INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Community Discovery in Mobile Games

Mário Lança Campos Salgado

Master in Computer Engineering

Supervisor:
Prof. Dr. Luís Miguel Martins Nunes, Associate Professor
ISCTE - Instituto Universitário de Lisboa

Co-Supervisor:
Profª. Drª. Ana Maria Carvalho de Almeida, Associate Professor
ISCTE - Instituto Universitário de Lisboa

October, 2020

”Creativity is seeing what everyone else has seen, and thinking what no one else has though.”
Albert Einstein

Acknowledgment

Throughout the writing of this dissertation, I got exceptional support and assistance.

I would first like to thank my parents Luísa and José for their thoughtful advice and tender ear. You are always there for me. My girlfriend, who was of great support in deliberating over obstacles and decisions, as well as providing cheering distraction to rest my mind outside of my research.

Secondly, an acknowledgement my supervisors whose expertise was invaluable in the formulating of the research topic and methodology in particular. This thesis was written during the pandemic crisis of Covid-19, and I could not wish for better remote support than the one provided by my supervisors.

I would like to thank my friend and boss Miguel Costa, and my company Miniclip, for supporting me with time, resources and data to develop this thesis. Nothing would be possible without you.

Lastly, a heartfelt thank you to ISCTE, an institution of excellence that helped me improve as individual.

Resumo

Os jogos tornaram-se uma das principais plataformas de entretenimento dos dias atuais, representando mais de 2,5 biliões de jogadores em todo o mundo que, apenas em 2019, gastaram cerca de 150 biliões de dólares em jogos.

De todos os jogos, os jogos para dispositivos móveis tornaram-se particularmente populares, com jogos como o Candy Crush a atingir 258 milhões de utilizadores ativos por mês e Call of Duty Mobile a ser descarregado 100 milhões de vezes na primeira semana de lançamento.

Com a crescente relevância dos jogos para dispositivos móveis, há uma necessidade de entender como os jogadores interagem e como as comunidades de jogos estão estruturadas. As comunidades nos jogos podem ser consideradas redes sociais de indivíduos, neste caso, jogadores, que interagem entre si a diversos níveis.

Nesta tese, tentamos entender como as comunidades estão estruturadas e evoluem através de uma rede de interações presentes frequentemente nos jogos para dispositivos móveis.

Testámos a nossa metodologia com conjuntos de dados de um jogo real para dispositivos móveis com 20 milhões de utilizadores ativos diários, recolhidos durante 60 dias em 3 países diferentes.

A principal contribuição desta tese é uma análise de diferentes métodos de descoberta de comunidades com este conjunto de dados reais. Fornecemos também um sistema de recomendação de amizades baseado em comunidades, que pode ser executado neste tipo de redes e atingir até 79 % de precisão em comunidades em crescimento.

Keywords: Jogos para dispositivos móveis, Descoberta de comunidades, Ciclo de vida de comunidades, Sistema de recomendação

Abstract

Gaming has become one of the leading entertainment platforms of current days, representing more than 2.5 billion gamers around the world that, just in 2019, spent around 150 billion dollars on games.

Across gaming, mobile gaming has become quite popular, with games such as the Candy Crush franchise achieving 258 monthly active users and Call of Duty Mobile being downloaded 100 million times in the first week.

With the increasing relevance of mobile gaming, there is a necessity to understand how players interact, and gaming communities structure themselves. Communities in games can be considered as social networks of individuals, in this case, players, that interact with each other at diverse levels.

In this thesis we try to understand how communities are structured and evolve through a network of common mobile gaming interactions.

We test our methodology with datasets from a real mobile game with 20 daily active users collected across 60 days in 3 different countries.

The main contribution of this thesis is an analysis of different community discovery methods with this a real mobile game datasets. We also provide a community based friendship recommendation system that can run on top such networks and deliver up to 79% precision on growing communities.

Keywords: Mobile gaming, Community discovery, Community life-cycle, Recommendation System

Contents

List of Figures

List of Tables

Glossary

Apache Kafka: Apache Kafka is a community distributed event streaming platform. Initially conceived as a messaging queue, Kafka is based on an abstraction of a distributed commit log.. 18

data lake: A data lake is a repository of data stored in its natural or raw format, usually files.. 19, 20

load balancing: Load balancing refers to the process of distributing a set of tasks over a set of resources (computing units), with the aim of making their overall processing more efficient. Load balancing techniques can optimize the response time for each task, avoiding unevenly overloading compute nodes while other compute nodes are left idle.. 18

Acronyms

DAU: daily active users. 28, 35

ETL: extract, transform and load. 18

MAU: monthly active users. 1

NMI: Normalized Mutual Information. 15

REST: Representational State Transfer. 18, 19

CHAPTER 1

Introduction

Gaming has become one of the leading entertainment platforms of current days, representing more than 2.5 billion gamers around the world that, just in 2019, spent around 150 billion dollars on games. From this global market, 45% players play on mobile devices (smartphone and tablet) [1].

Mobile gaming has become quite popular, with games such as the Candy Crush franchise achieving 258 monthly active users (MAU) [2] and Call of Duty Mobile being downloaded 100 million times in the first week [3].

1.1. Framework

The social components of most mobile games are essential features in current digital gaming because social networking sites (Facebook, Twitter) are successfully integrated and used over many gaming platforms. Several user acquisition specialists identify communities as the core of user acquisition and user retention [4] [5], which are essential for the prosperity of the industry.

Gaming communities can be considered as social networks of individuals, in this case, players, that interact with each other at diverse levels [6].

1.2. Motivation

With the increasing relevance of mobile gaming, there is a necessity to understand how players interact, and gaming communities structure themselves. This knowledge can genuinely help the gaming industry boost engagement and understand the impact of new features on players interaction.

This thesis was developed in partnership with Miniclip. Miniclip is a mobile gaming company that is a global leader in the sports category, with around 200 million monthly active players all around the world. The company saw this thesis as an opportunity to explore communities in their biggest game. All the data provided by the company for this thesis was collected and used anonymously and will not be available publicly.

1.3. Objectives

This thesis' objective is to find the best methodology to discover communities in mobile games. To find such methodology, several questions arise:

- How can we develop a methodology that is easily reproducible across other mobile games?
- How can we evaluate performance and quality in a real use case without user feedback?
- How should we deal with temporal data?
- How does community evolution behave in a mobile scenario?

1.4. Contributions

The main contribution of this thesis is the comparison and test of different proven community discovery methodologies with real mobile gaming data. This also includes data that changes over time and has scale that can challenge most methodologies.

We also present a research hypothesis about how we can use community knowledge to predict user friendships.

