain market share and more importantly for the users that will be able to discover highly relevant applications to suit their tastes and improve their quality of life.

4.4.1 Experimental Setup

Complex network models are used to model users' influence, relevance and capacity of distinct and duplicate items. The size of the users' network takes values from the set $S_1 = \{100, 150, 200, 250, 300\}$. The size of the items' network takes values from the set $S_2 = \{20, 30, 40\}$. The capacity for duplicate items of a user is assigned values from the set $S_3 = \{3, 4, 5, 6\}$, whereas the capacity for distinct items is assigned values from the set $S_4 = \{10, 11, 12, 13, 14, 15\}$. The item-user relevance $\mathbf{W}_{IU}$ follows a uniform distribution between 0 and 1, $w_{IU} \sim \mathcal{U}[0, 1]$. The parameter μ of the fitness function in GAIS is set to 1, in order to allow a wider exploration of the search space during the execution of the genetic algorithm. Larger values for the μ parameter have also been tested, but this leads to a limited exploration of the state space because only chromosomes that have no violations are passed to the next generation. Results for these setups are omitted. For each pair of the Cartesian product of S_1 and S_2, 25 distinct scale-free graphs are generated in order to achieve a more comprehensive analysis of SCoRe's behavior and averaged results are displayed in tables and figures below.

4.4.2 Comparison of SCoRe to DifRec framework

At first, the consequences of not considering the more advanced user constraints, described formally in subproblems 1 and 2, in Section 4.3, are presented by measuring the extent of the information overload. To accomplish this, the DifRec algorithm is executed on networks with number of nodes and number of items in $S_1 \times S_2$ as inputs. To set the value K of DifRec, which denotes the maximum number of different items that can be explicitly recommended to a user, the mean of matrix $\mathbf{c}$ is calculated, consisting of values in S_4, which is in all cases equal to 12. In order to quantify the results, $r_d(i, u)$ is defined as the number of times user u has seen item i, $V_C^{(i)} = \{u \in U : C(i, u) < r_d(i, u)\}$ to be the set of users whose capacity to duplicates of item i, as determined by matrix $\mathbf{C}$, is exceeded. $V_c = \{u \in U : c(u) < |R_e(u) \cup R_{im}(u)|\}$ is defined to be the set of users who were recommended of more than $c(u)$ distinct items, where $c(u) = K$, $\forall u \in U$. Table 4.1 summarizes the averaged results obtained for each setup. It is observed that in all cases a user can receive more duplicates than she can handle. Moreover, from this table it can be seen that, the average number of violations in the number of duplicates seen by users sometimes exceeds the average number of users that had at least one such violation, meaning that there are cases in which the decision process of a user can be ruined on more than one item. Also, in every case there exists a non negligible number of users ($10\% - 15\%$) that end up receiving more items than K when taking into account both the explicit and the implicit recommendations. These results showcase the need to tackle efficiently the issue of recommending items to different users through the RecSys, considering also the spread of information in the underlying social network, while applying strict constraints to users.

Table 4.1: DifRec results and violations

$\|U\|$	$\|I\|$	$\sum_{i\in I}\|V_C^{(i)}\|$	$\|\cup_{i\in I} V_C^{(i)}\|$	$\|V_c\|$
100	20	17.36	16.04	13.84
100	30	28.32	25.28	14.48
100	40	42.2	35.44	14.92
150	20	15.76	14.88	19.64
150	30	28.4	25.88	21.06
150	40	39.2	34.52	21.6
200	20	13.96	13.28	25.12
200	30	26.64	24.4	28.16
200	40	38.84	34.32	29.2
250	20	14.64	14.16	33.32
250	30	25.36	23.36	34.92
250	40	37.2	33.36	35.2
300	20	14.2	13.44	40.08
300	30	24.92	23.04	43.48
300	40	36.56	32.64	46.0

In Figure 4.4 the average number of recommended items per user for algorithm RecUs (similar results have been acquired for algorithm RecIt) is presented. It is observed that in all cases the average number of recommendations does not exceed 12, and it should be noted that no user received more duplicates of an item than her capacity. In the following subsections, results obtained using SCoRe in the same topologies are presented in depth and a discussion on the performance of RecIt and RecUs in terms of explicit and implicit recommendations is provided.

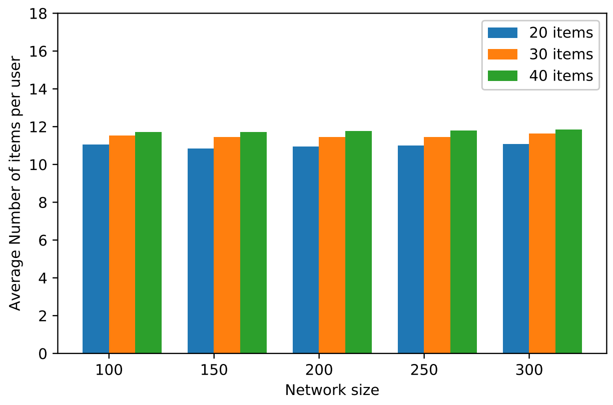

Figure 4.4: Average number of recommended items per user for networks of different size.

4.4.3 Comparison of GRIS and GAIS

The execution time of GRIS and GAIS is compared, as well as the average relevance score of the IS of 40 items, produced over 25 scale-free topologies of the same size $|U| \in S_1$, by these algorithms. The behavioral features of the network (capacity, item-user relevance) remain constant in order to study the effect of the influence network's structure on the IS.

The relevance score of the Independent Set of item i, $IS^{(i)}$, is defined as:

$$W(IS^{(i)}) = \sum_{u \in COV(IS^{(i)})} w_{IU}(i,u),$$

where $COV(IS^{(i)}) = \bigcup_{u \in IS^{(i)}} COV^{(i)}[u]$, whereas the maximum relevance score of $IS^{(i)}$ is defined as:

$$W_{max}(IS^{(i)}) = \sum_{u \in U} w_{IU}(i,u),$$

and it is the relevance score that would be achieved if all users received the recommendation of item i. Figure 4.5 depicts the average relevance score of the Independent Sets realized by GRIS and GAIS and the average maximum relevance score. It is noted that as the network size increases, so does the relevance score achieved by both algorithms, with GRIS outperforming GAIS sharply at high values of network size. In fact, it almost reaches the average maximum relevance score. This means that GRIS computes Independent Sets of users that cover the whole network with the corresponding items.

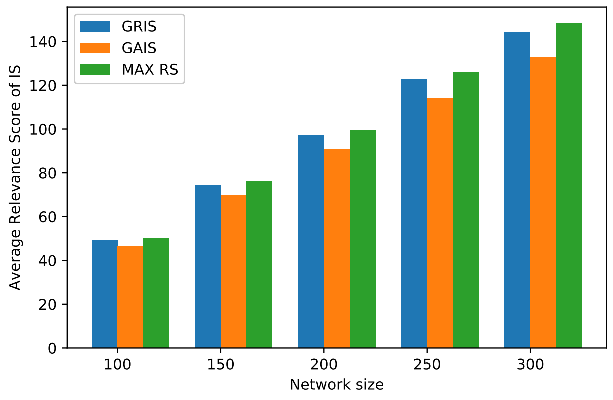

Figure 4.5: Average relevance score of the IS computed by GRIS and GAIS.

In Figure 4.6, it can be observed that the average size of the Independent Sets of GRIS is markedly smaller than the one of GAIS. The fact that the smaller Independent Sets of GRIS attain greater diffusion of items in the network is due to its criterion for user insertion in the Independent Sets: For each item, non-neighboring users with small sized out-neighborhood of high relevance to the former, are promoted as candidates for the Independent Set.

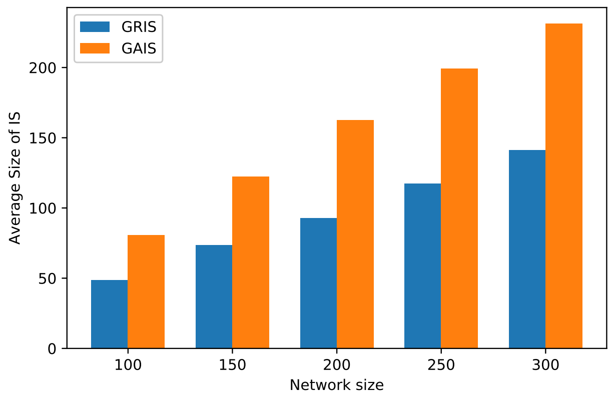

Figure 4.6: Average size of the ISs computed by GRIS and GAIS for 40 items.

In GAIS, no such preference criterion exists. Due to the scale-free features of the influence networks, the out-degree of users follows a power-law distribution (which models adequately the existence of influencers in an OSN), leading to the conclusion that most users of the Independent Set formed by GAIS will be of zero out-degree, thus they will not contribute to the diffusion of items with the mechanism of implicit recommendations in the network.

Regarding the execution time of GRIS and GAIS, as it can be seen from Table 4.2, GRIS is by orders of magnitude faster than GAIS. This is expected as many generations are needed in order for the genetic algorithm to converge to a solution that respects the constraints and achieves high relevance score.

Table 4.2: Average execution time (sec) of GRIS and GAIS for 40 items.

Network Size	GRIS	GAIS
100	0.005	248.104
150	0.012	487.812
200	0.021	496.726
250	0.038	502.403
300	0.058	863.680

4.4.4 Comparison of RecIt and RecUs

Because of the better results obtained by algorithm GRIS in terms of the achieved relevance score as depicted in Figure 4.5 its Independent Sets are used as inputs to the two different algorithms for the final item assignment, RecIt and RecUs. In order to highlight their differences, the algorithms are compared by measuring the extent of recommendations' spread in the network. The average number of both explicit and implicit recommendations in the networks (i.e., if an item is either recommended in any way in a network) is measured. Comparing the bars in Figure 4.7, it can be concluded that more items are explicitly recommended on average when applying RecUs in which cace, almost all available items are recommended to the users, whereas less items are considered in the case of RecIt.

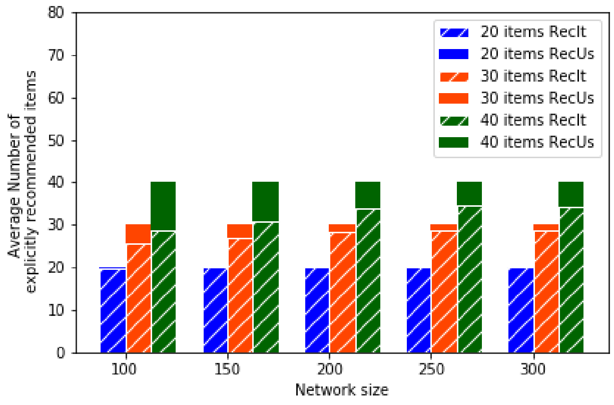

Figure 4.7: Average number of explicitly recommended items in influence networks of varying size with RecIt and RecUs.

Moreover, RecUs leads to more implicit recommendations in the network, as shown in Figure 4.8. This wider diffusion of items in the network by means of both implicit and explicit recommendations leads to greater final relevance score achieved when applying SCoRe with RecUs as it can be seen in Figure 4.9.

This is due to the manner in which the nodes are selected for the recommendation of an item. RecUs operates by sorting the relevance scores achieved by each IS (i.e., for every item) and then proceeds to assign items to sets of users designated by the IS, if there are no violations. This assignment may lead to users not being recommended of their most preferred item but of those that achieve greater score in aggregate. On the contrary, RecUs sorts the sets of users designated by the IS, according to the relevance scores achieved by

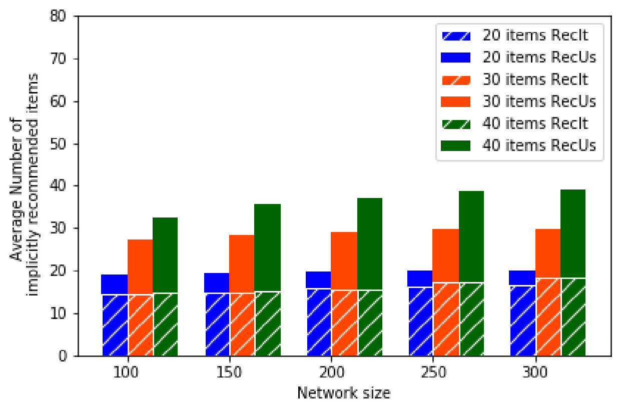

Figure 4.8: Average number of implicit recommendations in influence networks of varying size with RecIt and RecUs.

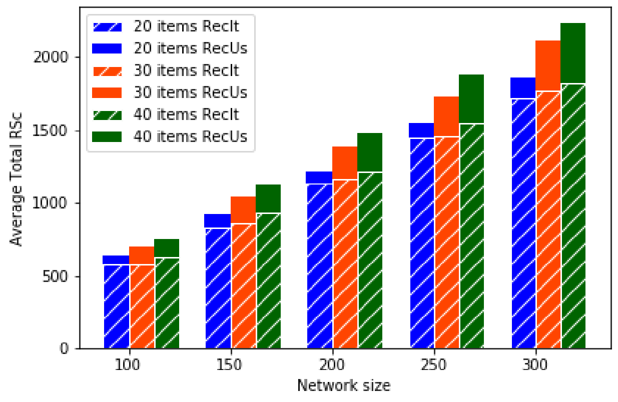

Figure 4.9: Average relevance score achieved by RecIt and RecUs algorithms.

recommending an item to them and their one-hop out-neighbors in the influence graph for all items. This increases the diversity of items that are finally recommended in the network because instead of examining a large set of users for each item (i.e., the ones covered by the IS) relatively small subsets are examined and their score for every item. In this way, items can be recommended to subsets of users that display high relevance for them despite the fact that these users can be included to ISs that may display larger relevance scores for other items.

Finally, from Figure 4.10 it can be seen that RecIt is faster than RecUs. In reality though, both algorithms are quite fast, as on average they terminate in less than 1s.

The above two greedy algorithms mark two distinct approaches to the overall behavior

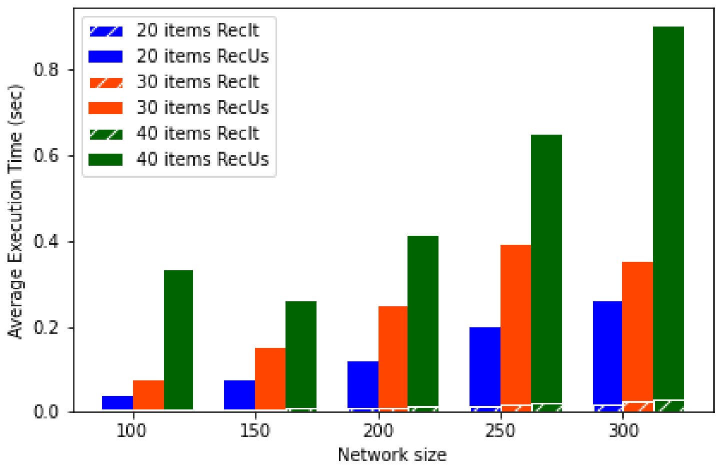

Figure 4.10: Average execution time of RecIt and RecUs.

of the Recommender System. When applying RecIt, the focus is on the more popular items and the aim is to recommend these to as many users as possible. This can lead to further increase in their popularity by dominating the information diffusion process and saturating users. On the other hand, when RecUs is applied, the emphasis is on the users and the RecSys focuses more on satisfying the individual preferences of the users by recommending to them more relevant items, even though these may not reach as many users as in the case of RecIt. This model can be very useful when there are many items that are considered as niche, meaning they display high relevance scores for a few users.

Chapter 5

Increasing Users' Quality of Experience via Caching and Recommendations at the Network Edge

In Chapter 4, the allocation of recommendations in a way that leads to high relevance scores for the users and respects their constraints on the amount of information was examined. Recommendations can be utilized for the spread of applications inside the examined environments, contributing among other things to the increase of the generated data and the further congestion of network links. The 5G topologies will potentially support up to 1 million of connected devices per km^2 [135], making feasible many applications in Smart Cities [136]. As an indication of the data sizes of the future, the forecast of the market advisory company IDC predicts that by 2025 73.1 ZB of data will be generated by smart things [137]. Forecasts like this highlight the need to develop appropriate techniques in order to handle this amount of data. Moreover, these data are necessary for the applications that are executed in the User Equipment (UE, e.g., smartphones, tablets, laptops, smart watches, etc.) to be able to provide the citizens of the Smart City with the latest measurements, allowing them a comprehensive review of what is going on in their places of interest.

Storing content (e.g., video files, sensor measurements, images and so on) at the network edge, i.e., edge caching [138], is a useful practice that can lead to significant improvements in many aspects of the network's performance. To begin with, caching can mitigate the repetitive transmission of the same highly requested data across the network, from the backhaul to the end users. In this way, heavily utilized links are decongested [139, 3]. Moreover, caching in the edge is considered a key enabler in the services of Smart Cities. Storing the necessary data for applications related to smart cities, close to the users can ensure fast delivery to the devices and can also lead to reduced energy consumption [139].

The caching of contents can be achieved by utilizing a limited amount of memory in Base Stations, that lie in the telecommunication cell [3, 4, 140], at dedicated helping nodes that are small devices with limited cache memory scattered across the cell, and also at the UEs (D2D caching) [141, 142]. In the past few years, there has been work toward reducing the required times for the transmission of data among the users' devices [143, 144], making the D2D solution even more feasible.

Moreover, it is a well established fact, that recommendations shape user demands. From published studies, it is revealed that the recommender system is responsible for around 80% of the streamed content on Netflix [145]. Also, about 30% of the overall views on Youtube originate from related video recommendations [146]. Thus, Recommender Systems can be employed in order to nudge users towards requesting cached content that is relevant to their preferences [5, 147]. Deciding the appropriate content to cache at the edge and also generating recommendation lists is treated as a problem of increasing the users' engagement to the platform by providing recommendations of high quality and reducing the content delivery time. In this way, both the users' quality of experience is increased and also the heavily utilized backhaul links are decongested, thus proving a beneficial strategy for the network operators.

At the beginning of this Chapter, related works on caching and on the interplay of caching and recommendations will be discussed. Then, four methods of assisting the caching done in a BS are proposed and evaluated. These methods utilize D2D communications in order to exchange cached content. Following that, a heuristic method for solving a quality of experience maximization problem by leveraging users' mobility patterns for caching at the edge and recommendations will be presented and evaluated.

5.1 Related Works on Caching and Recommendations at the Network Edge

As mentioned earlier, caching at the network edge has been extensively studied by many scientists and has been applied in order to assist in the improvement of many facets of the network's operation. For example, edge caching has been applied in order to minimize energy cost [148]. In this work, the authors propose the combination of caching in devices in close proximity to the end users with multicast broadcasting. In this way, users have access to content that is frequently requested during short periods of time from other users in the network. Although a user participating in this caching scheme may have to suffer some delays, this scheme manages to reduce the required energy costs associated with the repetitive fetching of the same content from the core network. Moreover, in [3] edge caching is considered as an enabler for Smart City applications. In this scenario, specialized network equipment like BSs, Mobile Edge Content (MEC) servers and content caches scattered around the city are used for caching video files. Through the use of Software Defined Networking (SDN) techniques, the system state, a synopsis of the status of all the equipment is given as input in a deep Q-network. By employing machine learning techniques, this network decides on an optimal policy for a certain user that requests access in information through the network's edge. In [149] the authors discuss the incorporation of machine learning techniques for caching at the edge in order to handle datasets of Big Data magnitude. Among other techniques, the clustering of UEs according to preferences, request history etc., similarly to the approaches presented later on this thesis, is also mentioned. The work presented in [150] also performs clustering on the users of the network and introduces the aspect of inter-cluster communication. This type of communication is used for the delivery of content not found inside the requester node's cluster but it is cached in another. The problem of content allocation is formulated as a delay minimization one which is proven to be NP-hard and the authors propose a greedy algorithm for its solution.

In [140], two caching schemes are proposed for storing content at the network edge. The approach aims at minimizing the total expected download delay for a given popularity of files and given the users' and helper nodes' locations. Caching is done in special helper nodes, and as in the case of the proposed algorithms in this thesis, every user has access to a non-empty subset of helper nodes from which it can receive content, if it is available. The first caching scheme is the uncoded one, where complete files (i.e., video files) are stored in the caches. This is also the case in the caching approaches presented in this thesis. The second caching scheme examined is the coded case, where files are segmented and parts of them are stored in each helper node. The problem of caching in the helper nodes is formed as a minimization of the total expected downloading delay. As part of this thesis, the problem of achieving acceptable delay for the offloading of content is tackled in the second part of this Chapter. In the case uncoded files the problem is proven to be NP-hard and a greedy strategy with approximation guarantee is proposed.

Helper nodes are also employed in [151], where the problem of assignment of content in Macro Base Stations and helper nodes is solved by means of stochastic geometry, aiming to maximize the success probability of satisfying user requests. From the analysis presented in the article, it becomes evident that prioritizing the caching of the more popular contents in helper nodes is a sound strategy. Stochastic geometry is again utilized in [152], where the problem of allocating content in BSs that belong in different layers of hierarchy is examined. In this paper, the authors take into account both the physical layer parameters (e.g., SIR thresholds) and the content-related parameters in order to maximize the cache hit ratio probability. For the physical locations of the BSs, a Poisson Point Process is employed, similarly to models considered in this Chapter.

Currently, efficient D2D communication has been made feasible by protocols that enable fast exchange of messages between devices at low energy over licensed (e.g., LTE) or unlicensed frequency bands. Examples of protocols for transmission in unlicensed bands include Bluetooth, that has a transfer speed of 25 Mbps over a distance of at least 200 feet and WiFi Direct that operates with speeds up to 656 feet [153]. Moreover, there exist multiple cognitive radio protocols allowing the communications of nearby nodes by opportunistically accessing licensed bands when they are vacant [154].

Besides storing content in special nodes lying inside a telecommunications cell, the usage of small portions of memory of UEs is discussed in numerous works. In [141] the authors propose a cooperative cache placement scheme in D2D cellular networks where the devices can transmit and receive data. This paper examines the problem by taking into account not only the popularity of items but also whether it is possible to send the requested data within feasible time. The problem is solved by difference of convex programming with an efficient algorithm. The authors prove that D2D caching is a viable solution in order to reduce the traffic in the core of the network. Another approach in D2D caching, is the one presented in [142], where devices cache content according to a certain probability that abstracts their popularity. An interesting distinction is also mentioned, where UEs can either transmit data or request it, but not at the same time. Also, the transmission reliability of content is taken into account and the focal point of the work is that besides the cache hit probability, the success probability of cached content delivery is a major factor that should be considered when designing edge caching solutions. The problem of optimizing the probabilistic caching placement is formulated as a cache hit maximization one, aiming at maximizing the probability of successful requests (i.e., cache hits). Moreover, the authors discuss the benefits of utilizing proactive caching in D2D caching schemes, a point that is also investigated in this thesis.

In addition, the usage of the UEs in caching schemes is discussed in [4]. The social features of the users are taken into account when proactively caching content in UEs, aiming to enhance the cache hit ratio through D2D communications. The proposed scheme identifies influential users according to their centrality in the social network, which is similar to the approach adopted in this thesis. Moreover, in [155], the authors compare mobile caching via D2D connectivity with local caching at the radio access network edge. Their results show that in dense networks, D2D caching may serve more user requests through cache-assisted D2D communication, whereas edge caching results in more cache hits due to great capacity of storage units in Small Base Stations (SBSs). In this thesis the advantages and limitations of both approaches are taken into account in order to develop the proposed methods described in the following sections. Aiming at improving content distribution in the network, in [156], data is cached in both the BS and the UEs via a second price sealed auction. This process manages to avoid the caching of the content in users that are in close proximity ensuring in this way greater variety in the cached content accessible by users residing inside the transmission range of the UE caches.

Limitations of UE caching, such as its relatively small storage space and short battery life, must be taken into account according to [157]. Multiple caching strategies are presented, examining the possible ways in which the requested content can be delivered, like cooperative D2D delivery and multihop D2D delivery by means of relay D2Ds, facilitating the content dissemination. In this paper, the technique mentioned as neighbor D2D delivery is applied. The authors present an architecture of multiple layers consisting of SBSs and UEs and utilize multihop D2D communication in order to reduce energy consumption and prolong the UEs' lifetime. In all the developed caching strategies presented in this Chapter, the storage limitations of UEs are taken into account, either by selecting to cache content to the ones with large memory storage or by considering global participation of UEs in caching, where each UE's contribution is proportionate to its capacity.

The need to take into account the user mobility as an intrinsic feature of wireless networks is highlighted in [158], which builds upon the work presented in [159]. In the latter, small base stations (SBSs) are geographically deployed over an area with non-overlapping coverage. Users come into contact with the helping nodes for a limited amount of time and then content can be offloaded. The SBSs fill their memories with parts of files by solving multiple knapsack-type problems, aiming to minimize the fraction of content served by the main base station. They treat the caching placement problem as a problem of monotone submodular maximization over a matroid constraint and design a greedy approximation algorithm for its solution. The authors in [160] propose a coded caching scheme for leveraging users' mobility. Similarly to the developed algorithm presented later on, they consider that users may receive content from D2D caching if the expected delay is less that a T_d threshold. In that case, the Macro Base Station will deliver the content. The problem is formulated as cache hit maximization one. The problem is proven to be NP-hard and a greedy algorithm is applied for deciding the cached contents in each device. Increasing the variety of the possible UE employed for caching, the authors in [161] propose two mobility-aware cooperative edge caching schemes, one including vehicles for caching and one without. They aim to minimize the average latency of the downloading process and employ a game-theoretic approach for its solution. They prove that caching with the use of mobile edge caches that move at a fast pace around the city, leads to reduced latency while improving the caching resource utilization.