1.5. Document Structure

This document is divided into 6 chapters that describe the whole thesis from its inception until the conclusion and possible future work.

- Introduction - Presents the framework, motivation, objectives and contributions of this thesis.
- Graphs and Dynamic Networks - Briefly describes graphs and dynamic networks used through the methodology.
- Literature Review - Presents communities, their characteristics, life cycles and instability. Describes each type of methodologies used for Dynamic Community Discovery, proposed optimisations and performance comparison.
- Methodology - Describes the methodology used on this thesis. From creating a dynamic network to performing community discovery and life cycle analysis. It also describes the research hypothesis around friendship recommendation.
- Results - Presents the results of different community discovery methodologies and analysis the research hypothesis.
- Conclusions and future research - This chapter concludes the presented work, presenting its potential benefit. It also extends the methodology and proposes new possibilities of further investigation.

CHAPTER 2

Graphs and Dynamic Networks

This chapter contains a brief overview of graphs and dynamic networks which, for the remainder of this thesis, will be used interchangeably.

2.1. Graphs

Symbol	Description
G	Graph representation of datasets
V	Set of nodes for graph G
E	Set of edges for graph G
N	Number of nodes, or $\left\|\mathrm{V}\right\|$
M	Number of edges, or $\left\|\mathrm{E}\right\|$
$e_{i,j}$	Edge between node i and node j
$w_{i,j}$	Weight on edge $e_{i,j}$
w_i	Weight of node i (sum of weights of incident edges)
A	0-1 Adjacency matrix of the unweighted graph
A_w	Real-value adjacency matrix of the weighted graph
$a_{i,j}$	Entry in matrix A
$\lambda 1$	Principal eigenvalue of unweighted graph
$\lambda 1, w$	Principal eigenvalue of weighted graph

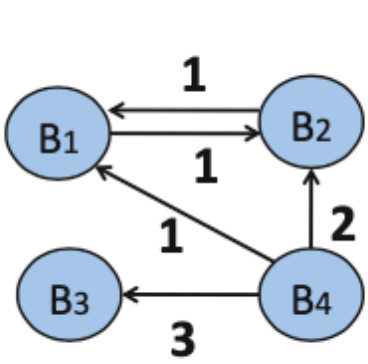
Table 1. List of notations from [7]

We can represent a network as a graph with the terms defined in table 1. A static, unweighted graph G consists of a set of nodes V and a set of edges such that $E \subseteq V \times V$. We represent the number of elements of V and E as N and E. A graph may be directed or undirected – for instance, a text message from one entity to another is represented as a directed edge to keep information about origin and destination of the message whereas a friendship between two persons is represented as undirected edge since it is a reciprocal relationship [7].

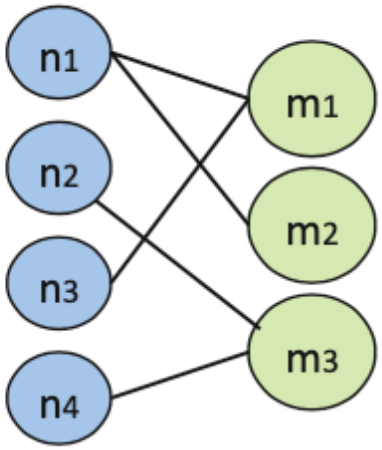
Graphs can also be weighted, where there may be many edges occurring between two nodes (e.g. different text messages) or specific edge weights (e.g. monetary amounts for transactions). In a weighted graph G, let $e_{i,j}$ be the edge between node i and node j. We will refer to these two nodes as the 'neighbouring nodes' or 'incident nodes' of edge $e_{i,j}$. Let $W_{i,j}$ be the weight on edge $e_{i,j}$. The total weight w_i of node i is defined as the sum of weights of all its edges.

We can also classify graphs as unipartite or multipartite [7]. A multipartite or k-partite graph is a graph whose nodes are partitioned into different types, and a unipartite graph only has one type of nodes. The graph representing a citation network were the only nodes are authors can be considered as unipartitie [8]. In the opposite, the movie-actor graph of IMDB [9] that consists of nodes that represent movies or actors is considered as a multipartite.

We can represent a graph visually or with an adjacency matrix A, where nodes are in rows and columns, and numbers in the matrix register the existence of edges. For unweighted graphs, all entries are 0 or 1; for weighted graphs, the adjacency matrix contains the values of the weights (figure 1).



	B1	B2	B3	B4
B1	0	1	0	0
B2	1	0	0	0
B3	0	0	0	0
B4	1	2	3	0



	n1	n2	n3	n4
m1	1	0	1	0
m2	1	0	0	0
m3	0	1	0	1

Figure 1. Illustrations of example graphs. On the left is a unipartite, directed, weighted graph and the corresponding adjacency matrix. On the right is an undirected, bipartite graph and the corresponding adjacency matrix. [7]

Another graph-related concept that we will need is the diameter. For a given (static) graph, its diameter is defined as the maximum distance between either two nodes, where distance is the minimum number of jumps (i.e., edges that must be crossed) on the path from one node to another, ignoring directionality.

Calculating graph diameter is $O(n^2)$, therefore, we choose to estimate graph's diameters by sampling nodes. For $s = \{1, 2, ..., S\}$, we pick two nodes at random and calculate the distance (using breadth-first search). We then record the 90 percentile value of distances, so we take the .9S largest recorded value. The distance operation is O(dk), where d is the graph diameter and k the maximum degree of any node on average this is a much smaller cost. Intuitively, the diameter represents how much of a "small world" the graph is– how quickly one can get from one "end" of the graph to another.

2.2. Dynamic Networks

A dynamic network is defined as a network of interactions or relationships, where the nodes consist of entities, and the edges consist of the relationships or interactions between these entities [10].

Also known as complex networks [11], dynamic networks have been subject of analysis by many researchers. The research extends from statistical analysis [7], community discovery [11] [12] [13], social tagging [14] and security anomaly detection [15].

With real-time data travelling through the internet and relationships between it evolving, we cannot consider networks only as static objects. As such we must consider dynamic networks as temporal networks that evolve with time. The static view of such a network is simply a snapshot of the network's state in a specific moment. Due to the online nature of web and, more recently, mobile applications, all the networks considered in this work are dynamic networks even when we work with snapshots at given moments [16].

2.2.1. Models

Time plays a crucial role in shaping network topologies. One of the most crucial issues to address in order to think about time-evolving networks is related to the mathematical formalism used to represent them.

In figure 2 from [12] the authors classify dynamic networks based on their model complexity and temporal impact, with four different conceptual solutions that progressively introduce the time dimension in the network modelling process.

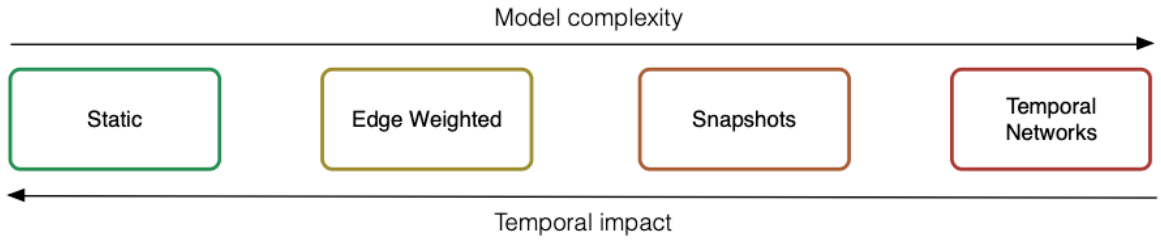


Figure 2. Network Representations [12]

On the one side, we can view the atemporal scenario (a single network capturing a static impression of a dynamic event). We can gradually introduce dynamics by using labels that weights nodes and edges, therefore catching an aggregate network. Following the same principle, weights model the number of occurrences of a given node/edge in an observation window, with this addition, we can make numerous time-dependent analyses that were impossible in the previous scenarios, such as tie strength estimation.

However, the strategy previously used suffers a severe limitation; it does not capture network dynamics. Taking this into consideration, many methods model dynamics with temporally ordered series of network snapshots. This mild modelling enables the tracking of perturbations occurring on the network topology. Nonetheless, with the improvement of the model definition,

the complexity of analysis also increases. When executing time-aware mining jobs on network snapshots, two concerns arise:

- How to keep track of multiple stages of the network life;
- How to harmonize the analytical results obtained in a snapshot with the outcome of the subsequent ones.

The constraining issue that affects dynamic network partition, as well as aggregation, is to define a threshold for temporal granularity. Since the identification of such a threshold is not easy, it is undoubtedly, context-dependent and deeply impactful for analytical results. Recent scientific research proposes to model network dynamics without any aggregation, keeping all temporal details. Such an approach usually details dynamic networks in their elementary bricks: temporally ordered, timestamped, interactions or relations. In the next subsection, we will describe such approach, specifically temporal networks modelling. Temporal networks allow for a full and fine-grained characterization of network dynamics.

Last but not least, we cannot forget that different problems, as well as accessible data, impose different modelling choices. Static networks, as well as weighted ones, are regularly used to identify stable patterns and to describe the current state of a network while snapshots and interactions are agents for the study of the increasing dynamic scenarios. With fine-grained temporal data, it is achievable to create all the other models and consequent aggregations.

In this thesis, we will focus on community detection methods that deal with temporal networks. As such for an in-depth overview of community detection methodology that focuses more on static scenarios refer to [11] [17].

2.2.1.1. Temporal Networks

Temporal networks [12] or Evolving Networks [18] represent the natural evolution over time of dynamic networks. Such phenomena is many times related with the volatility of real-time data.

Following the definition presented in [18] we can divide temporal networks into the following two categories:

- Slowly Evolving Networks - A network that evolves slowly over time and snapshot analysis can be used very effectively. In these situations, snapshots of the network at two distinct times t1 and t2 are used for analysis, and therefore offline analysis can be performed directly.
- Streaming Networks - A network that is created by transient interactions and due to that nature requires real-time analytical methods. Also titled as graph streams, this kind of networks are much more challenging due to the computation requirements needed for the analysis.

2.2.1.2. Network Snapshots

Regularly, network history partitions into a series of snapshots, each one of them matching either to the state of the network at a time t (relation network) or to the aggregation of observed interactions during a period (interaction network) [12].

2.2.2. Memory

A modelling decision that profoundly influences the analysis conducted on dynamic networks regards the system memory. When the analyzed data represents an interaction network (for instance, SMS or phone calls), not made of long-lasting relations, most methods require to convert it into a relation network.

Such conversion can generate snapshots or temporal networks. We need to take into consideration the two following scenarios:

- Perfect memory network
- Limited memory network.

In a perfect memory network, nodes and edges can only join the network, meaning that nodes/edges cannot disappear (e.g. a citation graph). In a limited memory network, nodes and edges vanish over time (e.g. social network relations). It is common to use the disappearance to model the decay of interactions in the network. We call this Time To Live (TTL), it is the artificially defined duration of an edge. There are several methodologies to find the correct TTL:

- Fixed Size Static Time Window: the TTL is equal for all network entities, and there is a finite set of possible periods whose start and end dates are defined in advance. This strategy produces network snapshots.
- Fixed Size Sliding Time Window: the TTL is equal for all network entities, but it starts independently for each one of them at the moment of their first appearance (i.e., an email graph in which each email is considered to have the same social persistence starting from the time it is made);
- Dynamic Size Time Window: the TTL is equal for all the network entities at a given time but depends on the current network status (i.e., a system for which it is interesting to capture bursts of interactions having different frequency);
- Global/Local Decay Function: the TTL of each node/edge is defined independently, usually as a function of the global/local network activity (i.e., fitting the inter-arrival time distribution of nodes or edges for the whole network as well as for the individual node pair).

The assumptions made on the persistence of network entities (e.g., the strategy identified to fix their TTL) play a crucial role in the results provided by time-aware mining algorithms.

CHAPTER 3

Literature Review

This chapter contains a brief overview of communities and their characteristics. We also describe the present methodologies for community discovery.

3.1. Communities

A community is another fundamental constituent in dynamic networks. There are several different definitions of community in the literature, but the most common is as follows: Definition: (Community) Given an undirected graph G = (V,E), a community C is a sub graph $C = (Vc, Ec)$ where $Vc \rightarrow V, Ec \rightarrow E$, and the edges in a community C are more densely connected with each other than the rest of the graph G.

3.1.1. Attributes

Communities can have several attributes that help to characterize community structure, such as overlap with other communities, weighted membership when members belong to different communities with different degrees of connection to each of them, roles of specific members and hierarchy within the members of a community.

As previously mentioned, a community is a set of vertices, and the membership of each vertex in a network was implicitly assumed to be the result of a boolean decision. In the context of mobile games, the concept of community and community membership may be more complex. For example, in some of the above community definitions, e.g., most of the local ones [19] [20], the one based on Clique Percolation [21], and others [22] [23], it is possible for communities to overlap (Fig 3a). Community overlap is essential in mobile games' networks since it is common for gamers to participate in multiple communities like family, friends and co-workers.