The impact of recommendations on user content requests is investigated in [162]. The authors aim to increase YouTube's caching efficiency by shaping the video demands of its users through recommendations. They introduce a reordering of viewers' related lists so that cached content is presented before non-cached content. In order to provide recommendations that contain cached content and that are relevant to each user, the authors in [163], propose CABaRet. This algorithm searches via Breadth First Search, the related videos of the original related video list that is provided by the YouTube API after watching a video, and recommends the most related cached ones to each user. Although works like these take into consideration the cached contents, they treat caching and recommendations as to completely separate processes. The problem of caching and recommendations is addressed jointly in [147] and [5]. The objective in [147] is to maximize the cache hit ratio by forming user demands towards cached content via recommendations. The authors present a heuristic algorithm that first places content in caches by considering the impact of recommendations and then makes recommendations that promote the cached content. Closer to the work presented in this Chapter, in [5], the authors study the problem of content allocation and recommendations as a problem of QoE maximization and they propose a polynomial-time algorithm with approximation guarantee for its solution. In this thesis, this work is extended from the viewpoint of user mobility, delivery delay tolerance and D2D caching.

Inspired by these works, this Chapter is divided in two parts. In the first one, several caching schemes are developed, examined and compared in terms of the achieved cache hit ratio. Reactive and proactive schemes have been developed and compared. The problem of content placement is modeled as a cache hit maximization one and solved through greedy algorithms. Following that, in the second part of the Chapter, the problem of caching and recommendations leveraging users' mobility is treated through a greedy heuristic algorithm, able to produce high quality results in terms of users' quality of experience in a timely manner compared to the approximative method presented in [5].

5.2 Examining Strategies of Static D2D Caching for Smart Interconnected Environments

In order to design an efficient caching scheme, one needs to answer the following questions: "*where to cache*", "*what to cache*" and "*when to cache*". The proposed algorithms, described later on, provide answers from various perspectives to the previous questions, aiming each time to maximize the achieved cache hit ratio. All the developed approaches answer the "*where*" question by allocating items both in BSs and UEs aiming to take advantage of the D2D communications in order for more requests to be satisfied and mitigate the need to access the more congested backhaul links. Regarding the issue of "*what*", it is decided that the most popular items are the ones that display the highest probability of being cached either inside the BS's memory or in the UE caches. Finally, answering the "*when*" question, both reactive and proactive caching schemes are developed, evaluated and compared. In the proactive schemes, the preferences of users to the items (i.e., how possible is that a user will request an item) are taken into account so that each user stores items that display high probability to be requested in their vicinity. On the other hand, in the reactive approach, each node, selfishly stores and updates the contents of its cache memory only according to its own requests.

As discussed above, in Smart Cities, caching is an integral part that ensures smooth operation and fast access to data for its users. In the following, although the techniques applied focus on caching units of information of higher level, meaning content that can be recommended by a RecSys (e.g., video files, images, news articles, etc), the same techniques can be applied in a "lower" level where the cached content are measurements of sensors that are stored either in the UEs or inside special centrally located sensors, that are accessible by many users.

5.2.1 Modeling Caching at the Network Edge

In the analyzed system there exists a content provision platform constituted by a set of users $U = \{u_1, ..., u_n\}$ who interact by sharing content. The data produced by the participation of users in the platform are used to derive the relevance of user $u \in U$ to different content. Similarly to [164], this is modeled by a feature vector $\mathbf{s}_u$ with $|\mathbf{s}_u| = B$. The j^{th} element of $\mathbf{s}_u$ reveals the preference of u to the content classified into thematic category j, $j = 1, ..., B$, derived by the operation of a RecSys in the platform. The value of B reflects the degree of resolution for the content classification. Users' similarity denoted by $sim(u, v)$ $\forall u, v \in U$, is derived by applying a similarity measure to the corresponding feature vectors, which, in this work is the cosine similarity [165]. For a small time period during which content popularity, hence, the patterns of content requests, may be considered fixed [166], an isolated cellular cell with a single BS is observed. A set of users, $V \subset U$ is located within its range of communication. The limited time frame of observation allows us to assume that users in V form a static network.

5.2.1.1 Content

The items of the content catalogue $I = \{i_1, i_2, ..., i_k\}$ can be any type of multimedia content or information data. Each item $i \in I$ has a finite size $M_i \in \mathbb{N}$ and it is described by a feature vector $\mathbf{s}_i$, with $|\mathbf{s}_i| = B$, which is equivalent to the users' feature vectors. Users' preference to items, $sim(u,i)$, $\forall u \in U, \forall i \in I$ is inferred by the corresponding feature vectors and measured with cosine similarity. The global utility of item i serves as an indicator of the item's popularity across the whole platform and it is defined as follows:

$$U_g(i) = \sum_{u \in U} sim(u,i). \tag{5.1}$$

5.2.1.2 Cache Memories

The cell under examination has a BS with a cache of capacity $c_{BS} \in \mathbb{N}$. The users' devices are equipped with on-board caches of different sizes for which it holds that $c_{UE}(u) < c_{BS}$, $\forall u \in V$.

5.2.1.3 Similarity Graph

The similarity between the platform's users in the BS cell under examination, is modeled by the graph $H(V, E_H)$. The edges in E_H connect the users who have a similarity score above a certain threshold θ. This threshold can be specified according to the objective of the application and controls the density of the similarity graph H. Low values of the similarity threshold result in dense graphs of moderately similar users, whereas high values result in sparser graphs, in which two users are unlikely to be considered similar unless they exhibit the same tendency in content preferences. In practical applications the threshold parameter will be a control factor of the system owner.

5.2.1.4 Location-based Graph

Each user $u \in U$ has an effective radius R_D that denotes the maximum distance at which he can transmit or receive data (e.g., messages, videos, etc.). Users lying within each other's range, determined by R_D, are able to exchange content directly through D2D communication. At the considered time interval, the users in V are the ones connected to the BS, communicating with it in a cellular mode in order to access items that are stored in its cache. Assuming the same communication radius for all the users' devices, the network $L(V, E_L, R_D)$ is formed. A connection $(u, v) \in E_L$ exists if $d(u, v) \le R_D$, where $d(u, v)$ is the Euclidean distance between users u and v. The opportunistic network L is modeled by a Random Geometric Graph (RGG), which according to various previous studies [167] is determined to be a realistic representation of the users' devices location and their connections.

5.2.1.5 System Graph

The intersection of the similarity graph H and the location-based graph L, results in the System Graph $G(V, E)$ with $E = E_H \cap E_L$. The edges of this graph denote the connections of the users that are closely related in terms of content preferences, while also being geographically close to exchange content in a $D2D$ fashion. An example of the system model used in this paper is displayed in Figure 5.1.

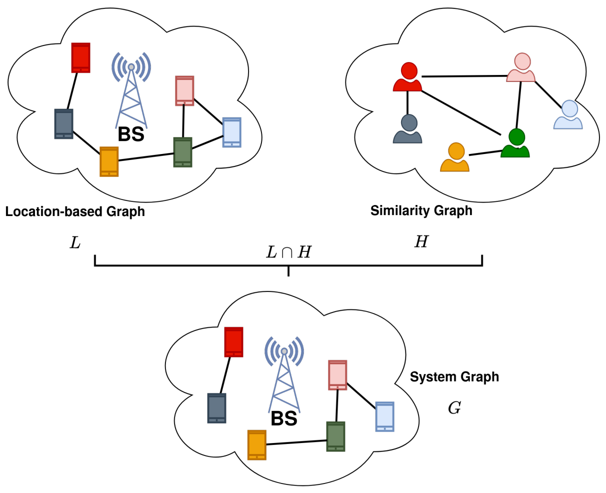

Figure 5.1: The system graph G obtained as $G = H \cap L$ from the similarity graph H and the location-based graph L.

5.2.1.6 User Requests Model

Based on the user-item similarity score $sim(u,i)$, a probabilistic model of requests is formulated, where user $u \in V$ chooses at random the items she will request with a probability proportionate to $sim(u,i)$. This is achieved through fitness proportionate selection to her r most preferable items. The set I_u with $|I_u| = r$ is the set of r items for which user u exhibits the highest similarity score. For user $u \in V$, the fitness of each item $i \in I_u$ is the probability derived by normalizing $sim(u,i)$ so that $\sum_{i \in I_u} sim(u,i) = 1$. User requests can be served in three possible ways:

- The requested item is stored in BS cache, or
- it is stored in the cache of a neighboring user, hence it is accessible via D2D communication, or
- it is not cached and it will be retrieved from the core network.

5.2.2 Formulating the Cache-hit Maximization Problem for Different Static Caching Strategies

In this section, the different approaches to caching in the UEs are discussed. The problem that each one addresses is defined formally and then the corresponding algorithm is presented. Three algorithms are developed, PL-CAUSE (i.e., Proactive Local Cachicng aided by User Equipment), which proactively stores contents in selected UEs denoted as CHs, Proactive Global CAUSE (PG-CAUSE) and Reactive Global CAUSE (RG-CAUSE) that cache contents in all the UEs proactively and reactively respectively. A visual overview of the considered approach and the proposed methods can be seen in Figure 5.2.

5.2.2.1 PL-CAUSE: Proactive Caching in Limited UEs

In PL-CAUSE, the users' similarity is exploited in order to predict their future demands in content and thus, optimize caching efficiency in terms of cache hit ratio. The problem of limited UE caching is divided into two subproblems. At first, the UEs that will be used as caches based on users' similarity and proximity are selected. Aiming to distribute the caches evenly in the network, the problem of selecting which nodes should cache content is mapped

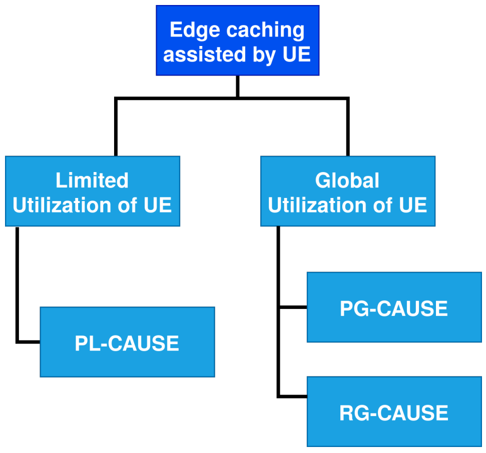

Figure 5.2: Overview of the proposed methods.

to a community detection one, since finding communities in the system graph is equivalent to identifying groups of users who are similar and geographically close. It should be noted that the terms community and cluster are used interchangeably throughout this work. A network partition is obtained in both algorithms, by applying the modularity maximization method described in [27]. A high value of modularity indicates a good community structure, thus, the partition corresponding to the maximum value of modularity on a graph is expected to be of good quality. The users may access cached content either from the BS or from specific neighbors belonging to their community, who are referred to as Cluster Heads (CHs). Next, items are placed in both BS and UE caches so that the cache hit ratio is maximized.

Selection of UE Caches.

The problem of selecting UE caches is treated as a problem of graph partitioning and coverage in G. The network G is partitioned to communities of homogeneous in taste users by applying a modularity maximization method [27]. Due to modularity's resolution limit, communities may differ significantly in size, therefore, a different number of UEs should be delegated for content caching to each community so that the majority of network users are covered with cached content. The number of CHs in a community will be determined by its corresponding size. Assuming that the outcome of modularity maximization is a set of k clusters, $\mathbf{A} = \{A_1, ..., A_k\}$, $\lceil p \cdot |A_i| \rceil$ is specified to be the number of CHs in cluster A_i, where $|A_i|$ is the number of users in community A_i and $p \in [0, 1]$ is determined through simulations to ensure that CHs cover most of the users in their community, i.e., they are connected, in aggregate, to most of the users in their community. The UE cache selection scheme takes into account not only the nodes' degree centrality in the respective communities' subgraphs, but also their storage capacity c_j, $j = 1, ...|V|$. A centrality measure which gives prominence to CHs with a good ratio of capacity to degree, while achieving low overlap between their neighborhoods in the corresponding community is defined. Let A_i be the community under examination. The set of its CHs is denoted by C_{A_i}. Community A_i forms the network $G_{A_i}(V_{A_i}, E_{A_i})$, with E_{A_i} to be the set of edges between the users in V_{A_i}. Given the average weight $\overline{M}$ of items in the content catalogue, the average number of items that can be assigned for caching to u_j is equal to

$$\tilde{c_j} = \frac{c_j}{\overline{M}} \tag{5.2}$$

The closed free neighborhood of u_j in community A_i, is the set of the neighbors of u_j in community A_i, who are not connected to any other CHs of A_i,

$$N_F[u_j|A_i] = \{u_k : (u_k, u_j) \in E_{A_i},\ (u_k, a) \notin E_{A_i},\ \forall a \in C_{A_i}\}.$$

The average number of items per user in $N_F[u_j|A_i]$ is

$$\frac{\tilde{c_j}}{|N_F[u_j|A_i]|}.$$

The centrality of a node u_j is defined as:

$$f(u_j) = \begin{cases} c_j - \frac{c_j}{|N_F[u_j|A_i]|}, & \text{if } lb \leq \frac{\tilde{c_j}}{|N_F[u_j|A_i]|} \leq ub, \\ 0, & \text{otherwise.} \end{cases} \tag{5.3}$$

The lower and upper bounds lb, ub are determined by the density of the networks under examination and the considered UE capacities. This will be further discussed in Section 5.2.4. The centrality metric f rewards users of high degree, when the average number of items per user in the closed free neighborhood of u_j is at least lb and less than or equal to ub. Also, it penalizes small degree and large capacity, i.e., users who have few or no neighbors in their community. The criterion for CH selection in a cluster is formulated as follows: For community A_i, while the number of CHs does not exceed the predetermined threshold $\lceil p \cdot |A_i| \rceil$, it is selected, among the users in A'_i, which is the subset of users in A_i with non-zero centrality values, the one who maximizes f, i.e.,

$$\operatorname*{argmax}_{u_j \in A'_i} f(u_j).$$

Each time a user is selected as a CH, the centrality scores for the rest of the users in the corresponding community are recalculated.

Content placement. Based on the distribution of content preferences, the items are allocated in both BS and UE caches so that the cache hit ratio is maximized. Content placement in PL-CAUSE was formulated as a Multiple Knapsack Problem and it was addressed by solving, successively, for the BS and for each CH, a $0-1$ Knapsack problem. The set of cached items in the BS is denoted as I_{BS}. Then, the candidate items for caching in UE is $I_{UE} = I \setminus I_{BS}$. Each item has different profit for each CH, which is determined by the content preferences of its closed neighborhood in the community it belongs to. Hence, the problem of optimal placement in CHs becomes an instance of a variant of the Multiple Knapsack Problem [168], referred to as the Unbounded Multiple Knapsack Problem with Variable Profits.

A utility function models the popularity of items in the community-defined neighborhood of CHs. The utility of item i to CH u at the network G_{A_k} of community A_k, is given by the expression:

$$H(i, u, A_k) = \sum_{v \in N[u|A_k]} sim(v, i), \tag{5.4}$$

where $N[u|A_i]$ is the closed neighborhood of u in G_{A_i}. For simplicity reasons, the utility of i to u will be denoted by h_{iu}.

Content Placement in BS. The content placement in the cache memory of the BS is formulated as a knapsack problem and solved by applying the algorithm presented in [169], in order to acquire an $1-\epsilon$, $\epsilon > 0$ approximation of the optimal solution of the KP and the process can be seen in the form of the pseudocode provided in Algorithm 6.

Content Placement in CHs. Given the set of CHs, $\mathbf{C} = \{b_1, ..., b_d\}$, the Unbounded Multiple Knapsack Problem with Variable Profits is addressed by solving $0-1$ Knapsack problems successively. An iterative algorithm is chosen, which, at iteration t, assigns content from the set of items I_{UE} to CH b_t, by solving a $0-1$ Knapsack problem. Let $V_i(t)$ be the

Algorithm 6: Algorithm for Content Placement in BS.

```
Input  : Set of items $I$, matrix of user-item similarity $\mathbf{S} = [sim(u,i)]$, BS cache size
         $c_{BS}$, matrix of item weights $\mathbf{M}$
Output: Set of items cached in BS, $I_{BS}$
1 %calculate the global utility of items
2 for $u \in N$ do
3     for $i \in I$ do
4         $U_g(i) += sim(u,i)$
5 % Use Dynamic Programming Approximation Algorithm (DPAA) to compute the set
  of items $I_{BS}$ to store in $c_{BS}$.
6 $I_{BS} = DPAA(\mathbf{U_g}, \mathbf{M}, c_{BS})$
```

set of users, who, until iteration t, have received item i via one or more CHs in $\bigcup_{t'<t} b_{t'}$. The utility of item i to clusterhead b_t of community A_k takes into account if item i has been cached to another CH at $t' < t$. Therefore, the utility of item i to b_t is expressed as follows:

$$h(i,t) = H(i, b_t, A_k \setminus V_i(t)) \tag{5.5}$$

The iterative algorithm, at time t, equivalently to the content placement in BS, given the capacity of b_t and the utility $h(i,t)$, $\forall i \in I_{UE}$, solves with the dynamic programming FPTAS algorithm [169] the $0-1$ Knapsack problem for the CH b_t.

5.2.2.2 PG-CAUSE: Caching in all the UEs

Extending the approach presented PL-CAUSE, the focus is transferred from approaches that utilize a limited subset of UEs, that satisfy certain conditions discussed earlier, towards a scheme that employs all of the existing users lying inside a single cell. In this scheme, all the UEs contribute by dedicating a limited amount of cache memory for the proactive storage of items. The items that are stored in each user's memory are selected by the platform so that they are the ones that fill in the available space, while at the same time maximizing the sum of the local utility scores for its one hop neighbors in the system graph G. This measure, is defined as:

$$U_l(u,i) = \sum_{v \in N[u|G]} sim(v,i) \quad \forall u \in V, i \in I. \tag{5.6}$$

This measures the popularity of item i in the closed neighborhood of user u in graph G. The set $N[u|G]$ consists of user u and her neighbors in G, that is, the users in her transmission range to whom she is also similar. As previously stated, two nodes u, v are considered similar if and only if the cosine similarity on their corresponding feature vectors $\mathbf{s}_u, \mathbf{s}_v$ exceeds a certain, pre-specified, threshold, θ.

In order to achieve a meaningful allocation of items to users the following maximization problem is defied, which is a variant of the Multiple Knapsack Problem, where each item can be assigned to all knapsacks and its profit varies depending to which knapsack is assigned. This problem is referred to as the Unbounded Multiple Knapsack Problem with Variable Profits and it is formulated as follows:

$$\underset{\mathbf{x}}{\operatorname{argmax}} \sum_{u \in V} \sum_{i \in I} x_{ui} \cdot U_l(u, i), \tag{5.7}$$

subject to:

$$\sum_{i \in I} M_i \cdot x_{ui} \leq c_{UE}(u), \quad \forall u \in V, \tag{5.8}$$

where:

$$x_{ui} = \begin{cases} 1, & \text{if item } i \text{ is cached in user } u, \\ 0, & \text{otherwise.} \end{cases} \tag{5.9}$$

By solving the above maximization problem the goal is to allocate the most relevant items to every user according to their preferences as well as those of their neighbors while respecting the constraint of the UE limited capacities. In order to solve the above problem a heuristic algorithm named Proactive Global CAUSE (PG-CAUSE) is proposed.

Algorithm 13 presents the operation of PG-CAUSE. First of all, the items selected to be stored in the BS are removed and are not considered again for possible caching in any UE, as that would be redundant. Then, for every node of the system graph G its degree is calculated and the list D contains the identities of the nodes sorted in descending order. The examination of the nodes in descending order allows to first fill the nodes that satisfy more users and then care for the overlap of caching the same items in neighboring users. To address the latter issue, the preference for each item is set to 0 for every user who can retrieve it from a neighboring node in G, so that it does not impede with the selection of

Algorithm 7: PG-CAUSE Proactive caching in UEs

```
Input  : Matrix $\mathbf{U_l}$ of preferences, set of user caches $\mathbf{c_{UE}}$, System Graph $G$, $I$ items
         available in the platform, $I_{BS}$ items stored in BS cache, $\mathbf{M}$ matrix of item
         weights.
Output: Binary matrix $\mathbf{X}$ of content placement $x_{u,i}$, where $u$ is a node and $i \in I$
1  $I = I \setminus I_{BS}$, % do not consider the items stored in the BS as possible candidates to
     be cached in UEs
2  Set $U_l(u,i) = 0 \;\; \forall u \in V, i \in I$
3  D: sorted list of nodes according to degree in descending order
4  for $u \in D$ do
5      for $v \in N[u|G]$ do
6          for $i \in I$ do
7              $U_l(u,i) += sim(v,i)$
8      $U_l(u,i) += sim(u,i)$
9      $I_u = DPAA(U_l(u,I), \mathbf{M}, c_{UE}(u))$
10     for $i \in I_u$ do
11         $x_{ui} = 1$
12         for $v \in N[u|G]$ do
13             $U_l(v,i) = 0$
```

items for the rest of the users. This is addressed in lines 10-13 of the algorithm. During the time that the users remain active in the platform a request is targeted first at the Base Station and if the content is not available there, then the request is targeted to all the UEs that are neighbors of the user in the location graph L, in order to increase the possibility of the request being satisfied by a UE in the user's proximity.

Regarding the time complexity of the above algorithm, the sorting of list D in line 3 can be done with any of the most efficient sorting algorithms (e.g., quicksort) thus it is of complexity $O(|V| log|V|)$. Then, the loop beginning at line 4 and ending at line 13 has a complexity of $O(|V| * avg.deg(G) * |I| + |V| * |I| * c_{UE})$, taking into account that the complexity imposed by the last loop is negligible, where $avg.degree(G)$ is the average degree of the system graph G. From the above analysis, it follows that the complexity of the algorithm depends on the density of the system graph G. In the extreme case that G is a complete graph, the complexity of the algorithm is $\Omega(|V|^2|I|)$. If G is sparse, the largest impact in the complexity of the algorithm comes from the Knapsack problem concerning the content placement in the UEs, so the complexity of the algorithm is $O(|V| \cdot |I|)$.

5.2.3 Reactive Caching in UEs

Another approach, radically different from those described earlier, is the reactive scheme presented in this section. Instead of some content being proactively cached in a subset of (or all) the UEs in order to assist the BS, reactive caching relies only on the users' requests. That relieves the strain on the content provider to track constantly the users that lie inside each telecommunications cell and make decisions about which content to store in each UE in a timely manner. Each user u is considered to cache the content she requested if there is enough cache memory space left. Otherwise, if the requested item matches her preferences more than those already stored inside her UE (i.e., displays higher similarity score), it replaces the minimum number of items which she likes the least (i.e., have the lowest $sim(u,i)$ score) in her cache. Moreover, each time a user requests an item, the BS cache memory, as well as the other UE caches lying in transmission range, are checked. If the item is retrieved from another UE cache then the user proceeds to store it too in her cache following the process described earlier. The modifications of the users' cache memories impose on the one hand a computational cost related to handling the swaps of content as well as wearing off the memory itself, but on the other hand, assist in the delivery of an item across the network over paths of similar users. The developed algorithm that implements the above procedure is named Reactive Global CAUSE (RG-CAUSE).

5.2.4 Evaluation of Static Caching Policies

In this section, numerical results demonstrating and evaluating the performance and trade-offs of the different methods of $D2D$ caching, discussed in Section 5.2.2, are presented. Firstly, the performance of the proactive scheme when utilizing a limited number of UEs is examined. Next, the two distinct proactive and reactive caching approaches of global UE utilization are compared in terms of the achieved cache hit ratio and discussed further in terms of suitability for caching in the edge.

5.2.4.1 Experimental Setup for Static Caching Policies Evaluation

For the experimental design and execution synthetic graphs are employed. In order to model the physical locations of users inside a cell the random geometric graph model is applied, where nodes are randomly sampled from the uniform distribution of the euclidean space $[0, 1)^2$ and have radius equal to 0.2, which was deemed as a radius that allows the connection of users located near each other (e.g., workplaces, means of public transport, etc). The number of thematic categories, B, is set equal to 10 for all the experiments. The feature vectors $\mathbf{s}_i$ and $\mathbf{s}_v$, denoting the relevance of items and users to the B thematic categories, follow the Pareto and exponential random distributions, respectively. The Pareto distribution with parameter $\alpha = 0.5$ was chosen in order to reflect the fact that an item displays greater relevance for some thematic categories. For example, a tennis ball may belong (i.e., display high relevance) to both the categories "sports" and "pets". On the other hand, the users are considered to display higher relevance to a smaller set of distinct categories. For this, the exponential distribution with parameter $\lambda = 2$ was considered as a suitable distribution to model such behavior.

Each item is considered to have a size that it is chosen randomly among $\{3, 4, 5\}$, while the UE cache sizes, c_{UE}, take integer values from the interval $[5, 10]$ and the BS cache size, c_{BS}, is set to 100. Regarding the similarity graph H, the threshold θ above which two nodes are considered similar is set to 0.8, allowing the formation of a graph where only very similar users are connected with each other. In the following experiments, 5000 users are considered to be the total number of users of the platform, and simulations have been conducted for $|V| = \{100, 200, 300, 400, 500\}$ number of users and $|I| = \{50, 100, 150, 200\}$ number of items. All results discussed in the following subsections are averaged over 10 different networks for each setup.

Regarding the process of simulating the users' requests to the items, every user is considered to make 5 requests out of her 10 most preferred items using proportionate selection according to their similarity to the items.