Also, there are additional attributes that vertices of a network may own about communities. For example, different vertices may participate with varying degrees in a community depending on their centrality within it (Fig 3b). Furthermore, vertices may have discrete roles: for example, [24] define two roles (hubs and outliers) for vertices that are unassigned to any community. Hubs are connected to multiple communities and act as links, thus enabling interactions among communities. Outliers connect to a single community through a single link, therefore they are regularly viewed as noise.

Roles also represent a relevant attribute of vertices when relating to their community [25]. The most relevant roles are "loners", "big fish", "bridges" and "ambassadors". A depiction of these roles can be seen in figure 3c.

It is also possible to impose hierarchical (Fig 3d) or multi-scale structures on communities. Community organization operates at many different levels in a variety of communities. For instance, a group of players may belong to a community focused on a particular game activity

(e.g. nine-ball pool players in mobile pool games), and at the same time, the members of this group can also be members of a broader community (pool games).

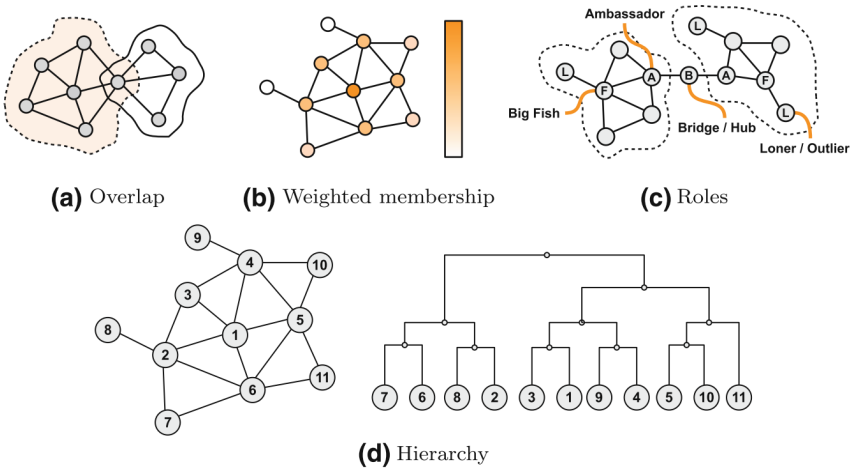


Figure 3. Community Attributes [26]

3.1.2. Life-Cycle

The persistence over time of communities that undergo continuous changes is a significant problem to tackle. Most efforts focused on community tracking agree on the set of simple actions that involve entities of a dynamic network: node/edge rise and vanish. Surely, such local and atomic operations can make perturbations in the network topology capable of affecting the outcomes delivered by community discovery algorithms. As a consequence of this set of actions, given a community C observed at different moments in time, it is mandatory to characterize the transformations it undergoes. The first categorization of the transformations that affect communities was introduced in [27], which lists six of them (Birth, Death, Growth, Contraction, Merge, Split). A seventh operation, "Continue", is sometimes added to these. In [28], the authors present an eight operation (Resurgence).

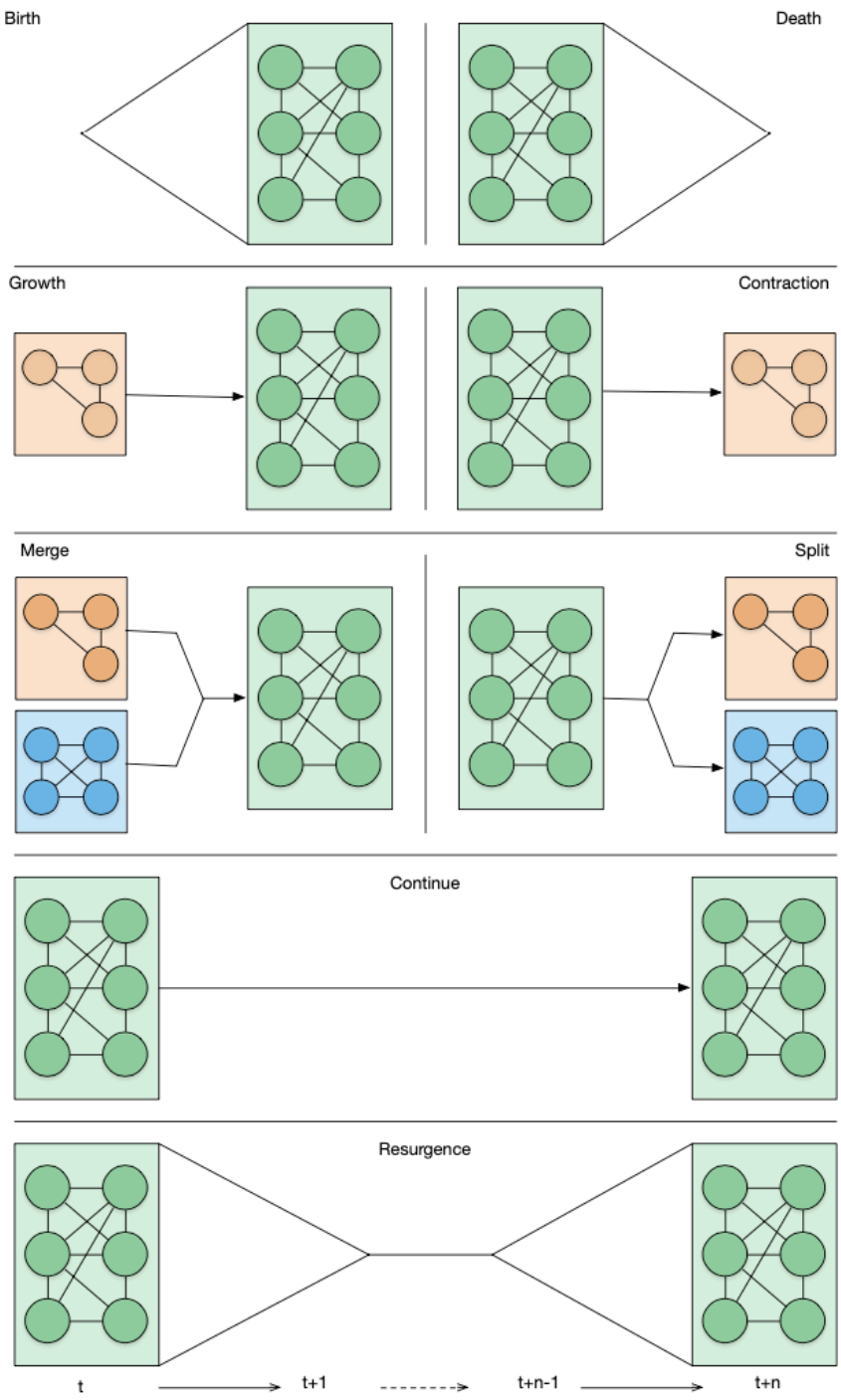


Figure 4. Community Events [12]

These transformations, represented in figure 4 from [12], are the following:

- Birth: the first appearance of a new community composed of any number of nodes;
- Death: the vanishing of a community: all nodes belonging to the vanished community lose this membership;
- Growth: new nodes increase the size of a community;
- Contraction: some nodes are rejected by a community thus reducing its size;
- Merge: two communities or more merge into a single one;
- Split: a community, as a consequence of node/edge vanishing, splits into two or more components;
- Continue: a community remains unchanged;
- Resurgence: a community vanishes for a period, then comes back without perturbations as if it has never stopped existing. This event can be seen as a fake death-birth pair involving the same node-set over a lagged period (example: seasonal behaviours).