5.2.4.2 PL-CAUSE

Determination of lb, ub thresholds for the CH centrality metric, f. The parameters lb, ub of the centrality metric f, introduced in Section 5.2.2.1, are determined by taking into account the average weights of items, the UE capacities and the density of the system graph G, which is produced with similarity threshold $\theta = 0.8$. Due to the high value of θ, a great number of edges in the location-based graph L are pruned, resulting in a sparse system graph G, as it is observed in Figure 5.3, where the average degree in system networks of 100 users is 0.84, reaching the value of 4.45 in system networks of 500 users. The items' average size is $\overline{M} = 4$, therefore the average number of items that users can store in their devices' cache is calculated with Eq. 5.3 and takes values from the set $\{1.25, 1.5, 1.75, 2, 2.25, 2.5\}$. By setting $lb = 0.5$ and $ub = 2$, the search space for CH candidates is reduced to the set of users with degree close to the average degree of the networks under examination (greater than 0 and less than or equal to 5). For example, for $lb = 0.5$, UEs which can store at most 1 item in their caches will be in the set of candidate CHs only if they have one or two neighbors, and since $ub = 2$, UEs with capacity of 2.5 items will be CH candidates if they have at least 2 and at most 5 neighbors.

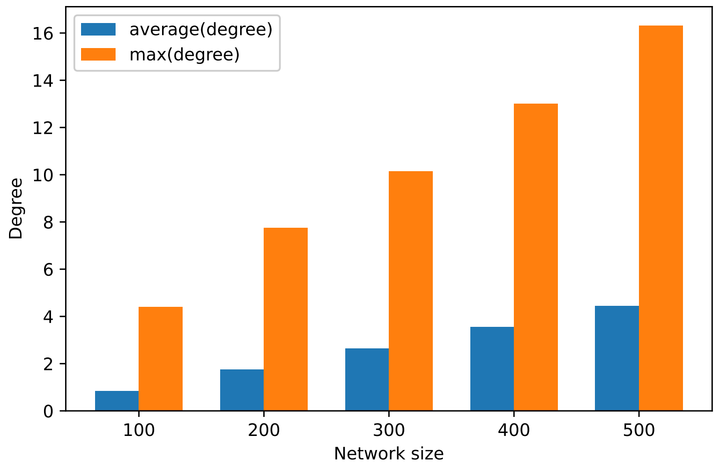

Figure 5.3: PL-CAUSE: Maximum and average degree of nodes in similarity networks of different size, produced with similarity threshold θ=0.8.

Impact of network size on the number of CHs The impact of network size on the number of detected communities is presented in Figure 5.4. As the network size increases, the probability of a user finding similar users at a small distance, increases as well, resulting in a dense topology. In dense networks, the modularity maximization algorithm is known to detect few communities of big size [47], in which many nodes can be designated for content caching, as observed in the networks of 500 nodes in Figure 5.4. On the contrary, in small networks ($|V| < 200$), due to their sparsity, a partition at which the majority of the communities consist of a single node is generated. Disconnected nodes cannot serve as CHs and rely solely on the BS for cached content delivery.

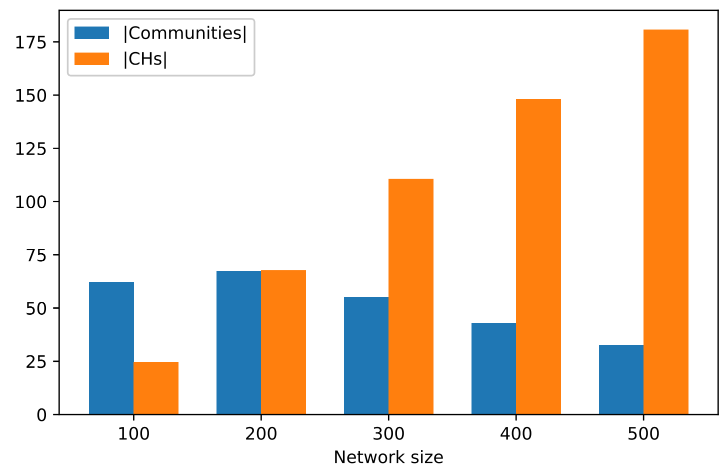

Figure 5.4: PL-CAUSE: Number of communities and number of CHs in similarity networks of different size, produced with similarity threshold θ=0.8

Impact of network size on the cache hit ratio In the following, it is investigated, separately, how the network size affects BS cache hit ratio and the UE cache hit ratio. Given a catalogue of items, the network size has a minor impact on the BS cache hit ratio, as shown in Figure 5.5 by the same colored bars. The intuition behind this is that the sets of users in the networks under examination are samples of the total users of the platform, based on whom the popularity of items is determined. Due to the relatively large sample sizes $\{100, 200, 300, 400, 500\}$ users out of 5000, (which is the size of the considered population), the information on items' popularity is reflected with precision in all the networks of users. This can be further justified by the results shown in Figures 5.6, 5.7, where the absolute frequency of the items' requests is presented for a catalogue of size $|I| = 50$ and networks with $|V| = 500$ and $|V| = 100$, respectively. The BS can store on average $\frac{c_{BS}}{M} = \frac{100}{4} = 25$ items of highest preference to users.

The 25 most popular items in the networks of different sizes illustrated in Figures 5.6, 5.7 coincide. For example, the most popular item in both networks is the item with index 39 which is requested by $\sim 30\%$ of the networks' users, i.e., 29 users in the network of |V|=100 and 152 users in the network of $|V| = 500$. The CH cache hit ratio increases proportionally to the network size. This is because bigger networks are denser, resulting in a more balanced partition, which in turn allows the selection of a great number of CHs, as shown in Figure 5.3. As expected, availability of more cached content in the network leads to a better CH

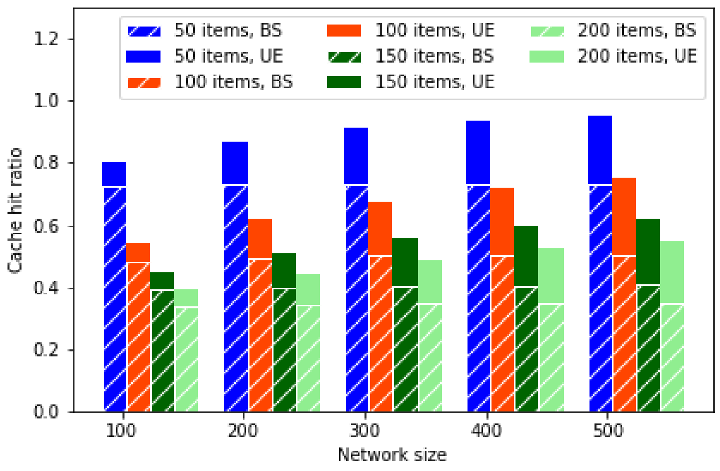

Figure 5.5: PL-CAUSE: BS and UE cache hit ratio for networks of different size, produced with similarity threshold θ=0.8 and lb = 0.5.

cache hit ratio.

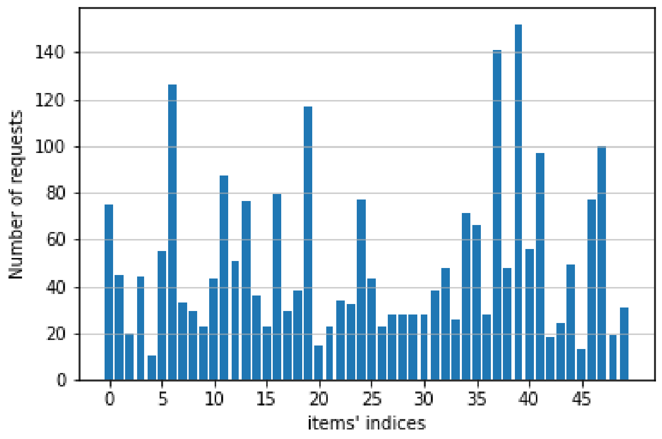

Figure 5.6: PL-CAUSE: Number of requests per item in a catalogue of size $|I|$ = 50 and a network of $|V|$ = 500.

5.2.4.3 Full utilization of UEs: PG-CAUSE, RG-CAUSE

In the following paragraphs the extent at which the collaboration among similar users is leveraged in cached content sharing is increased. PG-CAUSE, where all UEs participate in proactive caching, as well as RG-CAUSE, a reactive caching scheme with online content updates, which paves the way for peer-to-peer applications are examined. Such applications

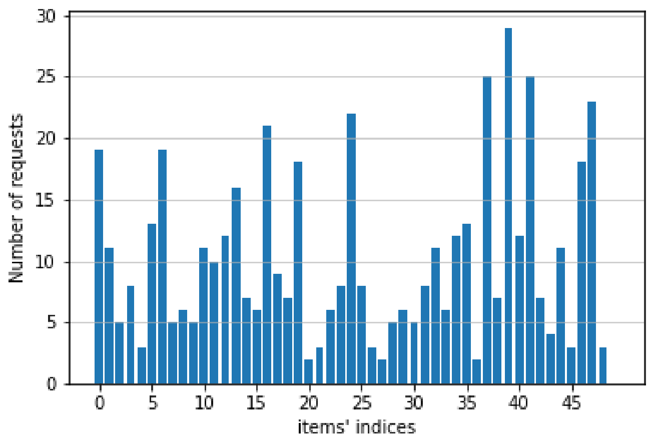

Figure 5.7: PL-CAUSE: Number of requests per item in a catalogue of size $|I| = 50$ and a network of $|V| = 100$.

would benefit both network users (by allowing them to share content in an ad-hoc manner) and network providers (by reducing the traffic at the backhaul links).

In Figure 5.8 the total cache hit ratio of each scheme is presented for the case of $|V| = 500$. From this figure and Figure 5.5, it is observed that global utilization of UEs results in better cache hit ratios than those observed in the limited utilization cases. This is expected since in the first case there is a lot more memory in aggregate available in the network. Comparing PG-CAUSE to RG-CAUSE, it can be seen that the performance is close in terms of the total achieved cache hit ratio, with the proactive scheme of PG-CAUSE being better in all setups. The decrease observed in both schemes as the number of available items increases is expected as the BS is able to cache a smaller percentage of the total items, while also the users' requests become more diverse.

Taking into account that the BS cache hit ratios are essentially the same for the two schemes under examination, the UE cache hit ratios achieved for each one of the two policies is the decisive factor that determines the overall total caching hit ratios. This is confirmed by the results displayed in Figure 5.9 where it is seen that the proactive scheme achieves in all cases higher cache hit ratios. This is due to the fact that in the proactive scheme, the users do not cache content selfishly but the preference of their neighbors to the items is taken into account.

Moreover, the proactive scheme is rather competent given the fact that there is no

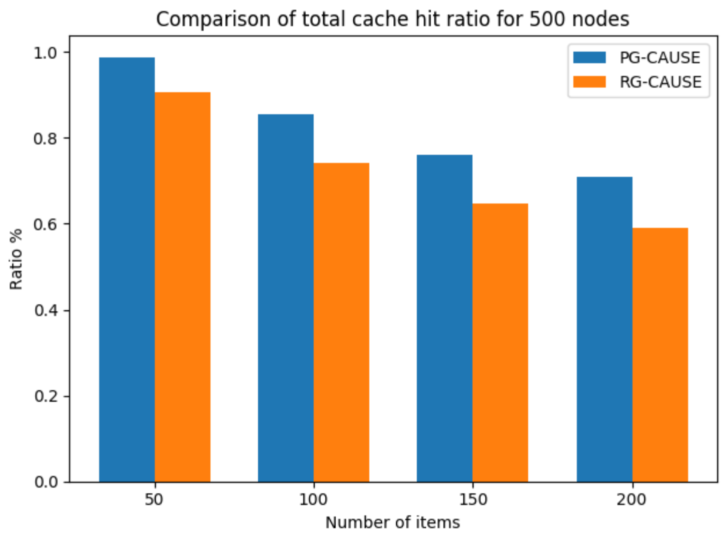

Figure 5.8: Total cache hit ratio comparison of PG-CAUSE and RG-CAUSE for various sizes of item catalogues - 500 nodes.

modification in the users' caches after the content decided by the platform is cached. The overall good performance is heavily based on the fact that the proactive scheme discussed earlier takes into account the preferences of all the neighbors of a user in the similarity graph, leading to the storage of content that is relevant to the user's neighborhood, meaning that it is highly possible to be requested by some of them. The reactive scheme is able to produce competent cache hit ratios, although lower than those of PG-CAUSE, enhanced by the replacement mechanism, which, via D2D communication, allows popular content to be spread over paths of similar users despite the cold start problem inherent with such approaches.

The process of deciding the appropriate content to store in each user's cache memory (i.e., solving the corresponding Knapsack problem) may impose a computational cost on the platform because it needs to monitor the network and quickly decide, based on the current population of the cell, which item(s) should be stored in which UEs. On the other hand, the reactive scheme is easy to implement, since the platform cares only for filling the BS's cache with items and then users make their requests, caching (or not) the requested items as described earlier, in hopes that their requests will be popular with similar users in their vicinity. Although the stored content is indeed requested by other neighboring users, as the bar plot indicates, it is important to take into consideration groups of similar users in order

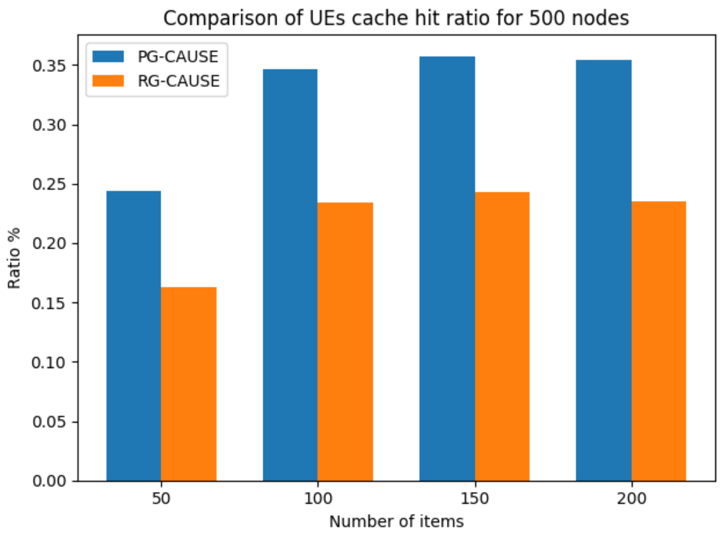

Figure 5.9: UE cache hit ratio comparison of PG-CAUSE and RG-CAUSE for various sizes of item catalogues - 500 nodes.

to increase the cache hits in such networks, possibly leading the way to the development of hybrid schemes.

Moreover, Figure 5.10 presents the total number of swaps (i.e., changes of cache memory contents) that are observed in the experiments.

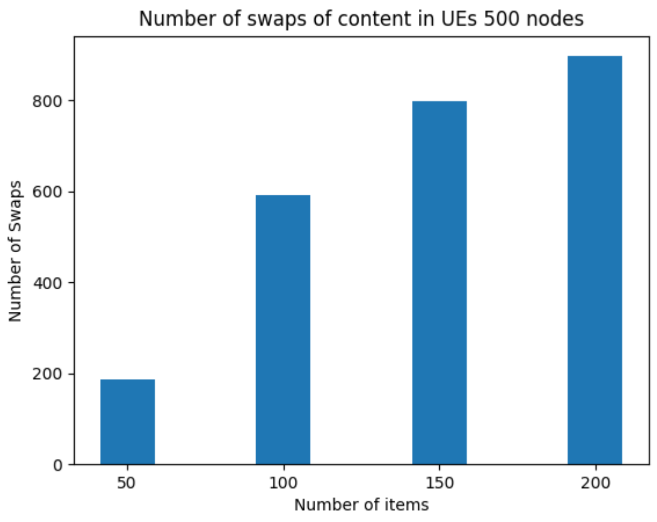

Figure 5.10: Number of swaps for 500 nodes in the reactive model RG-CAUSE.

The number of swaps increases as the catalogue size increases. This behavior is expected because as more items are available and the requests of the users become more diverse. The number of swaps needed in order for the reactive scheme to achieve the displayed hit ratios is quite large and imposes a computational cost in each UE in order to implement a proper swapping policy and also can, when taken into extremes, contribute to the faster wearing out of these memories and also the depletion of their batteries, a significant factor in mobile wireless communications.

5.3 Combining Caching and Recommendations at the Network Edge

It is a fact that in the last decade, mobile data traffic has greatly increased due to the faster connections and the demand for streaming content and the smarter user terminals [170]. As it was seen in the previous sections, proactive caching of highly requested content can aid in the reduction of the traffic load originating from the core network, thus alleviating backhaul congestion and improving user Quality of Experience (QoE) [171]. Providing the necessary incentives to users can make them willing to grant to the provider of the service part of their cache memory, making D2D communication a practically feasible solution for reducing traffic load in the network links [172].

Moreover, as it was discussed in Chapter 4, Recommender Systems play a crucial role in shaping user demands for content and are an irreplaceable part of modern platforms of streaming services [118]. When providing recommendations to users in real time, the delivery delay of the requested content as well as the relevance of the recommendations to the users' particular tastes are two major factors affecting the users' engagement to the platform.

The combination of caching and recommendations at the network edge has been studied in [147, 5] aiming to increase user engagement to the platforms, while minimizing the cost associated to the content delivery. Moving forward in this direction, this problem is investigated in a heterogeneous caching network with small cells and mobile users.

In this part of the Chapter, the problem of caching and recommendations at the network edge is treated by leveraging on the users' mobility patterns and their capability for D2D communication and data offloading, aiming to reduce the delivery delay experienced by the user when requesting content and, at the same time, provide the users with content relevant to their individual preferences. In the developed algorithm, a small portion of the users participate in content caching. Two different methods for selecting them are developed and their differences are highlighted. The problem of caching and recommendations is modeled as a QoE maximization one that is known to be NP-hard. In order to solve it, a heuristic method is developed. This approach is evaluated using synthetic datasets and it is compared to a state-of-the-art QoE maximization algorithm, proving its scalability as well as the proximity of the achieved QoE values.

5.3.1 Modeling Mobility and the Network Edge

The system model is considered to be a heterogeneous cellular network, consisting of a macro Base Station (BS), k Small Base Stations (SBS) $S = \{s_1, ..s_k\}$ scattered around the examined area, and n mobile users with smart devices (User Equipment, UE), $U = \{u_1, ..., u_n\}$ with transmission ranges $r_t(BS) > r_t(s) > r_t(u)$ with $r_t(s_i) = r_t(s)$ $\forall s_i \in S$ and $r_t(u_j) = r_t(u)$ $\forall u_j \in U$. The BS is able to communicate in an one-hop fashion with all the UEs, while a UE can communicate with an SBS, only if it is lies within its transmission range. If two users are within $r_t(u)$ distance, they can communicate with each other via D2D communication.

Available content. The available items for caching can be defined as a set $I = \{i_1, ..., i_m\}$ of m items with size z_i. The items' popularity can be estimated by a RecSys operating on the platform. The relevance of user u to an item i is given by $r : \{U \times I\} \rightarrow [0, 1]$.

Cache memories. Each SBS is equipped with a limited storage capacity. Content is cached in these devices in order to decongest the core network links. UEs also have some storage capacity $cap : \{S, U\} \rightarrow \mathbb{R}$. Each user $u \in U$ has less available memory for caching than that an SBS. Given the average size of an item $\bar{z}$ and the cache capacity of every user u, $cap(u)$, the average number of items that may be stored in every UE is defined as $k(u) = \lfloor \frac{cap(u)}{\bar{z}} \rfloor$. Content is stored at a predefined, by the operator, number of UEs. The operator needs to provide the users that will cache content with certain motives, such as a reduced charging policy or increased bandwidth. The selected users are referred to as ClusterHeads (CHs). The set of CHs is $C = \{c_1, ..., c_g\}$.

Recommendation lists. Streaming services, such as Netflix, employ RecSys to aid their users in deciding the desired content (e.g., a video file). Each user $u \in U$, who connects to the platform, is provided with a list of recommended content $\Gamma(u)$, where $\Gamma : U \rightarrow \mathbf{I}_\Gamma$

is a set-valued function and $\mathbf{I}_\Gamma$ is the family of the subsets of I with size equal to l. The recommendation list containing the most relevant content for each user is denoted as $\Gamma_b(u)$.

Contact rate between users. The meeting events between (c, u), $\forall u \in U$, $\forall c \in C$ are given by a Poisson process with rate $\lambda_s(c, u)$. The meeting rates can be drawn from an arbitrary probability distribution. It is assumed, for the sake of fairness in content sharing, that in a meeting event between u and c, u will receive only one content from c, even if the meeting duration is longer. Also, the expected inter-contact time coincides with the expected delay experienced by u in order to get content from c. The inter-contact distribution is exponential with rate equal to $\lambda_s(c, u)$. Therefore, the expected inter-contact time for (c, u) will be $\frac{1}{\lambda_s(c,u)}$ when $\lambda_s(c, u) \neq 0$. A realistic assumption regarding the meetings between users is that not all contacts between (c, u) lead to a successful content exchange (e.g., battery depletion of a device, etc.). Thus, the expected delay must be greater than the expected inter-contact time. Given a probability of successful content exchange, p_s, the number of successful meetings follows a Poisson distribution with rate $\lambda_{cu} = \lambda_s(c, u) \cdot p_s$. Therefore, the expected waiting time for successful content exchange between (c, u), when $\lambda_{cu} = \lambda_s(c, u) \cdot p_s \neq 0$ will be $\frac{1}{\lambda_{cu}} = \frac{1}{\lambda_s(c,u) \cdot p_s}$.

Expected Delay. The expected delay is defined as $f : \{U \times BS, U \times S, U \times C\} \rightarrow \mathbb{R}$. If a pair of users never meet, meaning than $\lambda_{uv} = 0$, then the expected delay is set to infinity. Otherwise, the expected delay can be defined as follows.

$$f(u, v) = \begin{cases} \frac{1}{\lambda_{uv}}, & \text{if } u \neq v, \\ 0, & \text{if } u = v. \end{cases} \tag{5.10}$$

Expected number of efficient meetings between users. Considering the expected time between the successful content exchange for each pair of users as the expected delay between users u and v, it is possible to estimate the number of times two users will meet in the examined period of duration T. A matrix $\mathbf{M} = (m_{uv}) \in \mathbb{N}^{|U| \times |U|}$ can be computed. This matrix is non-symmetrical and it is defined as follows:

$$m_{uv} = \begin{cases} \min\{k(u), \lfloor \lambda_{uv} T \rfloor, l\}, & \text{if } f(u,v) < T_s, \\ \min\{k(u), l\}, & \text{if } u = v, \\ 0, & \text{otherwise.} \end{cases} \tag{5.11}$$

Matrix **M** contains the expected number of efficient content exchanges between two nodes (during a meeting only a single contact exchange is allowed) that will result in successful delivery of a content with expected delay less than T_s.

Opportunistic offloading of user requests. Content is delivered to the users in U in three possible manners. Either from the CHs via D2D communication, the SBSs, or the core network through the BS. User $u \in U$ can wait for an amount of time, T_s, until she moves within range of a CH or a SBS in order to retrieve content $i \in \Gamma(u)$ from the corresponding caches. The requested content is retrieved from the cache resulting in the lowest delivery delay. If the maximum time, T_s is reached, the content is delivered directly through the BS. In this case, the delay experienced for the delivery of the item is set to be equal to T_s.

5.3.2 Quantifying User Experience

Quality of Recommendations (QoR). Following the approach mentioned in [5], the QoR of user $u \in U$ is defined to be a function of her relevance to the items of her recommendation list $\Gamma(u)$:

$$\phi(u, \Gamma(u), r) = \sum_{i \in \Gamma(u)} r(u,i). \tag{5.12}$$

Quality of Service (QoS). The QoS of user u depends on the expected tolerable delay experienced by her in order to access the contents of her $\Gamma(u)$ list. It is the deviation of the total expected waiting time of u from the total maximum tolerable waiting time $|\Gamma(u)|T_s$. A higher deviation is an indication of a better QoS. For user u, the set of caches $H_u = \{h \in \{C \cup S\} : m_{hu} > 0\}$ to which u is connected for efficient content exchange, is defined. It is also possible that content i is not cached anywhere in the cache-enabled device in H_u, thus, it will be delivered to u via the Base Station in T_s time. For this, the set $H_u^\dagger = \{H_u \cup BS\}$ is defined. The set $H_u^\dagger$ is sorted in increasing order of tolerable delay. In every case, the last element in $H_u^\dagger$ is the BS. The QoS of user u is defined as:

$$\psi(u, f, \Gamma(u), H_u, \Omega_u) = \sum_{j=1}^{|H_u^\dagger|} (T_s - f((j)_u, u) \sum_{i \in \Gamma(u)} \left[\prod_{\nu=1}^{j-1} (1 - \omega_{(\nu)_u i}) \right] \cdot \omega_{(j)_u i}, \tag{5.13}$$

where $f((j)_u, u)$ is the tolerable delay experienced by u for content delivered by the j-th element (device) of $H_u^\dagger$. The binary variable $\omega_{(j)_u i}$ indicates whether item i can be delivered via the device in the j-th position of $H_u^\dagger$. The product $\left[\prod_{\nu=1}^{j-1}(1 - \omega_{(\nu)_u i})\right] \cdot \omega_{(j)_u i}$ ensures that the cache of lowest delay containing item i in $H_u^\dagger$, will be the one selected for the delivery of item i to u. It is evident that in the case that an item is delivered through the core network (via the BS) the QoS will be equal to 0.

Quality of Experience (QoE). The QoE of user u is defined as a convex combination of the QoR and the QoS.

$$Q(u, \Gamma(u), r, f, H_u, \Omega_u) = a\psi(u, f, \Gamma(u), H_u, \Omega_u) + (1 - a)\phi(u, \Gamma(u), r), \tag{5.14}$$

where $a \in [0, 1]$ is a parameter indicating the trade-off of the QoR and the QoS. In case of $a = 0$, the users do not mind waiting even for time equal to T_s in order to consume their favorite content. On the contrary, if $a = 1$, the users do not tolerate any delays and are willing to consume less relevant content in exchange for faster delivery.