3.1.3. Instability and Temporal Smoothing

One of the main issues encountered by dynamic community discovery methodologies is the instability of solutions. Such a problem comes from the very essence of communities. It is widely accepted that there is not a single valid breakdown in communities of a dynamic network issued from field data but, instead, several possible ones. Besides, most algorithms generate partitionings in which a vertice is assigned unambiguously to one and only one community. Indeed, we are aware that such a scenario represents a simplification: vertices often belong to many communities, and the belonging to each of them is not necessarily binary [12].

3.2. Community Discovery Methods

3.2.1. Classification

The literature contains several methods to detect communities for each community definition available. On this section, we will review the most prestigious groups of such methods according to [26] and associate them with the definitions presented in the previous section.

Before advancing with the analysis of the methods, we will first describe the relationship between the problem of community discovery with that of graph partitioning and graph clustering. Graph partitioning is a well-specified problem: divide the vertices of a graph into n groups of given sizes such that the number of edges lying between the groups (cut size) is minimum.

Community discovery is different from graph partitioning in two primary aspects. In the first place, community discovery requires neither the number of groups nor their sizes as input in order to extract them. Furthermore, the result of community discovery may not be a partition, i.e. a set of vertex sets, whose union is the set of all graph vertices and whose pairwise intersections always result in the empty set. As became apparent in the previous section, communities in a network may present overlapping, and there may be vertices in a network that do not belong to any community.

Community discovery is almost inter-changeably used with graph clustering [17] [29]. In both problems, the goal is to identify groups of vertices on a graph that connect strongly with

each other than with the rest of the network. Nevertheless, differentiation among the two problems regards the requirement for knowing the number of communities/clusters that a method is expected to identify. Community discovery methods usually do not need the number of communities to be provided, but instead, the number of communities is one of the method outputs. In contrast, there are diverse graph clustering techniques that require the number of clusters as input. Due to the large scale and evolving nature of mobile gaming, it is nearly unthinkable to know or even to estimate the number of communities in a mobile game network.

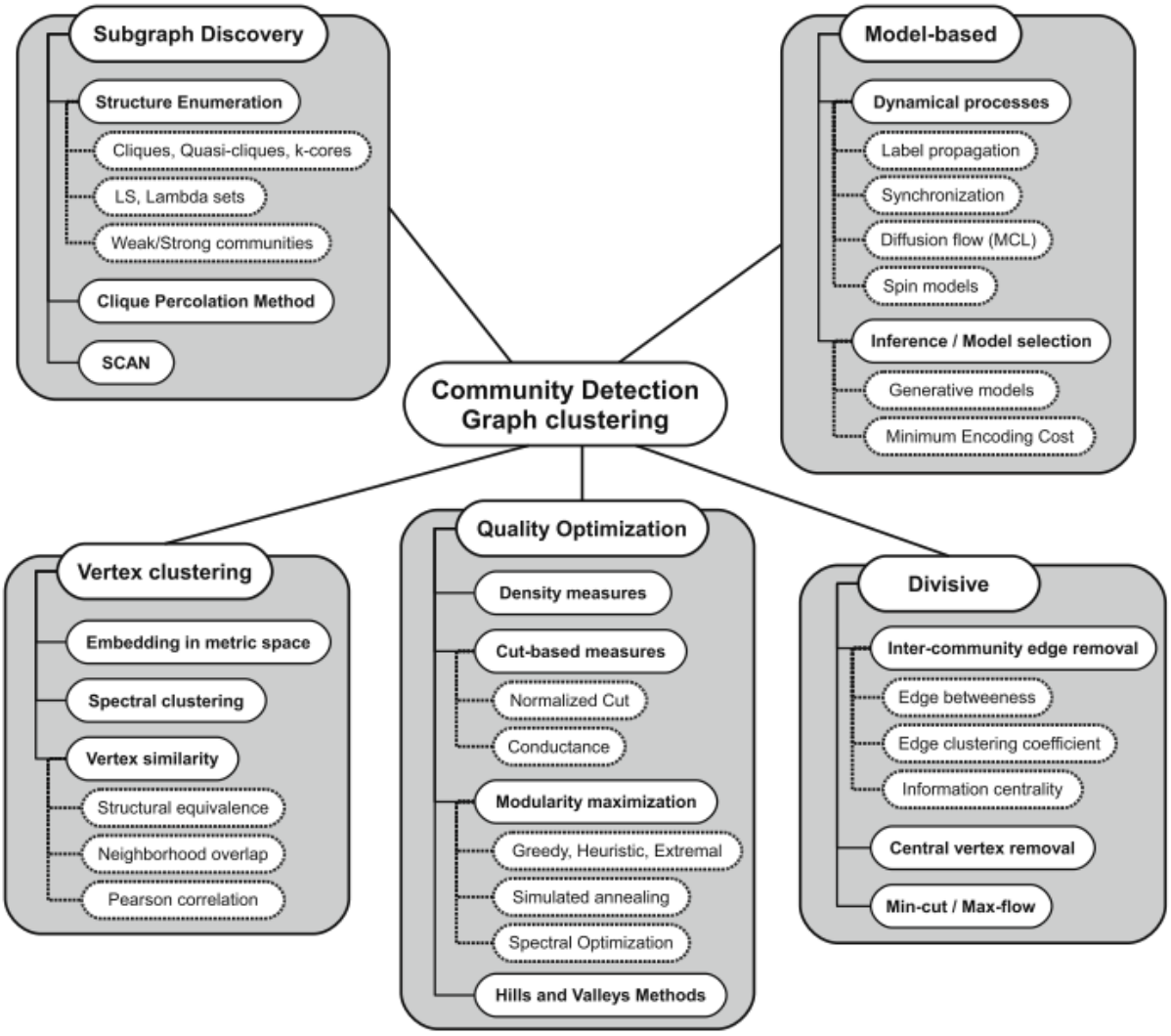


Figure 5. Classification of community discovery and graph clustering methods [26]

3.2.1.1. Cohesive Subgraph Discovery

For cohesive subgraph discovery, the structural properties that the subgraph must satisfy in order to be considered a community must be defined a priory. Once a subgraph structure is specified, methods involve the enumeration of such structures in the network under study.

The local community definitions presented in [26] are examples of such cohesive structures and therefore algorithmic schemes for enumerating such structures, such as the Bron–Kerbosch algorithm [30] and the efficient k-core decomposition algorithm of [8], belong to this class of

community discovery methods. Also, methods such as the Clique Percolation Method [21] and the SCAN algorithm [24], which point to the discovery of subgraph structures with well-specified properties, fall under the same class of methods.

3.2.1.2. Vertex Clustering

The methods presented in this subsection originate from conventional data clustering research. The usual way of casting a graph vertex clustering problem to one that we can solve with traditional data clustering methods is through setting graph vertices in a vector space, where we can determine pairwise distances between vertices.

An alternative but yet popular method is to use the spectrum of the graph for mapping graph vertices to points in a low-dimensional space, where the cluster structure is more profound [31] [32].

Other vertex similarity measures such as the structural equivalence [33] and the neighbourhood overlap have been used to compute similarities between graph vertices [34].

Last but not least, Walktrap is a unique method that derives an optimal vertex clustering structure. The method uses random-walk based similarity between vertices and communities together with modularity in a hierarchical agglomerative clustering scheme [35].

3.2.1.3. Community Quality Optimization

A vast number of methods for community discovery work through the optimization of measures related to community quality. The first ones to quantify the community quality with network division into clusters were the normalized cut [36] and conductance [37]. The measure of modularity incited a whole new surge of research. There are many maximization techniques in the literature for modularity. This category includes all of them, starting from the most straightforward methods such as the seminal greedy optimization technique of Newman (2004) and faster versions of it such as max-heap based agglomeration [38] and iterative heuristic schemes [39] [40], until the more sophisticated, such as, extremal optimization [41], simulated annealing [42] and spectral optimization (Newman 2006).

Methods aiming at the optimization of local measures of community quality, such as local and subgraph modularity [19][43], also belong to this category. Lastly, this category incorporates methods that exploit the ”hills” and ”valleys” in the distribution of network-based node or edge functions, e.g. the ModuLand framework proposed by [44] and the ”reachability” measure by [23].

3.2.1.4. Divisive

Divisive methods rely on the identification of network elements (edges and vertices) that are between communities. For example, the seminal algorithm by [45] progressively removes the edges of a network based on an edge betweenness measure until communities emerge as detached components of the graph. Edge betweenness was measured in several ways, for instance, edge, random-walk, and current-flow betweenness [45], as well as information centrality [46] and the edge clustering coefficient [47]. Vertex removal methods [48] adopt the same principle;

such methods remove vertices in order to expose community structure. Lastly, min-cut/max-flow methods [49] [50] adopt a different divisive viewpoint: they try to identify graph cuts (i.e. sets of edges that separate the graph in pieces) that have a minimum size.

3.2.1.5. Model-based

Model-based methods either consider a dynamic process taking place on the network, which reveals its communities or an underlying model of statistical nature that can generate the division of the network into communities. Examples of dynamic methods are label propagation [51] [52] [22], synchronization of Kuramoto oscillators [53], diffusion flow, also known as Markov Cluster Algorithm [54], and the famous SPIN model by [55]. Also, community discovery can be cast as a statistical inference problem (Hastings 2006), assuming some underlying probabilistic model, such as the planted partition model, that generates the community structure and estimates the parameters of this model. Other model-based approaches rely on the principle that a low encoding cost determines a functional clustering, so they perform community discovery by finding the cluster structure that results in the lowest possible cluster encoding cost [56].

3.2.2. Performance

Several studies have been done in comparing the performance of the different community discovery methods. In [26] the author identifies computational complexity and memory requirements as two key factors when considering different methods. In table 2, we can see the authors comparison of all methods complexity based on two different scenarios. From table 2 the author concludes that vertex clustering and divisive approaches present complexities higher than quadratic to the number of network vertices, which renders them improper for large scale networks such as social networks.

The author also develops an experimental study that compares eight methods using synthetic benchmark graphs from [57]. The study measured execution time, Normalized Mutual Information (NMI) and peak memory consumption for each method. The author concluded that the fastest methods were label propagation [51] (0.5s) and community folding [39] (0.6s) with the lowest ones being MCL [54] (75s) and SPIN [55] (19min). Community folding [39] was also the method with least memory consumption. All methods suffered considerable losses in NMI with the increasing number of nodes but community folding [39], walktrap [35] and SPIN [55] were the most resilient.

From table 2 we selected three methods to perform community discovery in our methodology. Since mobile games tend to have a high number of players [2] we chose methods that were expected to have good performance in populations of more than 100,000 players. The selected methods were Label propagation [51] [52], Community folding [39] and Spectral optimization [13].

Method	Complex-A	Complex-B	Scale
Cohesive substructure detection			
Algorithm 457 [58]	$O(3^{n/3})$	$O(3^{n/3})$	S
k-core detection [59]	$O(n^2)$	$O(n)$	L
SCAN [24]	$O(n^2)$	$O(n)$	L
Vertex clustering			
Embedding in space + k-means	$O(C\,n^2)$	$O(C\,n^2)$	M
Walktrap [35]	$O(n^4)$	$O(n^2\,log\,n)$	M
Laplacian eigenvectors [31]	$O(C\,n^2)$	$O(C\,n^2)$	M
Community quality optimization			
[38]	$O(n^2\,d\,log\,n)$	$O(n\,log^2\,n)$	M
Extremal optimization [53]	$O(n^2\,log\,n)$	$O(n^2\,log\,n)$	M
Spectral optimization [13]	$O(n^2\,log\,n)$	$O(n^2\,log\,n)$	M
Community folding [39]	$O(n^2)$	$O(n)$	L
Divisive			
[45]	$O(n^5)$	$O(n^3)$	S
Information centrality [46]	$O(n^7)$	$O(n^4)$	S
Edge clustering coefficient [47]	$O(n^6)$	$O(n^2)$	M
Max flow + Gomory-Hu tree [50]	$O(n^4\,log\,n)$	$O(n^3\,log\,n)$	S
Model Based			
MCL [54]	$O(n^3)$	$O(n^3)$	M
Minimum encoding cost [56]	$O(n^2)$	$O(n)$	L
Label propagation [51] [52]	$O(n^2)$	$O(n)$	L
Infomap [60]	$O(n^2\,log\,n)$	$O(n\,log\,n)$	L

Table 2. Two bounds are provided, one for general graphs irrespective of density (Complexity-A)and one computed under the assumption that the graph is sparse (Complexity-B). Furthermore, the scale of graphs for which each method is appropriate is provided: S stands for small scale ($< 10^4$ nodes), M stands for medium scale ($< 10^6$ nodes), and L for large scale (10^6–10^9 nodes) [26]

CHAPTER 4

Research Methodology

This chapter presents the methodology used to analyse mobile gaming interactions. Firstly we will describe the ETL process used to collect data and build a network of interactions of a real mobile game. Secondly we will demonstrate how we discovered and classified communities based on their life-cycle events. Lastly we will present our hypotheses for this research through friendship recommendation.