5.3.2.1 Mobility-aware Clusterhead Selection

The criterion for the CH selection is the number of different users that each user encounters in time period $[t, t + T]$. For this purpose, the matrix $\mathbf{B} = (b_{uv}) \in [0, 1]^{n \times n}$, with $u, v \in U$ is employed, with its elements being

$$b_{uv} = \begin{cases} 1, & \text{if } m_{uv} > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{5.15}$$

This means that $b_{uv} = 1$ if u meets v with expected delay less than T_s. Having computed matrix $\mathbf{B}$, the vector $\mathbf{b}$ with $\mathbf{b}[u] = \sum_{v \in U} b_{uv}$ is computed. In this case, the CHs can be selected as follows:

$$\begin{aligned} A_0 = U \text{ and } E_1 = \{u \in A_0 : \underset{u}{\operatorname{argmax}}\, \mathbf{b}\} \\ A_i = A_{i-1} \setminus E_i \text{ and } E_{i+1} = E_i \cup \{u \in A_i : \underset{u}{\operatorname{argmax}}\, \mathbf{b}\} \end{aligned} \tag{5.16}$$

The set of CHs is then $C := E_g$ and it is formed by the g users who will encounter the most users in the examined time period. This method leverages on the fact that some of the nodes are expected to meet a lot of users during the examined time period. These nodes are selected as CHs and if the users follow recommendations for the content cached in their memories, this content will be delivered with tolerable delay.

5.3.3 The QoE problem formulation

For each user, given the set of caches (i.e., CHs and SBSs) to which she is connected for efficient delivery of recommended content, it is desired to maximize the total QoE for all the users in network. For this, the recommendation lists $\Gamma(u)$ $u \in U$ need to be produced, i.e., the user-item $|U| \times |I|$ binary matrix $\mathbf{\Gamma} = (\gamma_{ui})$ where $\gamma_{ui} = 1$ if $i \in \Gamma(u)$ and the $(|C| + |S| + 1) \times |I|$ binary matrix $\mathbf{\Omega} = (\omega_{ji})$ represents the caching of the items. The last row of $\mathbf{\Omega}$ concerns the items that can be transmitted from the BS, meaning that the elements of this row are all equal to 1. The problem is formulated as follows:

$$\underset{\mathbf{\Gamma},\mathbf{\Omega}}{\operatorname{argmax}} \sum_{u \in U} Q(u, \Gamma(u), r, f, H_u, \Omega_u), \tag{5.17}$$

subject to:

$$\sum_{i=1}^{|I|} \omega_{ji} \leq k(j) \quad \forall j = 1, ..., |C| + |S| \tag{5.18}$$

$$\sum_{i=1}^{|I|} \omega_{(|C|+|S|+1)i} = |I| \tag{5.19}$$

$$\sum_{i=1}^{|I|} \gamma_{ui} = l \quad \forall u \in U \tag{5.20}$$

Constraint (5.18) represents the capacity limits of the network caches. Constraint (5.19) makes sure that, in the worst case, all the items can be delivered through the BS (i.e., via the core network). Constraint (5.20) ensures that each user will receive a recommendation list of size equal to l.

This problem is equivalent to the QoE problem with equal-sized content constraints presented in [5], which, in that article is proven to be NP-hard. The authors develop a greedy algorithm with approximation guarantees for its solution. The proposed algorithm provides a $\frac{1}{2}$-approximation for the QoE problem but it is based on an exhaustive evaluation of all the possible solutions, marking it as a rather time consuming approach. In order to avoid this, in this thesis, the problem is divided into a problem of recommendations and one of content placement in networks of efficient content exchanges. A heuristic algorithm is developed that investigates the trade-off of the quality of the provided solution and the execution time.

5.3.4 Tuning QoR for Efficient Recommendations

In the proposed approach the focus is on generating recommendation lists that display QoR values above an operator-specified bound while achieving respectable QoS values. The problem of deciding which content to cache to the CHs and the contents of $\Gamma(u)$ for each user is solved sequentially as a content placement and assignment problem. At first, the weighted bipartite graph, $G_b = (H^\dagger, U, E_b)$, $H^\dagger = \{C \cup S \cup BS\}$ is constructed. The edges denote the realization of efficient meetings between the set of users U, the cache-enabled devices (i.e, CHs and SBSs), and the BS. The weight of these edges is the delay per efficient content exchange. In the case of an edge (c, u), $c \in C, u \in U$ its weight is equal to the expected delay $f(c, u)$, since only one item can be delivered when these nodes meet. In the case of an edge (s, u) $s \in S, u \in U$ the edge weight is equal to $\frac{f(s,u)}{n_d}$, as n_d items can be delivered in a meeting between a user and a SBS. An overview of the G_b graph can be seen in Figure 5.11.

In order to distinguish for each user the set of devices that she is expected to meet with lower delay, the bipartite graph $G'_b = (H^\dagger, U, E'_b)$ with $E'_b \subseteq E_b$ is computed. The set E'_b is constructed by adding for each node $u \in U$ the $\min\{l, |N_u|\}$ incident edges of minimum weight from E_b, where N_u is the set of neighbors of u in G_b.

Content Placement. By examining each cached-enabled device (i.e., CH or SBS) h in G'_b, the l most highly ranked items of each node lying in the neighborhood N_h denote

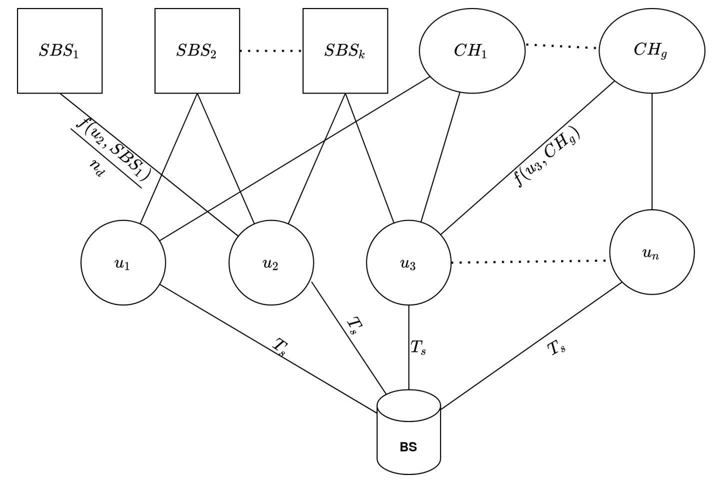

Figure 5.11: Bipartite network denoting the delay per efficient content exchange between the cache-enabled evices, the BS and the users of the system.

possible assignments of items to the device. Consider, without loss of generality, h and node u in N_h. The function $q(h,i), \forall h \in H^{\dagger}, \forall i \in \bigcup_{u \in V} \Gamma_b(u)$ is defined as:

$$q(h,i) = \begin{cases} \frac{\sum_{u \in N_h} r_{ui}}{max_{v \in N_h} f(h,u)}, \text{if } f(h,u) < T_s, \forall u \in N_h \\ 0, \text{otherwise.} \end{cases} \tag{5.21}$$

The values of the function q will be used as the items' utilities in knapsack problems. High values of q for a pair (h,i) indicate that this item is highly preferred by the nodes residing in N_h and also that it will be delivered to them in a relatively short amount of time. The summation over all the nodes in N_h is a measure of how much these nodes would be satisfied if recommended of item i. By dividing it with the maximum delay noticed in N_h, it is assumed that the requests made by nodes in N_h for the item in question and for which it holds that $f(h,u) \leq max_{u \in N_h}\{f(h,u)\}$ will be already satisfied by the time the device meets the node v for which it holds $f(h,v) = max_{u \in N_h}\{f(h,u)\}$. Finally, given each device's capacity, the items' size and the items' utilities, a knapsack problem can be solved for every $h \in H = H^{\dagger} \setminus \{BS\}$.

In order to place content in all caches we solve $|H|$ knapsack problems (one for every device). However, the caching of the same content across different devices if they are to meet the same users during the examined time period, is not desired. To address this, the relevance scores of nodes for items that are already placed in the cache of a device they are expected to meet are set to zero. In this way, they do not contribute in the process of content selection for the remaining devices.

In this approach, the order in which the devices are examined impacts the quality of the solution obtained. By considering only the l minimum weighted edges incident to each user, the search space is significantly reduced and only edges that reflect relatively small delays are maintained. In order to decide the order in which the memories of the cache-enabled devices are filled, they are examined in ascending order of the average delay experienced $\bar{f}(N_h) = \frac{\sum_{u \in N_h} f(h,u)}{|N_h|}$. Thus, devices that have small average delays for efficient content exchanges fill their caches first. Their neighboring nodes in G'_b are more likely to be served sooner if they request content from them than if the content was placed in another device. A high level pseudocode of the proposed TR content allocation approach can be seen in Algorithm 8.

Algorithm 8: Algorithm TR Content Allocation.

```
Input  : $G'_b$, $l$, the list $z$ of weights of the items in $I$, the set $H$ of caches, the
         expected delays $f$, the cache capacities $cap$, relevance scores $r$
Output: Cached contents for each device
1  $\bar{f} := list()$
2  for $h \in H$: do
3      $N_h = neighbors(G'_b, h)$
4      $\bar{f}[h] = \frac{\sum_{u \in N_h} f(h,u)}{|N_h|}$
5  S := argsort$\{\bar{f}\}$
6  Cached := $list()$
7  for $h \in S$ do
8      rws := $[q(h,i),\ \text{for } i \in I]$
9      Cached[h] := $knapsack(rws, z, cap(h))$
10     for $u \in N_h$ do
11         for $i \in Cached[h]$ do  $r(u,i) = 0$ ;
```

Recommendations. The recommendation lists $\Gamma(u)\ \forall u \in U$, are compiled by tuning a parameter denoted as δ. This parameter sets a maximum deviation in the relevance score of an item that is to be recommended to a user. The items that can be retrieved from the devices in the neighborhood N_u in G'_b (with the exception of the BS) are sorted in descending order of relevance score for this user. An item i will be included to the recommendation list if it holds that $\frac{|(r(u,i)-r^{top}_{min}|}{r^{top}_{min}} \leq \delta$, with r^{top}_{min} being the relevance score for the l-th item in $\Gamma_b(u)$ in terms of relevance. If the list does not yet have l elements, the search is continued to the rest of u's neighbors in the original graph G_b, excluding the BS. The items stored in them are candidates for recommendation and are included to $\Gamma(u)$ following the procedure described earlier. If still l recommendations have not been made to u, the remaining positions of the recommendation list are filled with items that are included in $\Gamma_b(u)$ and delivered via the BS.

Tuning the δ parameter allows setting the desirable lower bound to the obtained QoR. For example, if $\delta = 0$, then $\Gamma(u) = \Gamma_b(u)$. This will result in the maximum possible QoR but in the expense of the achieved QoS (i.e., longer delays experienced for more desirable content). This is equivalent to solving the QoE problem formulated in Eq. (5.17) for $a = 0$. On the other hand, by setting $\delta = 1$, then, the first l items of maximum relevance that can be found in the neighbors of u in G'_b will be included in the recommendation list. This will result in the maximum QoS obtained by this method but possibly at the expense of obtaining the lowest QoR (i.e., shorter delays but for less preferred content).

5.3.5 Evaluation of the TR algorithm

In this Section, the differences between the proposed CH selection approaches are discussed and then, the proposed TR algorithm for caching and recommendations is evaluated in terms of the achieved QoR, QoS and QoE scores. The obtained solutions are compared against the ones produced by the method proposed in [5], which we will refer to as JCR. For the evaluation purposes the employed settings contain variable number of users $V = \{100, 200, 300, 400, 500\}$ and content catalogues of $\{300, 1000\}$ items, all with size equal to 1. The user-item relevance scores are drawn uniformly at random from $[0, 1]$. The cache sizes of the nodes are measured in terms of the items they can store which can be $\{2, 3, 4\}$. We consider 5 SBSs equipped with cache memories of size $[5, 10] \subset \mathbf{N}$ items. Regarding the number of items that a user can get from an SBS, the n_d parameter is set equal to 3. Every user-user pair meets with rate chosen uniformly at random in the range $[0.01, 0.4]$, while every user and SBS pair meets with rate chosen uniformly at random in the range $[0.001, 0.15]$. The number of CHs g is equal to 5% of the users' number. The size of the user recommendation list is set to $l = 5$. The maximum tolerable delay is set to $T_s = 10$. The examined time period is set equal to $l \cdot Ts + 1 = 51$, indicating just over the maximum amount of time for a user to retrieve all of her recommendations from the BS. The parameter δ is set to 1, unless otherwise stated.

Regarding the QoR, $n\text{-}QoR = \frac{1}{|U|} \sum_{u \in U} \frac{QoR(u)}{QoR_{max}(u)}$, is a measure that reflects how close to the optimal QoR are the recommendation lists of the whole network. For each node, the achieved QoR is calculated based on Eq. (5.12) and divided by QoR_{max}, which is the QoR score achieved if $\Gamma(u) = \Gamma_b(u)$. These scores are averaged across the whole network. Regarding $n\text{-}QoS$, it is given by the expression: $n\text{-}QoS = \frac{1}{|U|} \sum_{u \in U} \frac{QoS(u)}{l \cdot T_s}$, where for every node u the achieved QoS score as computed by Eq. (5.13) is divided by the maximum total delay that would be experienced if all of its recommendations were retrieved from the BS. The $n\text{-}QoS$ is the result of the averaging process across all the users. Finally, for evaluating the QoE, a new measure $n\text{-}QoE$ is defined, which is equal to, $n\text{-}QoE = n\text{-}QoR + n\text{-}QoS$.

5.3.5.1 Impact of δ parameter in TR

As it can be seen in Figure 5.12, the δ parameter sets a lower bound to the achieved $n\text{-}QoR$ ratio but also impacts the achieved $n\text{-}QoS$. The maximum $n\text{-}QoR$ is obtained when $\delta = 0$ meaning that only the l items in $\Gamma_b(u)$ will be recommended for every user u. Of course, this means that a lot of items will have to be retrieved from the BS because it is impossible to have them all cached in the limited space of the CHs or even the SBSs. On the other hand, when $\delta = 1$, the maximum $n\text{-}QoS$ possible for the TR approach is achieved. In this case, the most relevant of the items cached to the devices that are neighbors to each user u in G'_b are recommended to the user. This explains the lower $n\text{-}QoR$ ratio achieved since it is possible that more relevant items are cached in other devices that the user is expected to meet in tolerable delay. So, as the value of δ decreases, the boundaries for the inclusion of items in the recommendation lists become more strict. This results in higher $n\text{-}QoR$ but lower $n\text{-}QoS$ as these items will have to be retrieved from devices that display higher delays.

5.3.5.2 Utilization of the caches

The location of the recommended items for the TR approach for both CH selection methods are examined in this paragraph. The fraction of recommended content retrieved by different locations is depicted in Figure 5.13. As it can be seen in the plot, the TR approach relies heavily on the CHs. This is due to the fact that in order to generate recommendation lists, the contents that each user can retrieve are sorted according to their relevance score regardless of where they were cached. Thus, giving CHs the edge, with the participation

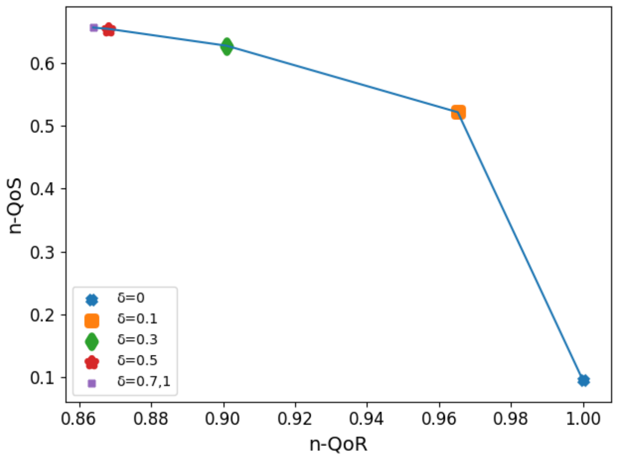

Figure 5.12: Impact of δ parameter in *n-QoR* and *n-QoS* metrics.

percentage increasing as the number of users increases.

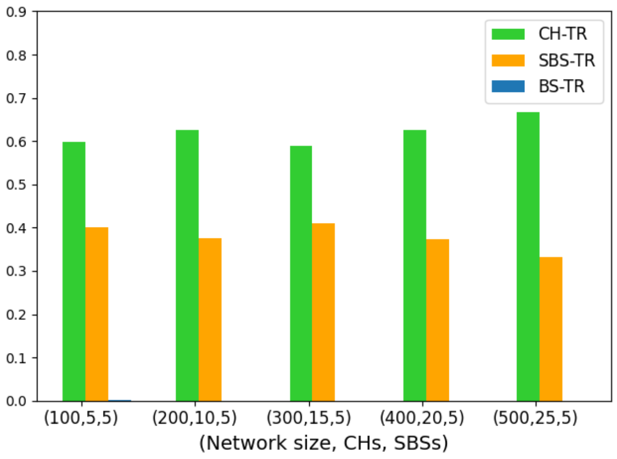

Figure 5.13: Location for the recommended items (CHs selected with CH-served).

5.3.5.3 Achieved *n-QoR*, *n-QoS* and *n-QoE* scores

In Figure 5.14 the TR and JCR approaches are compared in terms of their achieved *n-QoR*, *n-QoS* and *n-QoE* scores for the case of CH-served selected CHs and for a content catalogue of 300 items. Results for the case of CH-cover selected CHs are similar, so they are omitted. The TR method is able to achieve high values of *n-QoR*, even surpassing slightly those of JCR, even in the case of $\delta = 1$, by recommending items of high relevance stored in caches of greater delivery delay than those selected by JCR, which causes lower *n-QoS*. The results on the *n-QoE* for these methods are of similar quality. Regarding the JCR method, it achieves *n-QoR* scores higher but comparable to those of TR.

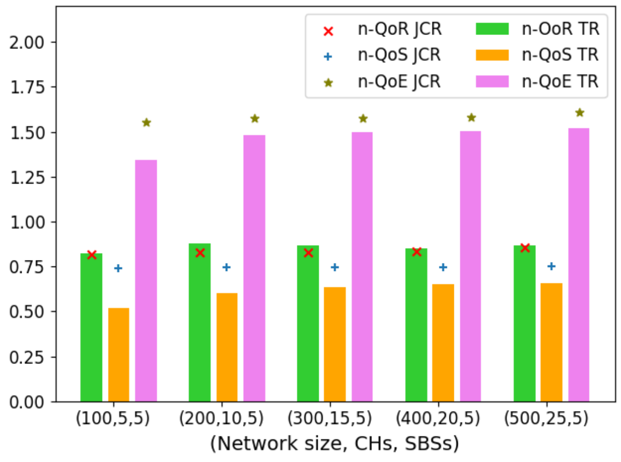

Figure 5.14: *n-QoR*, *n-QoS* and *n-QoE* scores for the recommended items.

The JCR's execution time though is quite high as depicted in Table 5.1 as opposed to TR, which is just a few seconds. Thus, the small loss in the *n-QoE* of TR is justifiable.

Table 5.1: Execution time (sec) of examined approaches

Network Setup	**TR**	**JCR**
(100,5,5)	0.07	8278.30
(200,10,5)	0.11	39684.54
(300,15,5)	0.13	162617.48
(400,20,5)	0.15	346948.55
(500,25,5)	0.20	757482.49

Chapter 6

Conclusion

6.1 Outcomes

In this study, multiple aspects for designing a holistic framework in order to further merge the "worlds" of OSNs and interconnected devices have been studied. In order to do so, the interdependency among various different types of actors is taken into account. Key problems have been identified, novel algorithms have been developed and their performance has been evaluated.

The first two problems encountered were those of data clustering and community detection. In **Chapter 3** a community detection algorithm, HGN, coupled with a graph database, has been proposed in order to deal with the above problems. This algorithm is able to detect communities in both social networks as well as proximity graphs constructed by joining data measurements. It relies on computations on the distances of the network's nodes performed faster in the hyperbolic space than by applying classic social network analysis techniques. The strong features of HGN were highlighted by its comparison with other recent, as well as more traditional and established algorithms. From the evaluation, it was shown that the algorithm was accurate in detecting clusters (or communities) in benchmark datasets with known ground truth clusters (or communities). Also, when applied to social networks, it lead to the formation of communities resulting in high modularity scores. Moreover, HGN was also applied on real sensor data obtained by a real Smart City. From the performed experiments in these real datasets two major conclusions could be drawn. The first one is that groups of sensors that remained clustered together across different time snapshots were detected. The second one is that mapping the problem of data clustering of the measurements to a community detection one leads to the formation of communities of high modularity values. Knowledge of such groups is crucial when designing fault tolerance mechanisms and also when taking into consideration energy consumption (i.e., sensor polling rates, existence of redundant sensors, etc).

Then, in **Chapter 4** the focus was turned in the information diffusion process that occurs in an OSN. The need to identify the influential nodes for every possible suggestion is essential for the efficient recommendation of pieces of information like applications, items, etc. Each available item for recommendation, is considered to display a fixed relevance score, quantifying how much relevant it is to each user of the system. Moreover, each user has certain information capacity thresholds that, if violated, may have a negative impact in the user's experience by overloading her with information. In order to efficiently allocate items to users, the problem was modeled as a relevance maximization one that was proven to be computationally difficult as it is an NP-hard problem with added constraints. The problem was divided into two subproblems each one addressing a capacity constraint that need to be solved in sequence. Heuristics, based on the information flow in the network, were employed for the solutions of each subproblem. The combination of solutions to these sub-problems provides a solution that achieves high relevance scores while managing to respect the user's constraints. Thus, it is suggested that taking into consideration the user boundaries is not contradictory to achieving recommendations of high relevance score that increase the user engagement to the platform and also lead to the recommendation of the vast majority of the available items to appropriate subsets of users.

Finally, in **Chapter 5**, the problem of increasing the users' QoE through combining recommendations and caching at the network edge is examined. In this thesis, user's equipment is employed alongside special network infrastructure such as (Small) Base Stations for the temporary storage and offloading of contents. In the first approach, the physical location of the users in a telecommunication's cell is taken into account and considered static as well as the similarity among them. The relevance of each item to each user and the probability of request is considered known from the operation of a recommender system in the platform. The problem was modeled as a cache hit maximization one. Multiple approaches for caching items are evaluated including a community based one where the objective is the coverage of as many nodes as possible from a subset of special nodes (i.e., clusterheads) acting as caches. Also, approaches where the total of the UEs act as caches are examined. Moreover, examining whether it is preferable to proactively store content in each UE as opposed to reactively store requested content, relevant algorithms were developed and compared. The benefits of cooperatively caching by taking into account the preferences of similar users in the user's vicinity contrary to selfishly caching requested content are highlighted.

Aiming at studying the interplay of caching at the network edge and recommendations and marking a step forward from other static approaches, a new algorithm is developed. This algorithm assigns cached content at a carefully selected subset of UEs by leveraging on users' mobility and provides each user with recommendations that aim at increasing their QoE. The user's QoE is expressed as a function of the achieved QoR and QoS. The problem of obtaining the solution that maximizes the QoE is NP-hard, so the proposed solution is a heuristic algorithm. Results obtained from simulations, showcase the comparable QoE values obtained in a timely manner with TR when compared to the results of an approximative, but exhaustive search-based, algorithm.

6.2 Recommendations for Further Research

Concluding this thesis, this final section sheds light on some of the possible future research directions that can be followed based on the outcomes of the presented work and the challenges faced during the research process. While the thesis treats problems that concern some of the most dynamic fields of current complex network analysis and highlights the need to examine various entities (i.e., connected devices, OSNs) as interdependent systems, there still exists much room for further development.

Regarding the topic of community detection aided by hyperbolic network embedding, the inclusion of more dynamic aspects towards developing an online algorithm, can aid in detecting communities (or clusters) efficiently in highly volatile and changing interconnected environments, without having to study snapshots of the network separately. An online dynamic HGN-like algorithm could maintain the strong features of HGN while reducing the total execution time when massive time dependent datasets need to be analyzed and clustered in real time. This direction of research, deems absolutely necessary the development of a modern distance preserving hyperbolic network embedding algorithm that will be able to re-calibrate distances in an online fashion (i.e., as new nodes join the system, new edges are created or older ones deleted, etc.), something lacking from the current bibliography.

Concerning the aspect of information diffusion. An interesting direction for further research, would be to involve aspects of community detection and more complex constraints. More specifically, clusters of similar items could be identified and the capacity of a user to similar items could be introduced. Investigating different thresholds of similarity between items and discovering groups of relevant items can aid in the scalability of the proposed framework by examining clusters rather than separate items. Also, it could pave the way for more diverse recommendations, further increasing the users' satisfaction. Moreover, measuring the impact of recommendations on users' interactions, namely, the effect of recommendations to the users' preferences and tastes in the social network is of high research and practical importance. In that way, trends can be identified and more sensitive IDARS can be implemented to easily adapt to changes of the demanded content. In addition, decentralized approaches on the recommendations allocation could lessen the computational burden of the content provider. The users could broadcast their preferences in their neighborhood and the UEs by executing proper distributed algorithms could "decide" on their own the proper allocation of recommendations that would not lead to information overload and then present it to the user-owner of the UE.

In terms of content caching, although both proactive and reactive approaches have been studied and compared, a way of expanding the work presented here is to merge these two approaches towards designing hybrid caching schemes, in order to further increase the achieved cache hit ratio and further minimize the times that information has to travel from the core to the edge. Moreover, the development of distributed approaches, where the UEs locally exchange short messages in order to elect CHs, will alleviate the need for a central authority to supervise the network and keep track of users' mobility patterns. In order for UEs to exchange content unobtrusively in large interconnected and interdependent environments where a multitude of devices transmits data at the same time over limited spectrum bands, there have been developed a multitude of solutions, such as cognitive radio approaches that can be studied and adapted to the aforementioned scenario. Towards further reducing the delay experienced by the users in order to receive content and also increase the diversity of content cached in the devices, schemes that employ the total of the available UEs will mark a step forward in increasing the obtained users' QoE. Of course, in order for a solution like this to be implemented, it is up to the content providers to come up with proper incentives in order to convince users to dedicate part of their devices' memories for the needs of offloading.