Through this chapter, we will describe several networks scenarios, and we will always refer to the vertices as players and the edges as relations.

4.1. Data Preparation

Having a reliable dynamic network is essential to make community discovery. To achieve that we defined two social interactions between players that we wanted to observe and that we also consider widely present in the mobile gaming industry. We did this to ensure that this methodology was reproducible and straightforward across all games.

The first relation is the most basic form of player interaction in mobile games which is playing a match between two players. More than two players can participate in a single match, but on this thesis, we will just consider matches of two players.

Mobile games are social games [61] so the second interaction we defined was friendship. Friendship is the most common form of social relationships found in social networks and subsequently in social games. It is common in mobile games to have a friends list where the player can see what friends (other players) are online or even what activity are they doing in the game.

To be able to build a relational network of these interactions, we defined the following set of events to observe:

- Match - A match played between two players.
- Add-friend - Friendship formed between two players.
- Remove-friend - Friendship broken between two players.

4.1.1. Collection

To collect events, we developed a REST web service (identified in figure 6 as T1) that integrated with the game and received a request each time an event was observed during gameplay. T1 was deployed in a scalable cluster with load balancing assure that we could handle large amounts of players and data.

As shown in figure 6, after the web service received an event, the JSON payload of the request was forwarded to a Apache Kafka [62] topic.

The topic in this process acted as a buffer mechanism that guaranteed that no events were lost until they were consumed by T2. Since our REST web service (T1) was internet-facing, it was important to delegate the file generation to other service (identified in figure 6 as T2). This service was also deployed in a scalable cluster that was solely responsible by consuming events from the topic and generating files into the data lake.

File size in modern data lakes should not be too small since it is prejudicial for performance. For this thesis we've chosen to use JSON files with an average of 256Mb and GZIP compression. Other file sizes or types are viable to use such as Parquet [63], ORC [63] or AVRO [64].

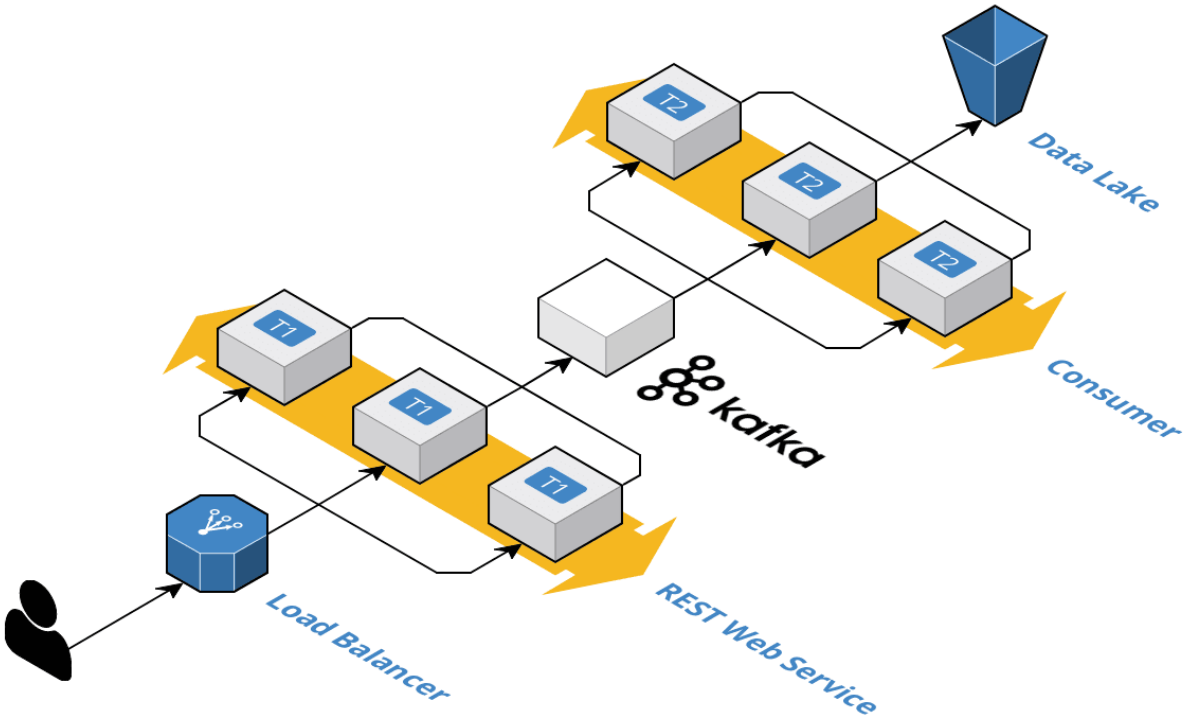


Figure 6. Data collection process

To assure that we did not have relevant data consistency issues, we defined schemas for each event that were validated by the REST web service (T1). If any event sent by the device did not meet the schema, it was discarded and not forwarded to the topic. Examples of the body of the requests are presented in the following listings.

Listing 4.1. Match event example

```
{
    "timestamp": 1598831892,
    "user_id": 1,
    "opponent_id": 2
}
```

Listing 4.2. Add-friend event example

```
{
    "timestamp": 1598831892,
    "user_id": 1,
    "friend_id": 2
}
```

Listing 4.3. Remove-friend event example

```
{
    "timestamp": 1598831892,
    "user_id": 1,
    "friend_id": 2
}
```

4.1.2. Transformation

With files on the data lake, we began to build our dynamic network using the graph database Neo4J [65]. Through the process presented in figure 7, we can see that a worker (T3) was used to pick up event data from the data lake and generate Cypher [66] statements to insert it into Neo4J.