Extensive Summary in Greek

Εκτεταμένη Περίληψη στα Ελληνικά

Η παρούσα διατριβή εστιάζει στην ανάπτυξη καινοτόμων τεχνικών με σκοπό την ανακάλυψη των σχέσεων και των κρυφών συσχετίσεων μεταξύ των οντοτήτων σύνθετων συστημάτων. Τα συστήματα αυτά αποτελούνται από διάφορους τύπους συσκευών αλλά και χρήστες. Ακόμα η διατριβή εστιάζει και στην ανάθεση πόρων στις οντότητες ενός σύνθετου συστήματος. Για να επιτευχθούν αυτοί οι στόχοι, οι προτεινόμενες μέθοδοι λαμβάνουν υπόψιν την αλληλεξάρτηση και τις ποικίλες σχέσεις μεταξύ των πολλών διαφορετικών οντοτήτων. Αυτές οι μέθοδοι βασίζονται σε τεχνικές και εργαλεία από τους τομείς της θεωρίας γραφημάτων και της ανάλυσης κοινωνικών δικτύων. Συστήματα σαν αυτά που μελετώνται σε αυτήν την διατριβή, παρατηρούνται σε σύγχρονα διασυνδεδεμένα περιβάλλοντα όπως αυτά που αποτελούν οι Έξυπνες Πόλεις και αναμένεται να γίνουν ακόμα περισσότερα στο μέλλον. Αυτά τα συστήματα συνδυάζουν την λειτουργία μεγάλων υποδομών με τις ενέργειες και τις απαιτήσεις των ανθρώπων που αποκτούν πρόσβαση σε αυτές. Για την ανεμπόδιστη λειτουργία τέτοιων τοπολογιών, οι διαχειριστές του δικτύου πρέπει να είναι σε θέση να επιθεωρούν τα δεδομένα που παράγονται, να εντοπίζουν πιθανώς περιττό υλικό και να ανακαλύπτουν παρόμοιες περιοχές. Ακόμα, οι άνθρωποι που χρησιμοποιούν τέτοια περιβάλλοντα χρειάζεται να έχουν γρήγορη πρόσβαση σε δεδομένα αλλά και να έχουν τη δυνατότητα να πληροφορούνται σχετικά με τις εφαρμογές που αναμένεται να κρατήσουν την ποιότητα της εμπειρίας που απολαμβάνουν σε υψηλά επίπεδα. Αυτές οι οντότητες (άνθρωποι, συσκευές, μετρήσεις) είναι τα στοιχεία που αποτελούν τα σύνθετα συστήματα και σχετίζονται μεταξύ τους με πολλούς τρόπους, δημιουργώντας πολυ-επίπεδα σύνθετα δίκτυα τα οποία χρειάζονται τα κατάλληλα εργαλεία για την ανάλυσή τους.

Με σκοπό τη δημιουργία ενός πλαισίου μεθόδων που θα ικανοποιεί τους παραπάνω στόχους, αυτή η διατριβή εστιάζει στα προβλήματα της ανίχνευσης κοινοτήτων και της ανάθεσης πόρων σε αλληλοεξαρτώμενα και διασυνδεδεμένα περιβάλλοντα. Η ανακάλυψη σημαντικών προβλημάτων στις περιοχές αυτές και η ανάπτυξη κατάλληλων λύσεων μπορεί να βοηθήσει στην ανακάλυψη ομάδων από παραπλήσιες συσκευές οι οποίες λειτουργούν σε αυτά τα περιβάλλοντα, ομάδων από παρόμοιους χρήστες καθώς και να ξεχωρίσει τους πιο επιδραστικούς από αυτούς από τη σκοπιά της διάχυσης πληροφορίας.

Κεφάλαιο 2

Αναλυτικότερα, στο Κεφάλαιο 2 εξηγούνται οι λόγοι για τους οποίους η Θεωρία γραφημάτων και πιο συγκεκριμένα, ο τομέας της ανάλυσης σύνθετων/κοινωνικών δικτύων αποτελούν το κύριο μαθηματικό υπόβαθρο των προσεγγίσεων που αναπτύσσονται στα πλαίσια της διατριβής. Τα μεγάλα και σύνθετα διασυνδεδεμένα περιβάλλοντα, όπως οι λεγόμενες Έξυπνες Πόλεις (Smart Cities), αποτελούνται από πολλά στοιχεία (π.χ., ανθρώπους, έξυπνες συσκευές, αισθητήρες κ.λπ.) που αλληλεπιδρούν ανταλλάσσοντας πληροφορίες και συνάπτοντας διάφορους τύπους σχέσεων. Η θεωρία γραφημάτων είναι ο κλάδος των μαθηματικών ο οποίος ασχολείται με την ανάλυση τέτοιου είδους συστημάτων τα οποία και μοντελοποιεί ως δίκτυα (γράφους). Εξετάζοντας τέτοια συστήματα υπό το πρίσμα της ανάλυσης κοινωνικών δικτύων γίνεται δυνατή η εξαγωγή χρήσιμων συμπερασμάτων για τα συστήματα αυτά, μέσω της μελέτης των τοπολογικών χαρακτηριστικών των αντίστοιχων γράφων. Τέτοια χρήσιμα συμπεράσματα μπορεί να είναι η ανακάλυψη των μοντέλων που ακολουθεί η εξέλιξή των συστημάτων στον χρόνο, ο εντοπισμός των πλέον επιδραστικών χρηστών, η ανίχνευση κοινοτήτων.

Στο κεφάλαιο αυτό, αρχικά, παρουσιάζονται τα βασικά στοιχεία της Θεωρίας γραφημάτων που είναι απαραίτητα για την κατανόηση των μεθόδων που αναλύονται στα επόμενα κεφάλαια. Δίνονται οι ορισμοί του γράφου, της μοντελοποίησής του μέσω πίνακα γειτνίασης, της γειτονιάς ενός κόμβου, του συντομότερου μονοπατιού κ.α. Στη συνέχεια, περιγράφονται συνοπτικά μετρικές οι οποίες είναι ιδιαίτερα χρήσιμες στη μελέτη δικτύων. Στο πλαίσιο αυτό, δίνονται οι ορισμοί των δυο ειδών κεντρικότητας που χρησιμοποιούνται ευρέως στα πλαίσια της παρούσας εργασίας. Αυτές αφορούν την κεντρικότητα κόμβου καθώς και την κεντρικότητα ενδιαμεσικότητας ακμής. Μετρικές όπως αυτές συντελούν στον εντοπισμό κόμβων και ακμών με επιθυμητές ιδιότητες. Τέλος, παρουσιάζονται κάποια από τα πλέον δημοφιλή μοντέλα σύνθετων δικτύων με την έμφαση να δίνεται στα δίκτυα ελεύθερης-κλίμακας (scale-free graphs), στα δίκτυα μικρού-κόσμου (small-world graphs) καθώς και στους τυχαίους γεωμετρικούς γράφους (Random Geometric Graphs). Αυτά τα μοντέλα δικτύων χρησιμοποιούνται τόσο στη βιβλιογραφία όσο και στη διατριβή αυτή για την αξιολόγηση της συμπεριφοράς αλγορίθμων που εφαρμόζονται σε γράφους.

Κεφάλαιο 3

Το Κεφάλαιο 3, αποτελεί το πρώτο μέρος της διατριβής και είναι αφιερωμένο στα προβλήματα της ανίχνευσης κοινοτήτων (community detection) και της ανακάλυψης συστάδων δεδομένων (data clustering). Τα μεγάλα διασυνδεδεμένα περιβάλλοντα που εξετάζονται, όπως οι Έξυπνες Πόλεις, παρουσιάζουν περιοχές με παραπλήσια χαρακτηριστικά όπως αυτά αποτυπώνονται από τις μετρήσεις κατάλληλων συσκευών όπως αισθητήρες. Ως παράδειγμα, τα γραφεία σε μια συγκεκριμένη πόλη αναμένεται να έχουν παραπλήσιες μετρήσεις όσον αφορά τα περιβαλλοντολογικά μεγέθη. Ακόμα, τα περιβάλλοντα που εξετάζονται σε αυτήν τη διατριβή περιλαμβάνουν και ανθρώπους οι οποίοι διατηρούν λογαριασμούς σε διαδεδομένα Κοινωνικά Δίκτυα. Ο εντοπισμός ομάδων παραπλήσιων αισθητήρων βοηθάει στην ανάλυση της τοπολογίας και μπορεί να οδηγήσει στον εντοπισμό περιττών αισθητήρων, ή στην πρόβλεψη των τιμών τους σε περίπτωση βλάβης. Η ανακάλυψη συστάδων από τα δεδομένα μέτρησης τέτοιων συσκευών (π.χ., θερμοκρασία, υγρασία, κ.λπ.) μεταφράζεται σε ένα πρόβλημα ανακάλυψης κοινοτήτων.

Η διαδικασία της ανίχνευσης κοινοτήτων είναι από τις πλέον σημαντικές για την ανάλυση κοινωνικών (αλλά και ευρύτερα) δικτύων. Καθώς τα σύνθετα δίκτυα, είτε χωρικά είτε κοινωνικά, εξελίσσονται στο χρόνο, οι κόμβοι τους τείνουν να οργανώνονται σε ομάδες (κοινότητες). Μια κοινότητα είναι ένα σύνολο από κόμβους που μοιράζονται μεταξύ τους περισσότερα κοινά χαρακτηριστικά από ότι με κόμβους που ανήκουν σε διαφορετικές κοινότητες. Σύμφωνα με την πλέον διαδεδομένη άποψη στη βιβλιογραφία, αναμένεται ότι, εντός ενός δικτύου, οι κόμβοι μιας κοινότητας είναι πιο πιθανό να συνδέονται μεταξύ τους σε σχέση με κόμβους που ανήκουν σε διαφορετικές κοινότητες. Σε γενικές γραμμές όμως, παρά το γεγονός ότι ο κλάδος της Θεωρίας γραφημάτων που ασχολείται με τα σύνθετα δίκτυα υφίσταται ήδη μερικές δεκαετίες, δεν υπάρχει ακριβής ορισμός της κοινότητας. Η ανίχνευση κοινοτήτων αφορά αναφέρεται και στο αλγοριθμικό κομμάτι που διαμερίζει τον γράφο σε σύνολα κόμβων έτσι ώστε να ικανοποιούνται ορισμένα κριτήρια. Εφόσον, όπως ειπώθηκε και παραπάνω δεν υπάρχει μοναδικός ορισμός της κοινότητας, έχει αναπτυχθεί μια πλειάδα τεχνικών-αλγορίθμων για την ανίχνευση κοινοτήτων όπου η κάθε μια εφαρμόζει τα δικά της ιδιαίτερα κριτήρια. Επιπλέον, στη βιβλιογραφία έχει αποδειχθεί ότι ποτέ δεν θα υπάρχει μοναδικός βέλτιστος αλγόριθμος για κάθε πιθανό τύπο δικτύου. Ταυτόχρονα αυτό συνεπάγεται και την ύπαρξη πολλών μεθόδων για την αξιολόγηση μιας διαμέρισης.

Όταν ένα πρόβλημα ανακάλυψης συστάδων σε δεδομένα, είτε αυτά αφορούν δομημένα σημασιολογικά δεδομένα όπως τα RDF, είτε μετρήσεις από αισθητήρες, μεταφράζεται σε ένα πρόβλημα ανακάλυψης κοινοτήτων σε δίκτυο, ακολουθούνται τα εξής βήματα. Αρχικά, τα μεγέθη που μετράει κάθε συσκευή θεωρούνται ως τιμές διαστάσεων ενός γεωμετρικού χώρου. Με αυτόν τον τρόπο κάθε μέτρηση αποτελεί ένα σημείο αρχικά σε αυτόν το χώρο και στη συνέχεια δημιουργείται ο πλήρης γράφος, με βάρη ακμών ίσα με τις αποστάσεις των σημείων. Ο τελικός γράφος εγγύτητας σχηματίζεται με χρήση του αλγορίθμου DMST ο οποίος βρίσκει ελάχιστα συνεκτικά δένδρα (minimum spanning trees) τα οποία και ενώνει σχηματίζοντας γράφο. Αυτός ο γράφος εγγύτητας, στη συνέχεια, θα χρησιμοποιηθεί για την ανακάλυψη κοινοτήτων. Σε αυτές τις περιπτώσεις, αναμένεται να υπάρχουν πυκνά συνδεδεμένες ομάδες (κοινότητες) κόμβων που ενώνονται μεταξύ τους με λίγες ακμές που μοιάζουν με γέφυρες, άρα έχουν υψηλή KEA. Ο αλγόριθμος που αναπτύσσεται και παρουσιάζεται σε αυτά τα κεφάλαια βασίζεται στη θεώρηση ότι η αφαίρεση κεντρικών ακμών οδηγεί στην ανακάλυψη κοινοτήτων που έχουν νόημα, δηλαδή που αντιστοιχούν σε συστάδες δεδομένων που παράγονται από παραπλήσιες συσκευές.

Παρόλο που η κάθε μεθοδολογία που έχει προταθεί αναπτύσσει τα δικά της κριτήρια για τον τρόπο με τον οποίο πραγματοποιεί την ανακάλυψη κοινοτήτων, είναι δυνατός ο χωρισμός τους σε ευρύτερες ομάδες. Στο κεφάλαιο αυτό παρουσιάζεται βιβλιογραφία σχετικά με τις διάφορες προσεγγίσεις στο ζήτημα της ανίχνευσης κοινοτήτων και στη συνέχεια αναλύεται ο αλγόριθμος Hyperbolic Girvan-Newman (HGN) που αναπτύχθηκε για τον σκοπό του εντοπισμού κοινοτήτων σε γράφους και η λειτουργία του συνδέθηκε με βάση-γράφο.

Ο αλγόριθμος HGN έχει ως έμπνευση το γνωστό αλγόριθμο των Girvan-Newman (GN). Ο αλγόριθμος αυτός λειτουργεί μέσω της ενσωμάτωσης του υπό εξέταση γράφου στον υπερβολικό γεωμετρικό χώρο. Με τον όρο ενσωμάτωση ενός δικτύου σε έναν γεωμετρικό χώρο εννοείται η απόδοση συντεταγμένων του χώρου σε κάθε κόμβο. Μέσω της ενσωμάτωσης, διάφοροι υπολογισμοί, όπως το μήκος του συντομότερου μονοπατιού που ενώνει δυο κόμβους μπορούν να υπολογιστούν ταχύτερα. Όσον αφορά τα σύνθετα δίκτυα, και ιδιαίτερα για όσα ακολουθούν κατανομή βαθμού τύπου "βαθμού-δύναμης" (power-law), έχει προταθεί στη βιβλιογραφία, ότι μια καλή επιλογή χώρου για την ενσωμάτωσή τους είναι ο υπερβολικός χώρος, για αυτό και επιλέγεται στα πλαίσια της διατριβής αυτής. Για την ενσωμάτωση ενός δικτύου στον υπερβολικό χώρο, έχουν αναπτυχθεί διάφορες τεχνικές και αλγόριθμοι. Ενδεικτικά αναφέρεται το μοντέλο Δημοφιλίας-Ομοιότητας (Popularity-Similarity). Στο μοντέλο αυτό, κάθε κόμβος λαμβάνει πολικές συντεταγμένες, όπου η ακτινική συνιστώσα αντανακλά τη δημοφιλία του, με κόμβους με μικρότερη ακτίνα να θεωρούνται δημοφιλέστεροι (π.χ., με περισσότερους γείτονες ή/και παλαιότεροι στο δίκτυο), ενώ η γωνιακή συνιστώσα χρησιμοποιείται για να εκφράσει την ομοιότητα μεταξύ ζευγών κόμβων. Στην εργασία αυτή, επιλέγεται ο αλγόριθμος ενσωμάτωσης Rigel. Ο αλγόριθμος αυτός ανήκει στην κατηγορία των αλγορίθμων που κατά τη διαδικασία της ενσωμάτωσης διατηρεί τις αποστάσεις μεταξύ των κόμβων (distance preserving) κοντά στο μήκος του συντομότερου μονοπατιού που τους συνδέει στον γράφο. Για τη λειτουργία του, ορίζονται κάποιοι κόμβοι ως "ορόσημα" (landmarks), που επιλέγονται, όπως έχει προταθεί ως βέλτιστη επιλογή στη σχετική βιβλιογραφία, οι κόμβοι με την υψηλότερη κεντρικότητα βαθμού. Αυτοί τοποθετούνται στο χώρο με τέτοιο τρόπο ώστε οι μεταξύ τους αποστάσεις να είναι κατά το δυνατόν ίσες με τις αποστάσεις τους στον γράφο και στη συνέχεια μέσω του αλγορίθμου Simplex κάθε κόμβος λαμβάνει συντεταγμένες επιλύοντας ένα πρόβλημα γραμμικού προγραμματισμού στο οποίο επιχειρείται η απόστασή του από ένα υποσύνολο οροσήμων να αντανακλά την απόστασή του από αυτά στον γράφο. Η μεθοδολογία του Rigel χρησιμοποιεί το μοντέλο του υπερβολοειδούς όπου η απόσταση ανάμεσα σε δυο κόμβους δίνεται από την Εξίσωση 3.2.

Μια βασική μετρική που χρησιμοποιείται στον αλγόριθμο GN είναι η κεντρικότητα ενδιαμεσικότητας ακμής (ΚΕΑ). Η μετρική αυτή εκφράζει το πόσο κεντρική είναι μια ακμή ως το ποσοστό των συντομότερων μονοπατιών μεταξύ κάθε ζεύγους κόμβων που διέρχονται από αυτήν. Η κεντρική ιδέα του αλγορίθμου είναι ότι οι ακμές που διασχίζουν κοινότητες (δηλαδή, ενώνουν κόμβους που ανήκουν σε διαφορετικές κοινότητες) είναι λιγότερες από όσες ενώνουν κόμβους της ίδιας κοινότητας. Συνεπώς, υπάρχουν ακμές με υψηλή τιμή ΚΕΑ που παρουσιάζουν ιδιότητες "γέφυρας" και η αφαίρεσή τους μπορεί να οδηγήσει στον κατακερματισμό του δικτύου (δηλαδή στην ανακάλυψη κοινότητας). Ο ακριβής υπολογισμός της ΚΕΑ είναι ιδιαίτερα απαιτητικός σε υπολογιστικό χρόνο για αυτό και προτείνονται οι παρακάτω τροποποιήσεις στον αλγόριθμο HGN σε σχέση με τον αρχικό GN.

Αρχικά, προτείνεται μια προσεγγιστική μέθοδος για τον υπολογισμό της ΚΕΑ, η οποία ονομάζεται Υπερβολική Κεντρικότητα Ενδιαμεσικότητας Ακμής (ΥΚΕΑ). Ο ψευδοκώδικας υπολογισμού της ΥΚΕΑ δίνεται από τον Αλγόριθμο 1. Για τον υπολογισμό της ΥΚΕΑ χρησιμοποιούνται οι συντεταγμένες που έχουν προκύψει μέσω της ενσωμάτωσης του δικτύου με τον αλγόριθμο Rigel. Οι αποστάσεις μεταξύ των κόμβων στον υπερβολικό χώρο χρησιμοποιούνται για τον υπολογισμό της ΥΚΕΑ. Παρόλο που η ΥΚΕΑ μιας ακμής δεν ταυτίζεται με την ακριβή τιμή της ΚΕΑ η οποία αποτελεί το κριτήριο στον αλγόριθμο GN, η κατάταξη των ακμών με βάση τις μετρικές αυτές είναι παραπλήσια, οπότε αναμένεται ακμές με υψηλή ΥΚΕΑ να είναι και ακμές με υψηλή ΚΕΑ. Τα αποτελέσματα που παρουσιάζονται στο Γράφημα 3.2 συνηγορούν υπέρ του συμπεράσματος αυτού, ιδιαίτερα για τα δίκτυα ελεύθερης κλίμακας.

Η δεύτερη τροποποίηση που προτείνεται είναι η αφαίρεση ακμών σε δέσμες (batches), αντί της αφαίρεσης κάθε φορά της ακμής με την υψηλότερη ΚΕΑ, όπως συμβαίνει στον GN. Στον HGN ορίζεται το μέγιστο πλήθος ακμών που μπορούν να αφαιρεθούν σε κάθε επανάληψη του αλγορίθμου. Συνοψίζοντας, ο αλγόριθμος λειτουργεί ως εξής. Ο γράφος ενσωματώνεται στον υπερβολικό χώρο, ρυθμίζοντας κατάλληλα τις παραμέτρους του αλγορίθμου Rigel. Κατόπιν, υπολογίζεται η ΥΚΕΑ κάθε ακμής. Στη συνέχεια, ακμές αφαιρούνται, μια τη φορά, έως ότου ένα από τα δυο παρακάτω ενδεχόμενα συμβεί. Είτε ο γράφος καθίσταται μη συνδεδεμένος, οπότε μια νέα κοινότητα έχει ανακαλυφθεί, είτε έχει αφαιρεθεί το μέγιστο πλήθος ακμών όπως ορίζεται από το μέγεθος της δέσμης που έχει επιλεγεί. Σε κάθε περίπτωση, επιλέγεται η μεγαλύτερη συνεκτική συνιστώσα και ενσωματώνεται ξανά στον υπερβολικό χώρο. Η διαδικασία αυτή επαναλαμβάνεται μέχρι να ανιχνευθεί ο ζητούμενος αριθμός κοινοτήτων. Από τα παραπάνω, προκύπτει ότι και το μέγεθος της δέσμης που θα επιλεγεί διαδραματίζει σημαντικό ρόλο στη διαμέριση που θα παραχθεί ως έξοδος του HGN αλλά και στο χρόνο εκτέλεσης του αλγορίθμου. Το γεγονός αυτό επιβεβαιώνεται και από το Γράφημα 3.3 όπου φαίνονται η παραγόμενη αρθρωτότητα αλλά και ο χρόνος εκτέλεσης ως παράμετρος του μεγέθους δέσμης. Από το διάγραμμα αυτό φαίνεται ότι παρόλο που η αύξηση του μεγέθους της δέσμης οδηγεί σε συντομότερο χρόνο εκτέλεσης, από ένα σημείο και έπειτα επηρεάζει αρνητικά την ποιότητα της παραγόμενης λύσης.

Οι παραπάνω δυο τροποποιήσεις συντελούν στην ταχύτερη εκτέλεση του αλγορίθμου σε σχέση με τον GN χωρίς να θυσιάζεται σημαντικά η ποιότητα της ευρισκόμενης διαμέρισης σε κοινότητες. Ακόμα, για να καταστεί ο αλγόριθμος ικανός να διαχειρίζεται μεγαλύτερα σύνολα δεδομένων πιο αποδοτικά, προτάθηκε η σύζευξη διάφορων συστατικών μερών του με κατάλληλες βάσεις δεδομένων. Έτσι οι συντεταγμένες των κόμβων μετά από κάθε ενσωμάτωση του δικτύου διατηρούνται σε κατάλληλα διαμορφωμένο πίνακα σχεσιακής βάσης δεδομένων, ενώ ο γράφος με πληροφορίες για κάθε κόμβο (π.χ., σε ποια κοινότητα ανήκει) αποθηκεύεται σε ειδική μη-σχεσιακή βάση-γράφο (Graph database), τη Neo4J. Η εισαγωγή μιας τέτοιας βάσης καθιστά δυνατή την αποθήκευση γράφων με μεγάλο πλήθος κόμβων καθώς και την αποδοτική υλοποίηση βασικών αλγορίθμων που χρειάζονται συχνά κατά τη διάρκεια εκτέλεσης του HGN, όπως το μήκος του ελάχιστου μονοπατιού, η εύρεση της μεγαλύτερης συνεκτικής συνιστώσας, κ.α. Μια εποπτική εικόνα της λειτουργίας του προτεινόμενου αλγορίθμου δίνεται από το διάγραμμα ροής της Εικόνας 3.1, όπου φαίνονται και τα σημεία όπου χρειάζονται οι βάσεις δεδομένων.

Για την αξιολόγηση των διαμερίσεων, στη διατριβή αυτή χρησιμοποιούνται οι εξής μέθοδοι ανάλογα με το αν η διαμέριση σε κοινότητες είναι γνωστή ή όχι. Στην περίπτωση που είναι γνωστή, χρησιμοποιείται η μετρική της Κανονικοποιημένης Κοινής Πληροφορίας (Normalized Mutual Information, NMI) ως μέτρο του πόσο κοντά στην πραγματική διαμέριση είναι η παραγόμενη από τον HGN. Σε περίπτωση άγνωστης διαμέρισης, χρησιμοποιείται η μετρική της αρθρωτότητας (modularity) που δίνεται στην Εξίσωση 3.1. Η μετρική αυτή αποτελεί μια ένδειξη για το πόσο καλύτερη είναι μια διαμέριση από μια εντελώς τυχαία διαμέριση του γράφου σε κοινότητες και είναι μια από τις πιο δημοφιλείς μετρικές για την αξιολόγηση διαμερίσεων γράφων. Για γράφους που παρουσιάζουν δομή σαν αυτή που περιγράφηκε παραπάνω, δηλαδή περισσότερες ακμές εντός των κοινοτήτων και λιγοστές που τις διασχίζουν, υψηλότερες τιμές αποτελούν ένδειξη ποιοτικότερης διαμέρισης.

Στο πλαίσιο της αξιολόγησής του ο HGN συγκρίθηκε τόσο με τον GN όπου πέτυχε είτε τα ίδια αποτελέσματα σε πολύ μικρότερο χρόνο, όπως φαίνεται και από τα αποτελέσματα του Πίνακα 3.7 είτε πέτυχε αποτελέσματα με κοντινό NMI σε σχέση με τον GN, όντας πολύ γρηγορότερος στην εκτέλεσή του και δικαιολογώντας έτσι την επιλογή του ως μια καλή εναλλακτική σε μεγαλύτερα δίκτυα από αυτά στα οποία μπορεί να εκτελεστεί αποδοτικά ο GN, όπως είναι φανερό από τον Πίνακα 3.3. Ακόμα, συγκρίθηκε και με άλλους, πιο σύγχρονους, αλγορίθμους επιτυγχάνοντας καλύτερες διαμερίσεις, δηλαδή πιο κοντά στις πραγματικές, σε δίκτυα με γνωστές κοινότητες. Ενδεικτικά, στην Εικόνα 3.4 μπορεί να φανεί η ακρίβεια του HGN σε σχέση με άλλους 3 γνωστούς αλγορίθμους.

Επιπλέον, ο αλγόριθμος HGN χρησιμοποιήθηκε στην ανάλυση δεδομένων τα οποία αντιστοιχούν σε μετρήσεις αισθητήρων από το Santander της Ισπανίας, μια πραγματική Έξυπνη Πόλη με χιλιάδες έξυπνες συσκευές που καταγράφουν ένα μεγάλο σύνολο δεδομένων, όπως περιβαλλοντολογικά δεδομένα, δεδομένα σχετικά με τη διαχείριση απορριμάτων, την κατανάλωση ενέργειας, κ.λπ. Κατά τη διάρκεια των πειραμάτων συλλέχθηκαν δεδομένα από αυτές τις συσκευές με διάφορους ρυθμούς συλλογής. Δημιουργώντας κατάλληλα δίκτυα που αντιστοιχούσαν σε διαφορετικές χρονικές περιόδους και εφαρμόζοντας τον HGN, ανιχνεύθηκαν κοινότητες με υψηλή αρθρωτότητα, συγκρίσιμη με αυτή του GN αλλά και συγκρίσιμη με την αρθρωτότητα που παράγεται από αλγόριθμο διαμέρισης μεγιστοποίησης της αρθρωτότητας (modularity maximization). Αυτά τα αποτελέσματα παρουσιάζονται στις Εικόνες 3.5 - 3.8. Ακόμα, επιβεβαιώθηκε ο σκοπός του πειράματος, δηλαδή αφενός ότι ο αλγόριθμος HGN είναι ικανός να παράγει διαμερίσεις υψηλής αρθρωτότητας όταν εφαρμόζεται σε γράφους εγγύτητας που παράχθηκαν από τέτοιου είδους δεδομένα, και αφ' ετέρου ότι σε ένα τέτοιο περιβάλλον υπάρχουν συσκευές, όπως φαίνεται στον Πίνακα 3.8, οι οποίες παραμένουν διαρκώς στις ίδιες κοινότητες κατά τη χρονική διάρκεια εξέλιξης του πειράματος, γεγονός που σημαίνει ότι παράγουν διαρκώς παραπλήσιες μετρήσεις. Τα συμπεράσματα θέτουν τις βάσεις για την ανάπτυξη ενός πλαισίου όπου η ανίχνευση κοινοτήτων μπορεί να χρησιμοποιηθεί για τη ρύθμιση του ρυθμού συλλογής δεδομένων αλλά και τη μείωση της κατανάλωσης ενέργειας.

Τέλος, ο αλγόριθμος δοκιμάστηκε με είσοδο δεδομένα τύπου RDF, τα οποία αποτελούν τριπλέτες της μορφής *(υποκείμενο, κατηγορούμενο, αντικείμενο)*. Τα δεδομένα αφορούσαν σε στοιχεία επιστημονικών δημοσιεύσεων (συγγραφείς, τίτλος, χρονολογία, αριθμός σελίδων κ.α.) που πραγματοποιήθηκαν σε συγκεκριμένη ομάδα συνεδρίων και επιστημονικών περιοδικών. Αυτές οι τριπλέτες, αφού καθαρίστηκαν από δεδομένα που δεν ήταν συμπληρωμένα σωστά, αντιμετωπίστηκαν ως σημεία ενός χώρου και ο γράφος εγγύτητας δημιουργήθηκε μέσω σταθμισμένου μέσου κατάλληλα επιλεγμένων μετρικών για κάθε ένα στοιχείο της τριπλέτας. Η χρήση του σταθμισμένου μέσου επιτρέπει την επιλογή των βαρών καταλλήλως, ανάλογα με την επιδιοκώμενη διαμέριση. Για παράδειγμα, αν η διαμέριση είναι επιθυμητό να εστιάσει στο *κατηγορούμενο* των δεδομένων RDF, ο συντελεστής που αντιστοιχεί σε αυτήν την θέση πρέπει να είναι μεγαλύτερος από ότι για τα άλλα δυο στοιχεία της τριπλέτας. Με αυτόν τον τρόπο, αντανακλώνται καλύτερα οι διαφορές των τριπλέτων στο επιλεγμενό μέρος, χωρίς όμως να υποβαθμίζεται η συνεισφορά των αποστάσεων για τα υπόλοιπα δυο κομμάτια. Επιπλέον, δημιουργήθηκε πλατφόρμα για την οπτικοποίηση και την εξερεύνηση με εύληπτο τρόπο τέτοιων συνόλων δεδομένων. Με ζητούμενο τον διαχωρισμό των κοινοτήτων με βάση το *κατηγορούμενο*, ο HGN πέτυχε 100% ακρίβεια στις διαμερίσεις του. Στιγμιότυπα της πλατφόρμας αυτής φαίνονται στις Εικόνες 3.10-3.12. Εκτός της ορθής διαμέρισης του γράφου, η πλατφόρμα που αναπτύχθηκε είναι αρκετά εύχρηστη και παρουσιάζει με κομψό τρόπο πληροφορίες σχετικά με τα δεδομένα εισόδου, ενώ παρέχει πεδία αναζήτησης, αλλά και δυνατότητα εκτέλεσης αναζητήσεων στη βάση.

Κεφάλαιο 4

Αναπόσπαστο κομμάτι των διασυνδεδεμένων και αλληλεξαρτώμενων συστημάτων που εξετάζονται στα πλαίσια της διατριβής είναι τα Κοινωνικά Δίκτυα (ΚΔ, Online Social Networks). Σε αυτά οι χρήστες αλληλεπιδρούν ανταλλάσσοντας μηνύματα αλλά και κάνοντας γνωστές τις αλληλεπιδράσεις τους σε συστάσεις. Ακόμα, με τη χρήση κατάλληλων εφαρμογών οι κάτοικοι της πόλης (ή γενικότερα του διασυνδεδεμένου περιβάλλοντος) αποκτούν πρόσβαση σε δεδομένα που καταγράφονται από διάφορες συσκευές μέσω της χρήσης κατάλληλων εφαρμογών. Εύλογα προκύπτει ότι για τη διάδοσή τους οι εφαρμογές αυτές χρειάζονται τη διαφήμισή τους. Στα ΚΔ πλέον, αναπόσπαστο κομμάτι τους αποτελούν τα συστήματα συστάσεων (ΣΣ). Τα συστήματα αυτά αποτελούν έναν από τους κύριους μηχανισμούς διάχυσης πληροφορίας στο δίκτυο. Τα ΣΣ επιλέγουν χρήστες σύμφωνα με κατάλληλα επιλεγμένα κριτήρια στους οποίους και προτείνουν αντικείμενα, τα οποία μπορεί να είναι οποιαδήποτε μορφή πληροφορίας (π.χ., αρχεία βίντεο, προτάσεις για φίλους, εφαρμογές, κ.λπ.).

Το Κεφάλαιο 4 πραγματεύεται το ζήτημα της ανάθεσης συστάσεων στους χρήστες ενός κοινωνικού δικτύου. Το πλαίσιο στο οποίο εξετάζεται το πρόβλημα της εύρεσης των κατάλληλων χρηστών για σύσταση κάθε αντικειμένου είναι τα Συστήματα Συστάσεων με Επίγνωση της Πληροφοριακής Διάδοσης (Information Diffusion Aware Recommender Systems, IDARS). Τέτοια συστήματα συστάσεων (ΣΣ) προβλέπουν και λαμβάνουν υπόψιν τον τρόπο με τον οποίο διαχέεται η πληροφορία σε ένα ΚΔ μέσω των αλληλεπιδράσεων των χρηστών. Το γεγονός αυτό επιτρέπει την αύξηση της ποικιλίας των προταθέντων αντικειμένων, αποφεύγοντας συστάσεις για αντικείμενα που οι χρήστες θα μάθουν έτσι και αλλιώς από τη δραστηριότητα του περίγυρου τους στα ΚΔ. Βασική υπόθεση για τα IDARS συστήματα, που επιβεβαιώνεται και από τη βιβλιογραφία, είναι ότι οι χρήστες προτιμούν τις συστάσεις που προέρχονται από φιλικά πρόσωπα παρά από ένα ΣΣ.

Παρά το γεγονός ότι μεθοδολογίες τύπου IDARS μειώνουν σημαντικά την ποσότητα των επαναλαμβανόμενων συστάσεων, η υπερφόρτωση χρηστών με πληροφορία είναι ακόμα πιθανή, με συνέπειες τη μη παρακολούθηση των συστάσεων, την μείωση της ικανοποίησης του χρήστη από το ΚΔ, ακόμα και την απομάκρυνσή του. Στη διατριβή αυτή, η έμφαση δίνεται στον σχεδιασμό ενός ΣΣ τύπου IDARS που μεγιστοποιεί τη συνάφεια των συστάσεων, δηλαδή βρίσκει ποιες συστάσεις και σε ποιους χρήστες οδηγούν στο μέγιστο σκορ συνάφειας σε όλο το δίκτυο, θέτοντας όμως αυστηρούς περιορισμούς που εγγυούνται ότι οι χρήστες δεν θα υπερφορτωθούν από μεγάλη ποσότητα πληροφορίας. Στόχος είναι ο σχεδιασμός μεθόδων που συντελούν στη διάδοση εφαρμογών σε ένα μεγάλο διασυνδεδεμένο περιβάλλον όπως μια Έξυπνη Πόλη με τις περισσότερες συστάσεις να είναι έμμεσες, δηλαδή αυτές που παράγονται μέσω των αλληλεπιδράσεων των χρηστών (π.χ., κοινοποιήσεις, προσκλήσεις, προωθήσεις σε φίλους, κ.λπ.) παρά άμεσες που είναι οι συστάσεις που προέρχονται κατευθείαν από το IDARS. Η καινοτομία της διατριβής έγκειται στο γεγονός πως για πρώτη φορά λαμβάνονται σύνθετοι περιορισμοί ως προς την ποσότητα και το είδος της πληροφορίας που μπορεί να λάβει ένας χρήστης κατά τη διάρκεια μιας δεδομένης χρονικής περιόδου.

Πιο συγκεκριμένα, εισάγονται δυο είδη περιορισμών για κάθε χρήστη. Το πρώτο αφορά τον μέγιστο αριθμό των επαναλήψεων της σύστασης του ίδιου αντικειμένου (διπλότυπα). Το δεύτερο αφορά το πλήθος των διαφορετικών αντικειμένων στα οποία μπορεί να εκτεθεί χωρίς να χαλάσει η εμπειρία του από την πλατφόρμα. Οι περιορισμοί αυτοί διαφέρουν για κάθε ζεύγος χρήστη-αντικειμένου, δίνοντας έτσι έμφαση στη μοναδικότητα των προτιμήσεων κάθε ατόμου. Επιπλέον, κάθε χρήστης έχει μια τιμή συνάφειας (relevance) προς κάθε αντικείμενο η οποία μπορεί να προσδιοριστεί από ιστορικά στοιχεία για τη συμπεριφορά του χρήστη στην πλατφόρμα ή να εξαχθεί με μεθόδους που χρησιμοποιούνται από παραδοσιακά ΣΣ, όπως τα βασισμένα-στο-περιεχόμενο (content-based) ή συνεργατικού φιλτραρίσματος (collaborative filtering). Ακόμα, το κοινωνικό δίκτυο θεωρείται ως ένας κατευθυνόμενος γράφος με βάρη, με ακμές που φανερώνουν το ποιος χρήστης επηρεάζει ποιον. Το βάρος κάθε ακμής είναι το μέτρο της επιρροής αυτής. Το βάρος αυτό εξάγεται παρατηρώντας το ιστορικό των αλληλεπιδράσεων μεταξύ κάθε ζεύγους χρηστών και μπορεί να υπολογιστεί από την Εξίσωση 4.2.

Στο μοντέλο που αναπτύχθηκε, βασισμένο και στη βιβλιογραφία, κάθε σύσταση σε χρήστη προκαλεί διάχυση της σχετικής πληροφορίας κατά μήκος κατευθυνόμενων μονοπατιών που εκκινούν από τους χρήστες οι οποίοι επιλέγονται από το ΣΣ για παροχή άμεσων συστάσεων.

Η έκταση του μονοπατιού σχετίζεται τόσο με το μέγεθος της επιρροής κάθε ακμής που είναι υποψήφια να ενταχθεί σε αυτό αλλά και με το μέγεθος της συνάφειας που έχει κάθε υποψήφιος χρήστης με το προταθέν από το ΣΣ αντικείμενο. Για τις ανάγκες της διατριβής, τροποποιείται γνωστός αλγόριθμος Monte Carlo δειγματοληψίας (Monte Carlo sampling), ο οποίος επιστρέφει έναν κατευθυνόμενο γράφο για κάθε πιθανό αντικείμενο που μπορεί να προταθεί στους χρήστες. Από αυτόν τον γράφο μπορεί να εξαχθεί άμεσα η πληροφορία σχετικά με το σε ποιους χρήστες θα φτάσει η πληροφορία (η σύσταση του αντικειμένου) ανάλογα με το ποιοι χρήστες θα επιλεχθούν για την άμεση σύστασή του. Ο γράφος αυτός ονομάζεται Γράφος Επιρροής (Influence Graph, ΓΕ). Ως επιπλέον περιορισμός στο προταθέν μοντέλο εισάγεται και η απαγόρευση της ίδιας άμεσης σύστασης σε γειτονικούς κόμβους του ΓΕ.

Σκοπός κάθε επιτυχημένου ΣΣ πρέπει να είναι η σύσταση σχετικών αντικειμένων σε κάθε χρήστη. Σε διαφορετική περίπτωση το ΣΣ θα έχανε το νόημά του και θα προκαλούσε σύγχυση στους χρήστες. Το πρόβλημα της εύρεσης των συστάσεων που μεγιστοποιούν τη συνολική συνάφεια, ως το άθροισμα της συνάφειας που έχει κάθε χρήστης ως προς κάθε αντικείμενο το οποίο του προτείνεται είτε άμεσα είτε εμμέσως, με την τήρηση των περιορισμών που αναφέρθηκαν παραπάνω, μοντελοποιείται ως ένα πρόβλημα μεγιστοποίησης το οποίο διαιρείται σε δυο υποπροβλήματα. Το ένα από αυτά τα προβλήματα αποτελεί επέκταση ενός προβλήματος της κλάσης NP-hard με επιπλέον περιορισμούς και έτσι τεκμηριώνεται και θεωρητικά η επιλογή που γίνεται για την αντιμετώπιση του προβλήματος με μη βέλτιστους, άπληστους αλγορίθμους.

Πιο αναλυτικά, το πρόβλημα διαιρείται σε δυο υποπροβλήματα τα οποία πρέπει να επιλυθούν διαδοχικά προκειμένου να παραχθεί η λύση. Το πρώτο υποπρόβλημα ασχολείται με την εύρεση, για κάθε αντικείμενο, του συνόλου από χρήστες το οποίο μεγιστοποιεί τη συνολική συνάφεια για το αντικείμενο αυτό. Τα σύνολα που δημιουργούνται, είναι γνωστά στη θεωρία γραφημάτων ως Ανεξάρτητα Σύνολα (ΑΣ, Independent Sets). Για εξεύρεση των κατάλληλων ΑΣ αναπτύσσονται δυο διαφορετικές προσεγγίσεις. Ο GRIS, ο οποίος περιγράφεται από τον ψευδοκώδικα που δίνεται στον Αλγόριθμο 2, και είναι ένας άπληστος αλγόριθμος που παράγει για κάθε αντικείμενο ένα ΑΣ τοποθετώντας εντός του κόμβους που επιτυγχάνουν υψηλό λόγο συνολικής συνάφειας για το αντικείμενο (συνάφεια για τον εαυτό τους και τους χρήστες που επηρεάζουν) προς αριθμό γειτόνων, υπό τη συνθήκη ότι δεν παραβιάζεται ο περιορισμός επί των διπλότυπων συστάσεων αντικειμένων. Σε αντίθεση με άλλες προσεγγίσεις, η συνεισφορά ενός κόμβου στη συνάφεια επανυπολογίζεται κάθε φορά, αφού αν έχει ήδη συνεισφέρει μια φορά δεν χρειάζεται να ξαναπροστεθεί η συνάφειά του στο συνολικό σκορ. Ο δεύτερος αλγόριθμος για το πρώτο υποπρόβλημα που προτείνεται και ονομάζεται GAIS, δίνεται στον Αλγόριθμο 3 και είναι ένας γενετικός αλγόριθμος όπου η λύση για κάθε αντικείμενο, δηλαδή το ΑΣ, μοντελοποιείται ως ένα χρωμόσωμα και θεωρείται συνάρτηση καταλληλότητας (fitness function) η οποία ευνοεί τα χρωμοσώματα εκείνα που δεν παρουσιάζουν παραβιάσεις των περιορισμών του προβλήματος. Για την εκτέλεσή του ακολουθούνται οι παραδοσιακές πράξεις της διασταύρωσης και της μετάλλαξης για έναν προκαθορισμένο αριθμό εποχών ίσο με 1000. Παρόλο που ο GAIS δύναται να βρει τη βέλτιστη λύση σε περιπτώσεις που αυτή δεν γίνεται να παραχθεί από τον GRIS, όπως φαίνεται και στο παράδειγμα της Εικόνας 4.3, είναι αρκετά χρονοβόρος όπως μπορεί να φανεί και από τον Πίνακα 4.2, κάτι που καθιστά τον GRIS καλύτερη επιλογή για μεγαλύτερα δίκτυα.

Έχοντας καταλήξει σε ένα ΑΣ ανά αντικείμενο, η διαδικασία συνεχίζεται με την επίλυση του δεύτερου υπο-προβλήματος που αφορά την κατανομή των άμεσων συστάσεων σε χρήστες. Για την επίλυσή του προβλήματος αυτού, αναπτύχθηκαν δυο άπληστοι αλγόριθμοι. Ο RecIt (Αλγόριθμος 4) εστιάζει στα αντικείμενα. Εξετάζει κάθε ΑΣ κατά φθίνουσα σειρά συνολικής συνάφειας και αναθέτει σε κάθε χρήστη που ανήκει στο ΑΣ το αντικείμενο που αντιστοιχεί στο υπό εξέταση ΑΣ όσο δεν παραβιάζεται ο περιορισμός επί του πλήθους των διαφορετικών αντικειμένων που μπορεί να λάβει κάθε χρήστης στη γειτονιά του χρήστη στο ΓΕ. Ο δεύτερος αλγόριθμος είναι ο RecUs, ο οποίος παρουσιάζεται στον Αλγόριθμο 5 και εστιάζει στους χρήστες, ταξινομώντας τους κατά φθίνουσα σειρά επιδραστικότερων (σε σχέση με τη συνάφεια που επιτυγχάνεται με την άμεση σύσταση αντικειμένου στο χρήστη). Παρόμοια με την προηγούμενη περίπτωση γίνεται ανάθεση συστάσεων όσο δεν παραβιάζεται το κριτήριο για το πλήθος των διαφορετικών αντικειμένων.

Ο συνδυασμός ενός εκ των GRIS και GAIS με έναν εκ των RecIt και RecUs, δημιουργεί το πλαίσιο των Κοινωνικά Περιορισμένων Συστάσεων (Socially Constraint Recommendations, SCoRe) που δημιουργήθηκε στα πλαίσια της παρούσας διατριβής. Η συνεισφορά του πλαισίου αυτού συνίσταται στην ικανότητά του να παρέχει συστάσεις που επιτυγχάνουν υψηλή συνολική συνάφεια, χωρίς να υπερφορτώνει τους χρήστες με πληροφορία.

Για την αξιολόγηση της προτεινόμενης μεθοδολογίας, το πλαίσιο SCoRe συγκρίθηκε με άλλον αλγόριθμο τύπου IDARS της βιβλιογραφίας, τον DifRec, ο οποίος δεν λαμβάνει υπόψιν και τους δύο περιορισμούς που εισάγονται στη διατριβή. Από τον Πίνακα 4.1 φαίνεται ότι η εφαρμόζοντας τον DifRec πάνω στα ίδια δίκτυα υπάρχουν παραβιάσεις σε αρκετούς χρήστες, όσον αφορά τους περιορισμούς που έχουν τεθεί, κάνοντας έτσι σαφή την ανάγκη για τη λήψη πιο σύνθετων περιορισμών. Όσον αφορά τον SCoRe, δεν υπάρχει καμία παραβίαση, εφόσον η λύση για το πρώτο υποπρόβλημα εγγυάται την μη ύπαρξη παραπάνω διπλότυπων από τα όρια του κάθε χρήστη, ενώ η λύση για το δεύτερο εγγυάται ότι κανείς χρήστης δεν θα λάβει περισσότερα διαφορετικά αντικείμενα από όσα θα άντεχε. Στη συνέχεια, συγκρίθηκαν μεταξύ τους οι δυο αλγόριθμοι σχετικά με το πρώτο υποπρόβλημα από όπου προέκυψε ότι παρά το γεγονός ότι ο GAIS οδηγεί στην ανίχνευση μεγαλύτερων ΑΣ κατά μέσο όρο από τον GRIS (Εικόνα 4.6) δεν οδηγεί σε μεγαλύτερο σκορ συνολικής συνάφειας, όπως άλλωστε φαίνεται και από την Εικόνα 4.5. Ακόμα, μέσω των προσομοιώσεων που έγιναν για την αξιολόγησή του, φάνηκε ότι οδηγεί στη διάδοση της συντριπτικής πλειοψηφίας των αντικειμένων εντός του δικτύου (Εικόνες 4.7 και 4.8. Τέλος, αν δεν χρησιμοποιηθεί για τη λύση του πρώτου υπο-προβλήματος ο GAIS, ο SCoRe μπορεί να αποφανθεί για την κατανομή συστάσεων σε μικρό χρονικό διάστημα, όπως φαίνεται και στην Εικόνα 4.10.

Κεφάλαιο 5

Τα διασυνδεδεμένα περιβάλλοντα αναμένεται να διαθέτουν υποδομές αποτελούμενες από πληθώρα συσκευών αλλά και χρήστες. Όλες αυτές οι συσκευές, καταμετρούν, παράγουν, μεταβάλλουν και μεταδίδουν δεδομένα, συντελώντας έτσι στην κατακόρυφη αύξηση της κίνησης εντός του δικτύου. Για τους σκοπούς των τηλεπικοινωνιών, ο χώρος χωρίζεται σε κυψέλες. Εντός τέτοιων κελιών, υπάρχει ειδικός εξοπλισμός προκειμένου να γίνεται δυνατή η σύνδεση των έξυπνων συσκευών με το Διαδίκτυο. Παραδείγματα τέτοιων συσκευών είναι οι Σταθμοί Βάσης (ΣΒ, Base Stations), τα eNodeBs, EPCs κ.λπ. Οι χρήστες που βρίσκονται εντός τέτοιων περιβάλλοντων και χρησιμοποιούν τις σχετικές εφαρμογές, χρειάζονται εύκολη και άμεση πρόσβαση σε μεγάλο πλήθος δεδομένων, είτε αυτά αφορούν μετρήσεις έξυπνων συσκευών είτε πολυμέσα.

Το Κεφάλαιο 5 είναι αφιερωμένο στο ζήτημα της ενίσχυσης της ποιότητας εμπειρίας όπως την αντιλαμβάνονται οι χρήστες ενός συστήματος με χρήση τεχνικών για τη προσωρινή αποθήκευση (caching) στα άκρα του δικτύου (network edge) με τη βοήθεια συστάσεων από ΣΣ. Η τεχνική της αποθήκευσης στα άκρα του δικτύου, είναι μια χρήσιμη πρακτική η οποία μπορεί να οδηγήσει στη βελτίωση πολλών παραμέτρων της επίδοσης ενός δικτύου. Αρχικά, η προσωρινή αποθήκευση μπορεί να περιορίσει την επαναλαμβανόμενη μετάδοση των ίδιων δεδομένων (π.χ., από δημοφιλή βίντεο), από την κεντρική δομή ενός δικτύου προς τον τελικό χρήστη, συντελώντας έτσι στην αποσυμφόρηση συνδέσεων του δικτύου. Ακόμα, η προσωρινή αποθήκευση στα άκρα του δικτύου είναι μια βασική παράμετρος για τα σύγχρονα αλληλοεξαρτώμενα περιβάλλοντα καθώς συντελεί στη πιο γρήγορη λήψη των δεδομένων από τους τελικούς χρήστες κάτι που ενισχύει τον βαθμό της ικανοποίησης του χρήστη από την υπηρεσία.

Η προσωρινή αποθήκευση δεδομένων μπορεί να γίνει στους Σταθμούς Βάσης χρησιμοποιώντας ένα ποσοστό της μνήμης τους, σε ειδικά διαμορφωμένους βοηθητικούς κόμβους, δηλαδή μικρές συσκευές με περιορισμένη μνήμη αλλά ακόμα και στις συσκευές των χρηστών. Ειδικά σήμερα, με την ανάπτυξη τεχνολογιών που επιτρέπουν τη γρήγορη μεταφορά δεδομένων μεταξύ συσκευών όπως είναι το Bluetooth και το Wifi-Direct, τέτοιες λύσεις γίνονται ολοένα και πιο εφικτές και συγκεντρώνουν μεγάλο ενδιαφέρον όπως φαίνεται και από τον όγκο της σχετικής βιβλιογραφίας.

Ακόμα, είναι γνωστό ότι οι συστάσεις που παράγονται από ΣΣ διαμορφώνουν τις απαιτήσεις των χρηστών σε περιεχόμενο. Έτσι, τα ΣΣ μπορούν να χρησιμοποιηθούν για να ωθήσουν τους χρήστες να ζητήσουν δεδομένα που έχουν ήδη αποθηκευτεί κοντά τους και τα οποία φυσικά να είναι κοντά στις προτιμήσεις τους. Ο υπολογισμός του κατάλληλου περιεχομένου που πρέπει να αποθηκευτεί σε κάθε συσκευή διαθέσιμη να παρέχει χώρο αποθήκευσης και η δημιουργία κατάλληλων συστάσεων στους χρήστες, μοντελοποιείται ως ένα πρόβλημα με στόχο τη μεγιστοποίηση της ικανοποίησης των χρηστών που οφείλεται στη παροχή ποιοτικών συστάσεων για περιεχόμενο που μπορεί να τους μεταδοθεί με χαμηλή καθυστέρηση.

Στο πλαίσιο της διατριβής εξετάζονται μέθοδοι για την αποθήκευση δεδομένων στα άκρα του δικτύου με χρήση, εκτός του εξειδικευμένου εξοπλισμού, όπως είναι οι ΣΒ, και περιορισμένου χώρου μνήμης στις συσκευές των χρηστών (user equipment). Δυο διαφορετικές προσεγγίσεις αναπτύχθηκαν και αξιολογήθηκαν. Η πρώτη αφορά τη θεώρηση των χρηστών ως στατικούς στο χώρο και την εκμετάλλευση εκτός από τη θέση τους και των κοινωνικών σχέσεων που εξάγονται μέσω της συμμετοχής των κατόχων των συσκευών σε ΚΔ, ενώ οι πιθανότητες αναζήτησης κάθε αντικειμένου από κάθε χρήστη μπορούν να εκτιμηθούν μέσω της λειτουργίας ενός ΣΣ στη πλατφόρμα. Στο μοντέλο της δεύτερης προσέγγισης, οι χρήστες θεωρείται ότι κινούνται εντός της εξεταζόμενης περιοχής και στόχος είναι η προσωρινή αποθήκευση δεδομένων σε ορισμένους κατάλληλα επιλεγμένους χρήστες σε συνδυασμό και η σύνθεση λίστας συστάσεων για όλους του χρήστες να οδηγεί στη γρήγορη λήψη αντικειμένων με υψηλή συνάφεια για κάθε χρήστη της πλατφόρμας.

Ειδικότερα, στην πρώτη περίπτωση, εξετάζονται τέσσερις παραλλαγές για το πρόβλημα της αποθήκευσης των κατάλληλων δεδομένων στις συσκευές των χρηστών με δεδομένες τις πιθανότητες να αιτηθεί κάθε χρήστης το κάθε αντικείμενο όπως έχουν εκτιμηθεί με τη βοήθεια ΣΣ. Για κάθε μια από αυτές τις παραλλαγές το πρόβλημα μοντελοποιείται ως ένα πρόβλημα μεγιστοποίησης του ποσοστού ευστοχίας με τους κατάλληλους περιορισμούς κάθε φορά. Επειδή η εύρεση των βέλτιστων λύσεων δεν είναι δυνατό να ευρεθεί σε εύλογο χρονικό διάστημα, για την επίλυσή τους αναπτύσσονται τέσσερις άπληστοι αλγόριθμοι οι οποίοι αξιοποιούν είτε ένα περιορισμένο υποσύνολο από τους χρήστες της πλατφόρμας είτε το σύνολό τους, ενώ η αποθήκευσή των δεδομένων γίνεται είτε προκαταβολικά είτε με διαδραστικό τρόπο. Σε όλες τις περιπτώσεις, χρησιμοποιείται ο γράφος που προκύπτει από την τομή του γράφου που αποκαλύπτει την τοποθεσία των κόμβων εντός του κελιού καθώς και το ποιοι κόμβοι βρίσκονται σε απόσταση που τους επιτρέπει να ανταλλάξουν δεδομένα με τον γράφο ομοιότητας, ο οποίος ενώνει κόμβους που θεωρείται ότι έχουν παραπλήσιες προτιμήσεις. Ο τελικός γράφος που προκύπτει παρουσιάζει το ποιοι κόμβοι είναι και σε κοντινή απόσταση και είναι παραπλήσιοι.

Σε κάθε μια από τις τέσσερις προσεγγίσεις ισχύουν οι ακόλουθες παραδοχές. Σε όλες τις περιπτώσεις η μνήμη του ΣΒ είναι η πρώτη που γεμίζει με τα πιο δημοφιλή αντικείμενα, που είναι τα αντικείμενα που είναι πιθανότερο να ζητηθούν από τους περισσότερους χρήστες. Όλοι οι χρήστες έχουν άμεση πρόσβαση στη μνήμη του ΣΒ καθώς θεωρείται ότι ο ΣΒ καλύπτει ολόκληρη τη περιοχή που περιλαμβάνει τους χρήστες. Κάθε αντικείμενο έχει ένα ορισμένο μέγεθος και οι συσκευές των χρηστών θεωρείται ότι έχουν διαθέσιμη μνήμη μικρότερη από αυτήν του ΣΒ.

Η πρώτη προσέγγιση που παρουσιάζεται ο αλγόριθμος PL-CAUSE, αρχικά τοποθετεί στη μνήμη του Σταθμού Βάσης τα δημοφιλέστερα αντικείμενα, λύνοντας ένα πρόβλημα σακιδίου (knapsack) με τις αξίες των αντικειμένων να υπολογίζονται από την Εξίσωση 5.4. Ο γράφος του συστήματος, διαμερίζεται σε κοινότητες με χρήση αλγορίθμου μεγιστοποίησης της αρθρωτότητας. Έπειτα, για κάθε κοινότητα εντοπίζεται ποιοι από κόμβους-συσκευές χρηστών που θα χρησιμοποιηθούν ως χώροι αποθήκευσης (Clusterheads, CHs). Για τον υπολογισμό του πλήθους των CHs, λαμβάνεται υπόψιν η ανισομετρία στα μεγέθη των κοινοτήτων που προκύπτει πολλές φορές από τον αλγόριθμο μεγιστοποίησης της αρθρωτότητας και αντί για προεπιλεγμένο αριθμό CHs ανά κοινότητα, επιλέγεται κατάλληλος αριθμός για να επιτευχθεί το επιθυμητό ποσοστό κάλυψης της κάθε κοινότητας από CHs (ποσοστό χρηστών που έχουν ως γείτονα τους ένα CH). Για τον σκοπό της επιλογής των κατάλληλων κόμβων ως CHs ορίζεται μια νέα μετρική κεντρικότητας που δίνεται από την Εξίσωση 5.3, που δίνει υψηλές τιμές για όσους χρήστες έχουν μεγάλο βαθμό και μεγάλο χώρο αποθήκευσης. Επίσης, απαλείφεται ο περιορισμός για μοναδική αποθήκευση κάθε αντικειμένου στο δίκτυο. Το πρόβλημα στη περίπτωση του PL-CAUSE αντιστοιχεί σε ένα πρόβλημα του μη-περιορισμένου πολλαπλού σακιδίου με μεταβλητές αξίες (unbounded multiple knapsack problem with variable profits). Το πρόβλημα αυτό επιλύεται με διαδοχικές εφαρμογές του απλού προβλήματος του σακιδίου.

Προχωρώντας σε προσεγγίσεις που περιλαμβάνουν το σύνολο των κόμβων στη διαδικασία της προσωρινής αποθήκευσης, το πρόβλημα μοντελοποιείται και σε αυτήν τη περίπτωση ως ένα πρόβλημα μεγιστοποίησης του ποσοστού ευστοχίας στη μνήμη (cache hit ratio). Για την επίλυσή του προτείνεται ο PG-CAUSE, με ψευδοκώδικα που δίνεται στον Αλγόριθμο 13. Όπως και στις δυο προηγούμενες προσεγγίσεις, ο Σταθμός Βάσης είναι ο πρώτος που γεμίζει τη μνήμη του με τα δημοφιλέστερα αντικείμενα και μετά, κάθε χρήστης, σε φθίνουσα σειρά κεντρικότητας κόμβου επιλύοντας και εδώ ένα πρόβλημα σακιδίου. Τέλος, αναπτύχθηκε, ο RG-CAUSE, μια μέθοδος διαδραστικής αποθήκευσης (reactive caching). Σε αυτήν, αντιθέτως με όλες τις προηγούμενες μεθοδολογίες που προκαταβολικά (proactive caching) αποθήκευαν δεδομένα στις συσκευές, αυτές ξεκινάνε άδειες και αποθηκεύουν δεδομένα τα οποία οι χρήστες τους ζητάνε. Η σημαντική διαφορά έγκειται ότι στις προηγούμενες προσεγγίσεις η αποθήκευση γίνονταν με κριτήριο όχι μόνο τον κόμβο που θα αποθήκευε δεδομένα, αλλά και τη γειτονιά του. Σε περίπτωση που η μνήμη έχει γεμίσει και ο χρήστης ζητάει ένα νέο αντικείμενο, αν αυτό που ζήτησε είναι πιο κοντά στις προτιμήσεις του από κάποια που έχει αποθηκεύσει, τότε αφαιρεί τον ελάχιστο απαιτούμενο αριθμό αντικειμένων που χρειάζεται για να το προσθέσει στη μνήμη του. Διαφέρει δηλαδή από τις προηγούμενες προσεγγίσεις και στη δυνατότητα αναδιοργάνωσης της μνήμης.

Οι μέθοδοι αυτές εξετάστηκαν σε συνθετικά σύνολα δεδομένων και ανέδειξαν τα πλεονεκτήματα της χρήσης εκτός από ειδικού εξοπλισμού (π.χ., Σταθμοί Βάσης) και των συσκευών των χρηστών. Τα αποτελέσματα σε όρους ποσοστού επιτυχίας αναζήτησης δεδομένων συνηγορούν υπέρ του συμπεράσματος αυτού. Η Εικόνα 5.8 όπου παρουσιάζονται τα μεγάλα ποσοστά ευστοχίας για τους δυο αλγορίθμους PG-CAUSE και RG-CAUSE κάνει φανερά τα πλεονεκτήματα της χρήσης ενός μικρού ποσοστού μνήμης από όλους τους κόμβους.

Στη συνέχεια, εξετάζεται το πρόβλημα της μεγιστοποίησης της ποιότητας της εμπειρίας (QoE) κάθε χρήστη με χρήση προσωρινής αποθήκευσης στα άκρα του δικτύου και παροχής συστάσεων. Στο μοντέλο που θεωρείται, σε αντιστοιχία με όσα προτείνονται στη βιβλιογραφία, οι χρήστες κινούνται εντός του χώρου και συναντιούνται μεταξύ τους με ρυθμό που καθορίζεται από μια κατανομή Poisson. Κατά τη διάρκεια ορισμένων από τις συναντήσεις τους, είναι εφικτή η ανταλλαγή δεδομένων. Η αναμενόμενη αναμονή μεταξύ δυο επιτυχημένων συναντήσεων δίνεται από την Εξίσωση 5.10. Ακόμα, μόνο ορισμένοι από τους χρήστες μπορούν να αποθηκεύσουν δεδομένα στις συσκευές τους, οι οποίοι θεωρείται ότι απολαμβάνουν συγκεκριμένα οφέλη από την παραχώρηση χώρου της μνήμης τους (π.χ., μειωμένες χρεώσεις). Για να επιλεγούν αυτοί οι χρήστες αναπτύσσονται δυο μεθοδολογίες. Η μια εστιάζει στον αριθμό χρηστών που συναντάει κάθε υποψήφιος κόμβος για να οριστεί ως κόμβος αποθήκευσης (για χάρη συνεκτικότητας, αναφέρεται και εδώ ως Clusterhead, CH). Η δεύτερη εστιάζει στην εύρεση των CHs με τρόπο που να καλύπτουν το σύνολο (ή έστω το μεγαλύτερο ποσοστό) των χρηστών. Δηλαδή να υπάρχει τουλάχιστον μια συνάντηση μεταξύ κάθε χρήστη και τουλάχιστον ενός CH.

Το QoE κάθε χρήστη ορίζεται σαν συνάρτηση της συνάφειας των συστάσεων (QoR) που λαμβάνει και του χρόνου στον οποίο θα λάβει τα αντικείμενα τα οποία αιτήθηκε (ποιότητα υπηρεσίας, Quality of Service, QoS). Έτσι η ολική ποιότητα της εμπειρίας υπολογίζεται από την Εξίσωση 5.14. Το εξεταζόμενο πρόβλημα μοντελοποιείται ως ένα πρόβλημα μεγιστοποίησης της ποιότητας εμπειρίας για όλους τους χρήστες και δίνεται στη Σχέση 5.17 με τους κατάλληλους περιορισμούς. Ένα τέτοιο πρόβλημα όμως έχει αποδειχθεί ότι ανήκει στην κλάση πολυπλοκότητας NP-hard, οπότε στα πλαίσια της διατριβής παρουσιάζεται άπληστος αλγόριθμος για την επίλυσή του.

Πιο συγκεκριμένα, αφού οριστούν οι κόμβοι που θα παίζουν τον ρόλο του CH με χρήση μιας εκ των δυο προτεινόμενων μεθοδολογιών, κάθε κόμβος εντοπίζει αυτούς με τους οποίους συναντιέται με τη μικρότερη καθυστέρηση. Βασισμένοι σε αυτούς τους χρήστες, οι CHs γεμίζουν τις μνήμες τους επιλύοντας και εδώ το πρόβλημα του σακιδίου. Κάθε κόμβος εξετάζεται και γεμίζει τη μνήμη του λαμβάνοντας υπόψιν τη μέση καθυστέρηση που θα έχει για να παραδώσει τα αντικείμενα στη γειτονιά του. Με αυτόν τον τρόπο επιλέγονται πρώτα να γεμίσουν τις μνήμες τους οι κόμβοι εκείνοι που θα συναντήσουν ταχύτερα άλλους κόμβους. Έπειτα, μηδενίζεται, για κάθε χρήστη στη γειτονιά του CH, η αξία των αντικειμένων που συμπεριλήφθηκαν στα αποθηκευμένα αντικείμενα της μνήμης του, έτσι ώστε να μην επηρεάσουν την αποθήκευση στους επόμενους CHs. Στη συνέχεια για κάθε κόμβο, υπολογίζονται τα αντικείμενα που μπορούν αν του προταθούν και να παραδοθούν από τους κόμβους τους οποίους συναντά συχνότερα. Ένα αντικείμενο εισάγεται στη λίστα συστάσεων του χρήστη αν και μόνο αν η συνάφεια του είναι εντός ενός συγκεκριμένου, καθορισμένου από τον πάροχο, εύρους από τα πιο αγαπημένα αντικείμενα. Ο Αλγόριθμος 8 παρουσιάζει τον ψευδοκώδικα της μεθοδολογίας αυτής.

Ο αλγόριθμος αυτός, καταφέρνει να παρέχει υψηλό QoR σε αποδεκτό QoS καθιστώντας τον ως μια καλή εναλλακτική σε δημοσιευμένο προσεγγιστικό αλγόριθμο εξαντλητικής αναζήτησης ο οποίος πετυχαίνει υψηλότερο, αλλά συγκρίσιμο QoE, όπως μπορεί να φανεί και από το διάγραμμα της Εικόνας 5.14, απαιτεί όμως τάξεις μεγέθους υψηλότερο χρόνο εκτέλεσης κάτι που φαίνεται στα αποτελέσματα που καταγράφονται στον Πίνακα 5.1 .

Κεφάλαιο 6

Το Κεφάλαιο συνοψίζει το σύνολο της ερευνητικής δουλειάς που παρουσιάζεται στο πλαίσιο της διατριβής, εκθέτοντας συνοπτικά τη σπουδαιότητα των ερευνητικών προβλημάτων και συνοψίζοντας τα συμπεράσματα από τις προτεινόμενες λύσεις που αναπτύχθηκαν. Στη συνέχεια, παρουσιάζονται πιθανές μελλοντικές ερευνητικές κατευθύνσεις που μπορούν να έχουν ως αφετηρία τα συμπεράσματα αλλά και τις μεθόδους που παρουσιάστηκαν σε αυτή τη διατριβή.

Bibliography

[1] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[2] "Global Smart City Market Size 2019-2025." `https://www.statista.com/statistics/1111626/worldwide-smart-city-market-revenue/`, 2019. [Online; accessed 08-February-2021].

[3] Y. He, F. R. Yu, N. Zhao, V. C. Leung, and H. Yin, "Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 31–37, 2017.

[4] E. Baştuğ, M. Bennis, M. Kountouris, and M. Debbah, "Cache-enabled small cell networks: Modeling and tradeoffs," *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, no. 1, pp. 1–11, 2015.

[5] D. Tsigkari and T. Spyropoulos, "User-centric optimization of caching and recommendations in edge cache networks," in *2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*, pp. 244–253, IEEE, 2020.

[6] V. Karyotis, E. Stai, and S. Papavassiliou, *Evolutionary dynamics of complex communications networks.* CRC Press, 2013.

[7] R. C. Prim, "Shortest connection networks and some generalizations," *The Bell System Technical Journal*, vol. 36, no. 6, pp. 1389–1401, 1957.

[8] J. B. Kruskal, “On the shortest spanning subtree of a graph and the traveling salesman problem,” *Proceedings of the American Mathematical society*, vol. 7, no. 1, pp. 48–50, 1956.

[9] A. Landherr, B. Friedl, and J. Heidemann, “A critical review of centrality measures in social networks,” *Business & Information Systems Engineering*, vol. 2, no. 6, pp. 371–385, 2010.

[10] L. Katz, “A new status index derived from sociometric analysis,” *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[11] L. Page, “The pagerank citation ranking: Bringing order to the web. technical report,” *Stanford Digital Library Technologies Project, 1998*, 1998.

[12] U. Brandes, “On variants of shortest-path betweenness centrality and their generic computation,” *Social Networks*, vol. 30, no. 2, pp. 136–145, 2008.

[13] M. Girvan and M. E. Newman, “Community structure in social and biological networks,” *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[14] E. N. Gilbert, “Random graphs,” *The Annals of Mathematical Statistics*, vol. 30, no. 4, pp. 1141–1144, 1959.

[15] P. Erdős and A. Rényi, “On the evolution of random graphs,” *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960.

[16] I. Glauche, W. Krause, R. Sollacher, and M. Greiner, “Continuum percolation of wireless ad hoc communication networks,” *Physica A: Statistical Mechanics and its Applications*, vol. 325, no. 3-4, pp. 577–600, 2003.

[17] J. Dall and M. Christensen, “Random geometric graphs,” *Physical review E*, vol. 66, no. 1, p. 016121, 2002.

[18] D. J. Watts, *Small worlds: the dynamics of networks between order and randomness.* Princeton university press, 2004.

[19] A. D. Broido and A. Clauset, “Scale-free networks are rare,” *Nature communications*, vol. 10, no. 1, pp. 1–10, 2019.

[20] R. Albert and A.-L. Barabási, “Statistical mechanics of complex networks,” *Reviews of modern physics*, vol. 74, no. 1, p. 47, 2002.

[21] O. Narayan and I. Saniee, “Large-scale curvature of networks,” *Physical Review E*, vol. 84, no. 6, p. 066108, 2011.

[22] D. Krioukov, F. Papadopoulos, A. Vahdat, and M. Boguná, “Curvature and temperature of complex networks,” *Physical Review E*, vol. 80, no. 3, p. 035101, 2009.

[23] P. Chunaev, “Community detection in node-attributed social networks: a survey,” *Computer Science Review*, vol. 37, p. 100286, 2020.

[24] S. Fortunato, “Community detection in graphs,” *Physics reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[25] M. Rosvall, J.-C. Delvenne, M. T. Schaub, and R. Lambiotte, “Different approaches to community detection,” *Advances in network clustering and blockmodeling*, pp. 105–119, 2019.

[26] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, p. P10008, 2008.

[27] A. Clauset, M. E. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical review E*, vol. 70, no. 6, p. 066111, 2004.

[28] S. vanDongen, “A cluster algorithm for graphs,” *Information Systems [INS]*, vol. R 0010, 2000.

[29] J. Xie, S. Kelley, and B. K. Szymanski, “Overlapping community detection in networks: The state-of-the-art and comparative study,” *Acm computing surveys (csur)*, vol. 45, no. 4, pp. 1–35, 2013.

[30] M. T. Schaub, J.-C. Delvenne, M. Rosvall, and R. Lambiotte, “The many facets of community detection in complex networks,” *Applied network science*, vol. 2, no. 1, p. 4, 2017.

[31] P. Ponveni and J. Visumathi, “A review on community detection algorithms and evaluation measures in social networks,” in *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 1, pp. 1892–1899, IEEE, 2021.

[32] S. Moon, J.-G. Lee, M. Kang, M. Choy, and J.-w. Lee, “Parallel community detection on large graphs with mapreduce and graphchi,” *Data & Knowledge Engineering*, vol. 104, pp. 17–31, 2016.

[33] A. Kyrola, G. Blelloch, and C. Guestrin, “Graphchi: Large-scale graph computation on just a {PC},” in *Presented as part of the 10th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 12)*, pp. 31–46, 2012.

[34] S. Fortunato, V. Latora, and M. Marchiori, “Method to find community structures based on information centrality,” *Physical review E*, vol. 70, no. 5, p. 056104, 2004.

[35] M. Arasteh and S. Alizadeh, “A fast divisive community detection algorithm based on edge degree betweenness centrality,” *Applied Intelligence*, vol. 49, no. 2, pp. 689–702, 2019.

[36] Q. Ji, D. Li, and Z. Jin, “Divisive algorithm based on node clustering coefficient for community detection,” *IEEE Access*, vol. 8, pp. 142337–142347, 2020.

[37] X. Ding, J. Zhang, J. Yang, and Y. Shen, “An autonomous divisive algorithm for community detection based on weak link and link-break strategy,” *Mathematical Problems in Engineering*, vol. 2018, 2018.

[38] U. Von Luxburg, “A tutorial on spectral clustering,” *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[39] S. Lloyd, “Least square quantization in pcm. bell telephone laboratories paper. published in journal much later: Lloyd, sp: Least squares quantization in pcm,” *IEEE Trans. Inform. Theor.(1957/1982)*, vol. 18, 1957.

[40] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[41] A. Saade, F. Krzakala, and L. Zdeborová, “Spectral clustering of graphs with the bethe hessian,” in *Advances in Neural Information Processing Systems*, pp. 406–414, 2014.

[42] U. Von Luxburg, M. Belkin, and O. Bousquet, “Consistency of spectral clustering,” *The Annals of Statistics*, pp. 555–586, 2008.

[43] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner, “Maximizing modularity is hard,” *arXiv preprint physics/0608255*, 2006.

[44] M. E. J. Newman, “Fast algorithm for detecting community structure in networks,” *Phys. Rev. E*, vol. 69, p. 066133, Jun 2004.

[45] M. Cordeiro, R. P. Sarmento, and J. Gama, “Dynamic community detection in evolving networks using locality modularity optimization,” *Social Network Analysis and Mining*, vol. 6, no. 1, p. 15, 2016.

[46] M. Seifikar, S. Farzi, and M. Barati, “C-blondel: An efficient louvain-based dynamic community detection algorithm,” *IEEE Transactions on Computational Social Systems*, vol. 7, no. 2, pp. 308–318, 2020.

[47] A. Lancichinetti and S. Fortunato, “Limits of modularity maximization in community detection,” *Physical review E*, vol. 84, no. 6, p. 066122, 2011.

[48] S. Fortunato and M. Barthelemy, “Resolution limit in community detection,” *Proceedings of the national academy of sciences*, vol. 104, no. 1, pp. 36–41, 2007.

[49] X. Lu, K. Kuzmin, M. Chen, and B. K. Szymanski, “Adaptive modularity maximization via edge weighting scheme,” *Information Sciences*, vol. 424, pp. 55–68, 2018.

[50] R. Guimera, M. Sales-Pardo, and L. A. N. Amaral, “Modularity from fluctuations in random graphs and complex networks,” *Physical Review E*, vol. 70, no. 2, p. 025101, 2004.

[51] A. J. Enright, S. Van Dongen, and C. A. Ouzounis, “An efficient algorithm for large-scale detection of protein families,” *Nucleic acids research*, vol. 30, no. 7, pp. 1575–1584, 2002.

[52] A. Bustamam, T. Siswantining, N. Febriyani, I. Novitasari, and R. Cahyaningrum, "Protein sequences clustering of herpes virus by using tribe markov clustering (tribe-mcl)," in *AIP Conference Proceedings*, vol. 1862.1, p. 030150, AIP Publishing LLC, 2017.

[53] A. Azad, G. A. Pavlopoulos, C. A. Ouzounis, N. C. Kyrpides, and A. Buluç, "Hipmcl: a high-performance parallel implementation of the markov clustering algorithm for large-scale networks," *Nucleic acids research*, vol. 46, no. 6, pp. e33–e33, 2018.

[54] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proceedings of the National Academy of Sciences*, vol. 105, no. 4, pp. 1118–1123, 2008.

[55] M. Rosvall, D. Axelsson, and C. T. Bergstrom, "The map equation," *The European Physical Journal Special Topics*, vol. 178, no. 1, pp. 13–23, 2009.

[56] B. Rozemberczki, R. Davies, R. Sarkar, and C. Sutton, "Gemsec: Graph embedding with self clustering," in *Proceedings of the 2019 IEEE/ACM international conference on advances in social networks analysis and mining*, pp. 65–72, 2019.

[57] F. Parés, D. G. Gasulla, A. Vilalta, J. Moreno, E. Ayguadé, J. Labarta, U. Cortés, and T. Suzumura, "Fluid communities: A competitive, scalable and diverse community detection algorithm," in *International Conference on Complex Networks and their Applications*, pp. 229–240, Springer, 2017.

[58] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical review E*, vol. 76, no. 3, p. 036106, 2007.

[59] J. Xie and B. K. Szymanski, "Community detection using a neighborhood strength driven label propagation algorithm," in *2011 IEEE Network Science Workshop*, pp. 188–195, IEEE, 2011.

[60] M. Asadi and F. Ghaderi, "Incremental community detection in social networks using label propagation method," in *2018 23rd Conference of Open Innovations Association (FRUCT)*, pp. 39–47, IEEE, 2018.

[61] X.-K. Zhang, J. Ren, C. Song, J. Jia, and Q. Zhang, "Label propagation algorithm for community detection based on node importance and label influence," *Physics Letters A*, vol. 381, no. 33, pp. 2691–2698, 2017.

[62] Q. Gui, R. Deng, P. Xue, and X. Cheng, "A community discovery algorithm based on boundary nodes and label propagation," *Pattern Recognition Letters*, vol. 109, pp. 103–109, 2018.

[63] M. Azaouzi and L. B. Romdhane, "An evidential influence-based label propagation algorithm for distributed community detection in social networks," *Procedia computer science*, vol. 112, pp. 407–416, 2017.

[64] S. Li, H. Lou, W. Jiang, and J. Tang, "Detecting community structure via synchronous label propagation," *Neurocomputing*, vol. 151, pp. 1063–1075, 2015.

[65] B. Huang, C. Wang, and B. Wang, "Nmlpa: Uncovering overlapping communities in attributed networks via a multi-label propagation approach," *Sensors*, vol. 19, no. 2, p. 260, 2019.

[66] J. Xie and B. K. Szymanski, "Labelrank: A stabilized label propagation algorithm for community detection in networks," in *2013 IEEE 2nd Network Science Workshop (NSW)*, pp. 138–143, IEEE, 2013.

[67] M. Tasgin and H. O. Bingol, "Community detection using preference networks," *Physica A: Statistical Mechanics and its Applications*, vol. 495, pp. 126–136, 2018.

[68] S. Ahajjam, M. El Haddad, and H. Badir, "Leadersrank: Towards a new approach for community detection in social networks," in *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*, pp. 1–8, 2015.

[69] S. Ahajjam, M. El Haddad, and H. Badir, "A new scalable leader-community detection approach for community detection in social networks," *Social Networks*, vol. 54, pp. 41–49, 2018.

[70] T. Ma, Q. Liu, J. Cao, Y. Tian, A. Al-Dhelaan, and M. Al-Rodhaan, "Lgiem: Global and local node influence based community detection," *Future Generation Computer Systems*, vol. 105, pp. 533–546, 2020.

[71] F. Papadopoulos, D. Krioukov, M. Boguñá, and A. Vahdat, "Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces," in *2010 Proceedings IEEE INFOCOM*, pp. 1–9, IEEE, 2010.

[72] G. Alanis-Lobato, P. Mier, and M. A. Andrade-Navarro, "Efficient embedding of complex networks to hyperbolic space via their laplacian," *Scientific reports*, vol. 6, p. 30108, 2016.

[73] F. Papadopoulos, C. Psomas, and D. Krioukov, "Network mapping by replaying hyperbolic growth," *IEEE/ACM Transactions on Networking*, vol. 23, no. 1, pp. 198–211, 2014.

[74] G. García-Pérez, A. Allard, M. Á. Serrano, and M. Boguñá, "Mercator: uncovering faithful hyperbolic embeddings of complex networks," *New Journal of Physics*, vol. 21, no. 12, p. 123033, 2019.

[75] A. Cvetkovski and M. Crovella, "Hyperbolic embedding and routing for dynamic graphs," in *IEEE INFOCOM 2009*, pp. 1647–1655, IEEE, 2009.

[76] X. Zhao, A. Sala, H. Zheng, and B. Y. Zhao, "Efficient shortest paths on massive social graphs," in *7th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, pp. 77–86, IEEE, 2011.

[77] F. Papadopoulos, M. Kitsak, M. Á. Serrano, M. Boguná, and D. Krioukov, "Popularity versus similarity in growing networks," *Nature*, vol. 489, no. 7417, pp. 537–540, 2012.

[78] F. Papadopoulos, R. Aldecoa, and D. Krioukov, "Network geometry inference using common neighbors," *Physical Review E*, vol. 92, no. 2, p. 022807, 2015.

[79] X. Zhao, A. Sala, C. Wilson, H. Zheng, and B. Y. Zhao, "Orion: shortest path estimation for large social graphs," *networks*, vol. 1, p. 5, 2010.

[80] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[81] W. P. Thurston, *Three-dimensional geometry and topology.* Princeton university press, 1997.

[82] I. Robinson, J. Webber, and E. Eifrem, *Graph databases.* " O'Reilly Media, Inc.", 2013.

[83] J. Webber, "A programmatic introduction to neo4j," in *Proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity*, pp. 217–218, 2012.

[84] R. S. Zemel and M. Á. Carreira-Perpiñán, "Proximity graphs for clustering and manifold learning," in *Advances in neural information processing systems*, pp. 225–232, 2005.

[85] "Network Data." `http://www-personal.umich.edu/~mejn/netdata/`, 2013. [Online; accessed 08-February-2021].

[86] X. Cai, D. Huang, C.-D. Wang, and C.-K. Kwoh, "Spectral clustering by subspace randomization and graph fusion for high-dimensional data," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 330–342, Springer, 2020.

[87] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical review E*, vol. 78, no. 4, p. 046110, 2008.

[88] J. Shao, Z. Han, Q. Yang, and T. Zhou, "Community detection based on distance dynamics," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1075–1084, 2015.

[89] "Smart Santander." `https://www.smartsantander.eu/`, 2020. [Online; accessed 09-March-2021].

[90] "Fiesta-IOT-Federated Interoperable Semantic IoT Testbeds and Applications ." `http://fiesta-iot.eu/`. [Online; accessed 08-February-2021].

[91] C. Nikas, G. Kadilierakis, P. Fafalios, and Y. Tzitzikas, "Keyword search over rdf: Is a single perspective enough?," *Big Data and Cognitive Computing*, vol. 4, no. 3, p. 22, 2020.

[92] "Neo4j + vis.js = neovis.js. Graph visualizations in the browser with data from Neo4j ." `https://github.com/neo4j-contrib/neovis.js`. [Online; accessed 17-May-2021].

[93] "Laboratoire d'analyse et d'architecture des systèmes | LAAS-CNRS ." `https://www.laas.fr/public/`. [Online; accessed 17-May-2021].

[94] E. Bakshy, I. Rosenn, C. Marlow, and L. Adamic, "The role of social networks in information diffusion," in *Proceedings of the 21st international conference on World Wide Web*, pp. 519–528, 2012.

[95] Q. Huang, L. Wang, and Y. Yang, "Secure and privacy-preserving data sharing and collaboration in mobile healthcare social networks of smart cities," *Security and Communication Networks*, vol. 2017, 2017.

[96] N. Ravi, R. Manoranjani, K. Seshadri, *et al.*, "Leveraging social networks for smart cities: A case-study in mitigation of air pollution," in *International Conference on Intelligent Information Technologies*, pp. 179–193, Springer, 2017.

[97] V. Moustaka, Z. Theodosiou, A. Vakali, A. Kounoudes, and L. G. Anthopoulos, "Enhancing social networking in smart cities: Privacy and security borderlines," *Technological Forecasting and Social Change*, vol. 142, pp. 285–300, 2019.

[98] N. N. Petrović, V. Dimovski, J. Peterlin, M. Meško, and V. Roblek, "Data-driven solutions in smart cities: The case of covid-19 apps," in *Proceedings of the World Wide Web Conference*, pp. 1–9, 2021.

[99] H. Vahabi, I. Koutsopoulos, F. Gullo, and M. Halkidi, "Difrec: A social-diffusion-aware recommender system," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1481–1490, 2015.

[100] H. Mahyar, R. Hasheminezhad, E. Ghalebi, A. Nazemian, R. Grosu, A. Movaghar, and H. R. Rabiee, "Identifying central nodes for information flow in social networks using compressive sensing," *Social Network Analysis and Mining*, vol. 8, no. 1, pp. 1–24, 2018.

[101] E. Gudes and N. Voloch, "An information-flow control model for online social networks based on user-attribute credibility and connection-strength factors," in *International Symposium on Cyber Security Cryptography and Machine Learning*, pp. 55–67, Springer, 2018.

[102] Y. Pan, F. Cong, K. Chen, and Y. Yu, "Diffusion-aware personalized social update recommendation," in *Proceedings of the 7th ACM conference on Recommender systems*, pp. 69–76, 2013.

[103] M. Vitoropoulou, K. Tsitseklis, V. Karyotis, and S. Papavassiliou, "Cover: An information diffusion aware approach for efficient recommendations under user coverage constraints," *IEEE Transactions on Computational Social Systems*, 2021.

[104] S. Naseri, A. Bahrehmand, and C. Ding, "An improved collaborative recommendation system by integration of social tagging data," in *Recommendation and Search in Social Networks*, pp. 119–138, Springer, 2015.

[105] E. Van den Broeck, K. Poels, and M. Walrave, "An experimental study on the effect of ad placement, product involvement and motives on facebook ad avoidance," *Telematics and Informatics*, vol. 35, no. 2, pp. 470–479, 2018.

[106] X. Zhang, X. Ding, and L. Ma, "The influences of information overload and social overload on intention to switch in social media," *Behaviour & Information Technology*, pp. 1–14, 2020.

[107] C. Musto, P. Lops, M. de Gemmis, and G. Semeraro, "Semantics-aware recommender systems exploiting linked open data and graph-based features," *Knowledge-Based Systems*, vol. 136, pp. 1–14, 2017.

[108] P. Lops, D. Jannach, C. Musto, T. Bogers, and M. Koolen, "Trends in content-based recommendation," *User Modeling and User-Adapted Interaction*, vol. 29, no. 2, pp. 239–249, 2019.

[109] F. Vasile, E. Smirnova, and A. Conneau, "Meta-prod2vec: Product embeddings using side-information for recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 225–232, 2016.

[110] P. B. Thorat, R. Goudar, and S. Barve, "Survey on collaborative filtering, content-based filtering and hybrid recommendation system," *International Journal of Computer Applications*, vol. 110, no. 4, pp. 31–36, 2015.

[111] F. Zhang, V. E. Lee, R. Jin, S. Garg, K.-K. R. Choo, M. Maasberg, L. Dong, and C. Cheng, "Privacy-aware smart city: A case study in collaborative filtering recommender systems," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 145–159, 2019.

[112] G. Carullo, A. Castiglione, and A. De Santis, "Friendship recommendations in online social networks," in *2014 International Conference on Intelligent Networking and Collaborative Systems*, pp. 42–48, IEEE, 2014.

[113] R. Burke, "Knowledge-based recommender systems," *Encyclopedia of library and information systems*, vol. 69, no. Supplement 32, pp. 175–186, 2000.

[114] J. K. Tarus, Z. Niu, and G. Mustafa, "Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning," *Artificial intelligence review*, vol. 50, no. 1, pp. 21–48, 2018.

[115] H. Gao, J. Tang, X. Hu, and H. Liu, "Content-aware point of interest recommendation on location-based social networks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 29.1, 2015.

[116] D. Lian, Y. Wu, Y. Ge, X. Xie, and E. Chen, "Geography-aware sequential location recommendation," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2009–2019, 2020.

[117] S. Bouraga, I. Jureta, S. Faulkner, and C. Herssens, "Knowledge-based recommendation systems: a survey," *International Journal of Intelligent Information Technologies (IJIIT)*, vol. 10, no. 2, pp. 1–19, 2014.

[118] V. Karyotis, M. Vitoropoulou, N. Kalatzis, I. Roussaki, and S. Papavassiliou, "Efficient and socio-aware recommendation approaches for bigdata networked systems," *Big Data Recommender Systems: Algorithms, Architectures, Big Data, Security and Trust*, vol. 1, p. 41, 2019.

[119] K. Al Falahi, N. Mavridis, and Y. Atif, "Social networks and recommender systems: a world of current and future synergies," in *Computational Social Networks*, pp. 445–465, Springer, 2012.

[120] E. Negre and C. Rosenthal-Sabroux, "Recommendations to improve the smartness of a city," in *Smart City*, pp. 101–115, Springer, 2014.

[121] L. Quijano-Sánchez, I. Cantador, M. E. Cortés-Cediel, and O. Gil, "Recommender systems for smart cities," *Information systems*, vol. 92, p. 101545, 2020.

[122] M. Vitoropoulou, V. Karyotis, and S. Papavassiliou, "Sensing and monitoring of information diffusion in complex online social networks," *Peer-to-Peer Networking and Applications*, vol. 12, no. 3, pp. 604–619, 2019.

[123] M. Gan and R. Jiang, "Flower: Fusing global and local associations towards personalized social recommendation," *Future Generation Computer Systems*, vol. 78, pp. 462–473, 2018.

[124] D. Margaris, C. Vassilakis, and P. Georgiadis, "Query personalization using social network information and collaborative filtering techniques," *Future Generation Computer Systems*, vol. 78, pp. 440–450, 2018.

[125] M. G. Rodriguez, K. Gummadi, and B. Schoelkopf, "Quantifying information overload in social media and its impact on social contagions," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 8.1, 2014.

[126] G. Gigerenzer, *Gut feelings: The intelligence of the unconscious.* Penguin, 2007.

[127] K. R. Apt, E. Markakis, and S. Simon, "Paradoxes in social networks with multiple products," *Synthese*, vol. 193, no. 3, pp. 663–687, 2016.

[128] V. Zverovich and E. Avineri, "Braess' paradox in a generalised traffic network," *Journal of Advanced Transportation*, vol. 49, no. 1, pp. 114–138, 2015.

[129] F. de Nijs, G. Theocharous, N. Vlassis, M. M. de Weerdt, and M. T. J. Spaan, "Capacity-aware sequential recommendations," in *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018)*, IFAAMAS, 2018.

[130] X. Song, B. L. Tseng, C.-Y. Lin, and M.-T. Sun, "Personalized recommendation driven by information flow," in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 509–516, 2006.

[131] F. Ullah and S. Lee, "Social content recommendation based on spatial-temporal aware diffusion modeling in social networks," *Symmetry*, vol. 8, no. 9, p. 89, 2016.

[132] A. Guille, H. Hacid, C. Favre, and D. A. Zighed, "Information diffusion in online social networks: A survey," *ACM Sigmod Record*, vol. 42, no. 2, pp. 17–28, 2013.

[133] M. Granovetter, "Threshold models of collective behavior," *American journal of sociology*, vol. 83, no. 6, pp. 1420–1443, 1978.

[134] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 4, pp. 1–37, 2012.

[135] G. Liu, Y. Huang, F. Wang, J. Liu, and Q. Wang, "5g features from operation perspective and fundamental performance validation by field trial," *China Communications*, vol. 15, no. 11, pp. 33–50, 2018.

[136] L. Guevara and F. Auat Cheein, "The role of 5g technologies: Challenges in smart cities and intelligent transportation systems," *Sustainability*, vol. 12, no. 16, p. 6469, 2020.

[137] "IoT Growth Demands Rethink of Long-Term Storage Strategies, says IDC." `https://www.idc.com/getdoc.jsp?containerId=prAP46737220/`, 2020. [Online; accessed 09-March-2021].

[138] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *Ieee Access*, vol. 5, pp. 6757–6779, 2017.

[139] L. U. Khan, I. Yaqoob, N. H. Tran, S. A. Kazmi, T. N. Dang, and C. S. Hong, "Edge computing enabled smart cities: A comprehensive survey," *IEEE Internet of Things Journal*, 2020.

[140] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[141] S. Soleimani and X. Tao, "Cooperative crossing cache placement in cache-enabled device to device-aided cellular networks," *Applied Sciences*, vol. 8, no. 9, p. 1578, 2018.

[142] Z. Chen, N. Pappas, and M. Kountouris, "Probabilistic caching in wireless d2d networks: Cache hit optimal versus throughput optimal," *IEEE Communications Letters*, vol. 21, no. 3, pp. 584–587, 2016.

[143] Y. Xu, "On the performance of device-to-device communications with delay constraint," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 11, pp. 9330–9344, 2016.

[144] M. D. Amentie, M. Sheng, J. Song, and J. Liu, "Minimum delay guaranteed cooperative device-to-device caching in 5g wireless networks," in *2016 8th International Conference on Wireless Communications & Signal Processing (WCSP)*, pp. 1–5, IEEE, 2016.

[145] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Transactions on Management Information Systems (TMIS)*, vol. 6, no. 4, pp. 1–19, 2015.

[146] R. Zhou, S. Khemmarat, and L. Gao, "The impact of youtube recommendation system on video views," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pp. 404–410, 2010.

[147] L. E. Chatzieleftheriou, M. Karaliopoulos, and I. Koutsopoulos, "Jointly optimizing content caching and recommendations in small cell networks," *IEEE Transactions on Mobile Computing*, vol. 18, no. 1, pp. 125–138, 2018.

[148] K. Poularakis, G. Iosifidis, V. Sourlas, and L. Tassiulas, "Exploiting caching and multicast for 5g wireless networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 4, pp. 2995–3007, 2016.

[149] Z. Chang, L. Lei, Z. Zhou, S. Mao, and T. Ristaniemi, "Learn to cache: Machine learning for network edge caching in the big data era," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 28–35, 2018.

[150] R. Amer, M. M. Butt, M. Bennis, and N. Marchetti, "Inter-cluster cooperation for wireless d2d caching networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 9, pp. 6108–6121, 2018.

[151] D. Liu and C. Yang, "Caching policy toward maximal success probability and area spectral efficiency of cache-enabled hetnets," *IEEE Transactions on Communications*, vol. 65, no. 6, pp. 2699–2714, 2017.

[152] J. Wen, K. Huang, S. Yang, and V. O. Li, "Cache-enabled heterogeneous cellular networks: Optimal tier-level content placement," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5939–5952, 2017.

[153] X. Hu, T. H. Chu, V. C. Leung, E. C.-H. Ngai, P. Kruchten, and H. C. Chan, "A survey on mobile social networks: Applications, platforms, system architectures, and future research directions," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1557–1581, 2014.

[154] G. Kakkavas, K. Tsitseklis, V. Karyotis, and S. Papavassiliou, "A software defined radio cross-layer resource allocation approach for cognitive radio networks: From theory to practice," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 2, pp. 740–755, 2020.

[155] Z. Chen and M. Kountouris, "D2d caching vs. small cell caching: Where to cache content in a wireless network?," in *2016 IEEE 17th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–6, IEEE, 2016.

[156] T. Zhang, X. Fang, Y. Liu, and A. Nallanathan, "Content-centric mobile edge caching," *IEEE Access*, vol. 8, pp. 11722–11731, 2019.

[157] M. Sheng, C. Xu, J. Liu, J. Song, X. Ma, and J. Li, "Enhancement for content delivery with proximity communications in caching enabled wireless networks: Architecture and challenges," *IEEE Communications Magazine*, vol. 54, no. 8, pp. 70–76, 2016.

[158] R. Wang, J. Zhang, S. Song, and K. B. Letaief, "Mobility-aware caching in d2d networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 5001–5015, 2017.

[159] K. Poularakis and L. Tassiulas, "Exploiting user mobility for wireless content delivery," in *2013 IEEE International Symposium on Information Theory*, pp. 1017–1021, IEEE, 2013.

[160] M. Chen, Y. Hao, L. Hu, K. Huang, and V. K. Lau, "Green and mobility-aware caching in 5g networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 12, pp. 8347–8361, 2017.

[161] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Cooperative content caching in 5g networks with mobile edge computing," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 80–87, 2018.

[162] D. K. Krishnappa, M. Zink, C. Griwodz, and P. Halvorsen, "Cache-centric video recommendation: an approach to improve the efficiency of youtube caches," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 11, no. 4, pp. 1–20, 2015.

[163] S. Kastanakis, P. Sermpezis, V. Kotronis, and X. Dimitropoulos, "Cabaret: Leveraging recommendation systems for mobile edge caching," in *Proceedings of the 2018 Workshop on Mobile Edge Communications*, pp. 19–24, 2018.

[164] L. E. Chatzieleftheriou, M. Karaliopoulos, and I. Koutsopoulos, "Caching-aware recommendations: Nudging user preferences towards better caching performance," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pp. 1–9, IEEE, 2017.

[165] J. Vegelius, S. Janson, and F. Johansson, "Measures of similarity between distributions," *Quality and Quantity*, vol. 20, no. 4, pp. 437–441, 1986.

[166] S. Traverso, M. Ahmed, M. Garetto, P. Giaccone, E. Leonardi, and S. Niccolini, "Temporal locality in today's content caching: why it matters and how to model it," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 5, pp. 5–12, 2013.

[167] J.-Y. Le Boudec and M. Vojnovic, "Perfect simulation and stationarity of a class of mobility models," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 4, pp. 2743–2754, IEEE, 2005.

[168] H. Kellerer, U. Pferschy, and D. Pisinger, “Multidimensional knapsack problems,” in *Knapsack problems*, pp. 235–283, Springer, 2004.

[169] V. Vazirani, “Approximation algorithms springer-verlag,” *New York*, 2001.

[170] G. Huang, B. Zhang, Z. Yao, and C. Li, “Quality-aware video streaming for green cellular networks with hybrid energy sources,” *IEEE Internet of Things Journal*, vol. 8, no. 10, pp. 8543–8556, 2020.

[171] K. Poularakis, G. Iosifidis, A. Argyriou, and L. Tassiulas, “Video delivery over heterogeneous cellular networks: Optimizing cost and performance,” in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, pp. 1078–1086, IEEE, 2014.

[172] P. Sermpezis and T. Spyropoulos, “Offloading on the edge: Performance and cost analysis of local data storage and offloading in hetnets,” in *2017 13th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, pp. 49–56, IEEE, 2017.

Publications

Publications in Peer Reviewed Journals

- Margarita Vitoropoulou, Konstantinos Tsitseklis, Vasileios Karyotis, and Symeon Papavassiliou, "CoveR: An Information Diffusion Aware Approach for Efficient Recommendations under User Coverage Constraints," in *IEEE Transactions on Computational Social Systems*, doi: 10.1109/TCSS.2021.3067711, 2021.

- Konstantinos Tsitseklis, Margarita Vitoropoulou, Vasileios Karyotis, and Symeon Papavassiliou, "Socio-aware Recommendations under Complex User Constraints," in *IEEE Transactions on Computational Social Systems*, vol. 8, no. 2, pp. 377-387, April 2021.

- Konstantinos Tsitseklis, Maria Krommyda, Vasileios Karyotis, Verena Kantere, and Symeon Papavassiliou, "Scalable Community Detection for Complex Data Graphs via Hyperbolic Network Embedding and Graph Databases," In *IEEE Transactions on Network Science and Engineering*, IEEE, 2020.

- Grigorios Kakkavas, Konstantinos Tsitseklis, Vasileios Karyotis, and Symeon Papavassiliou, "A Software Defined Radio Cross-Layer Resource Allocation Approach for Cognitive Radio Networks: From Theory to Practice," In *IEEE Transactions on Cognitive Communications and Networking*, IEEE, Vol 6.2, pp. 740-755, 2020.

- Adamantia Stamou, Grigorios Kakkavas, Konstantinos Tsitseklis, Vasileios Karyotis, and Symeon Papavassiliou, "Autonomic network management and cross-layer optimization in software defined radio environments," In *Future Internet*, Vol. 11.2, pp. 37-54, 2019.

- Vasileios Karyotis, Konstantinos Tsitseklis, Konstantinos Sotiropoulos, and Symeon Papavassiliou, “Big data clustering via community detection and hyperbolic network embedding in IoT applications,” In *Sensors*, Vol. 18.4, pp. 1205-1225, 2018.

Publications in Peer Reviewed International Conferences

- Maria Krommyda, Konstantinos Tsitseklis, Vassiliki Kantere, Vasileios Karyotis, and Symeon Papavassiliou, “Visualizing and Exploring Big Datasets based on Semantic Community Detection,” In *24th International Conference on Extending Database Technology (EDBT)*, Mar. 22-26, 2021
- Margarita Vitoropoulou, Konstantinos Tsitseklis, Angelos Karakoulias, Vasileios Karyotis, and Symeon Papavassiliou, “CAUSE: Caching Aided by USer Equipment,” In *2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, IEEE, pp. 713-721, Rhodes, Nov. 2-6, 2020.
- Konstantinos Tsitseklis, Grigorios Kakkavas, Vasileios Karyotis, and Symeon Papavassiliou, “A Realistic Evaluation of MRF-based Resource Allocation for SDR Cognitive Radio Networks,” In *2019 IEEE 44th Conference on Local Computer Networks (LCN)*, IEEE, pp. 185-192, Osnabruck, Oct. 14-17, 2019.
- Vasileios Karyotis, Konstantinos Tsitseklis, Konstantinos Sotiropoulos, and Symeon Papavassiliou, “Enhancing Community Detection for Big Sensor Data Clustering via Hyperbolic Network Embedding,” In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, IEEE, pp. 266-271, March 19-23, 2018.