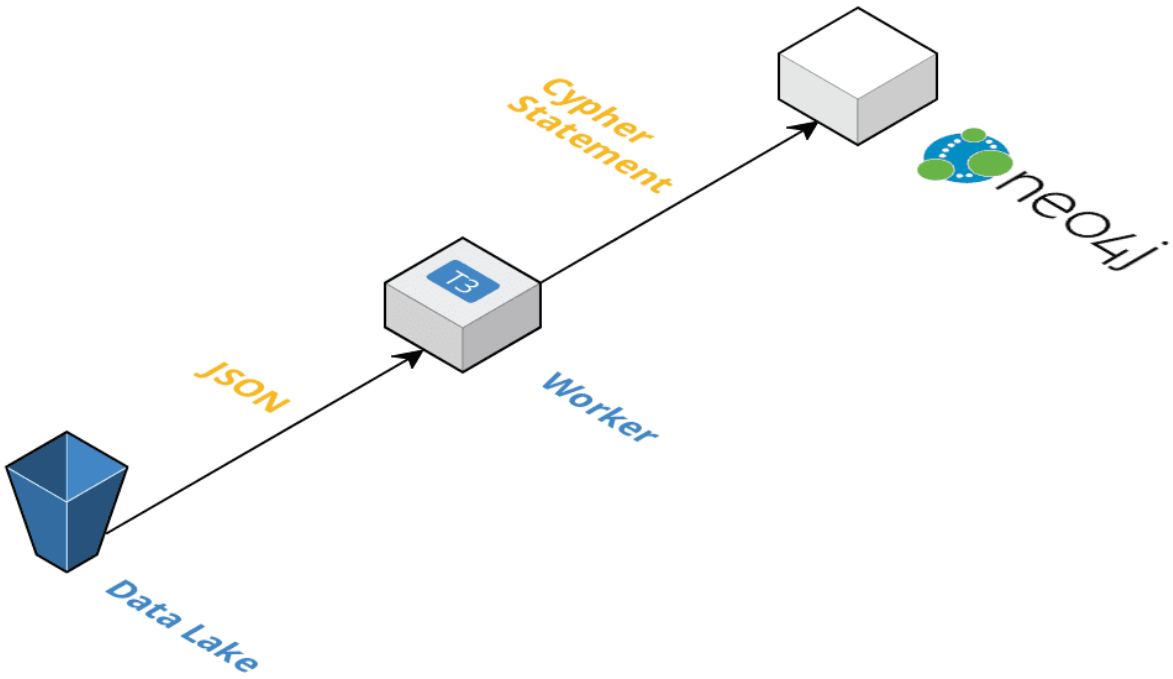


Figure 7. Data transformation process

For the worker to convert each event into Cypher statements we needed to make some data modelling assumptions.

4.1.3. Modelling

In this subsection we will present the assumptions used to improve the quality of the dynamic network and provide a better foundation for community discovery.

4.1.3.1. Match

Each time two players played a match, we considered that as a relation between them such as presented in figure 8.

Figure 8. Unweighted match relation

This relation between players was undirected and could be unweighted. Although since players could, and did, play more than one match between them through time, we made this relation weighted, and each new match between the same players would increment the weight by one. In figure 9, we can see that relation illustrated after three match events.

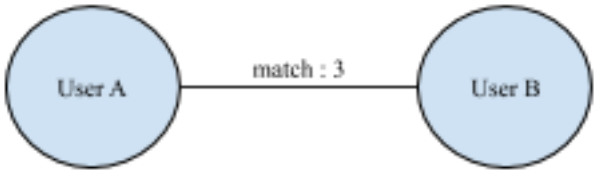


Figure 9. Weighted match relation

4.1.3.2. Friendship

Each time a player becomes an in-game friend of another player, we considered that as a relation between them such as presented in figure 10.

Figure 10. Friends relation

The friendship between two players was an undirected and unweighted relation. The relation was removed if the two players lost friendship, or by other words, generated a remove-friend event.

4.1.3.3. Dynamic Network

With the assumptions previously presented, we built a dynamic network in Neo4J that had all the events relational data. As presented in figure 11 players could have friendship and match relations, or both.

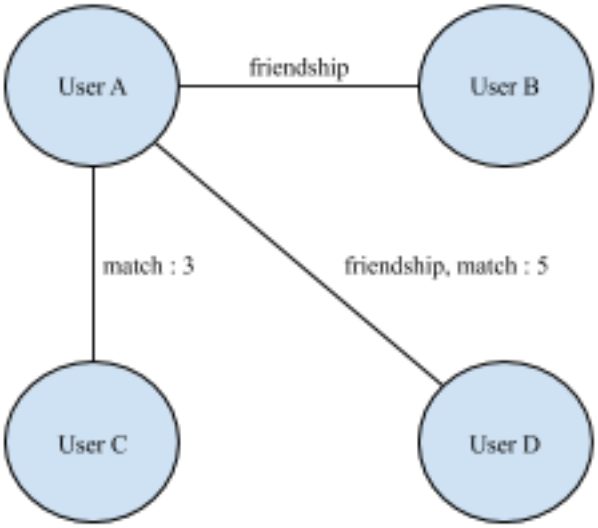


Figure 11. Network example

4.1.4. Snapshots

For the dynamic network created, we defined that data would be collected for 60 days. This period of 60 days had five instants of observation separated by periods of 15 days, as illustrated in figure 12.

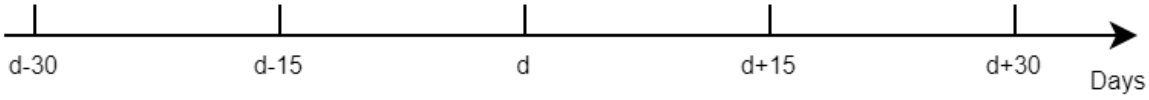


Figure 12. Snapshots time division

With this distribution data would be partitioned into 5 snapshots correspondent to each instant in time.

4.2. Community Discovery

To apply community discovery methodology to our network, we needed to understand what methods would work best. Mobile games tend to have a high number of players [2] so we have selected three methods that, based on previous research [26], were expected to have better performance in populations of more than 100,000 players, as shown in table 3.

Name	Complexity	Ref
Label propagation	$O(n^2)$	[51]
Community folding	$O(n^2)$	[39]
Spectral optimization	$O(n^2 \, log \, n)$	[13]

Table 3. Methods selected

Label propagation, as the name implies, works by propagating labels across the network and forming communities based on this. The method starts each node with a unique community label and then at every iteration updates nodes labels with the label that the maximum number of neighbours belong to. Ties are resolved arbitrarily. The methodology takes the assumption that a single label can quickly become dominant in a densely connected group of nodes, but will have trouble crossing a sparsely connected region. After a determined number of iterations the nodes that end with the same label are considered part of the same community.

Community folding and spectral optimization are both modularity based methods. They work with the target of achieving the highest possible modularity score. Modularity is a measure of the structure of a graph, measuring the density of connections within a module or community. Graphs with a high modularity score will have many connections within a community but only few pointing outwards to other communities.

Community folding merges nodes into communities at each iteration taking into account which merge will provide the highest modularity increase possible. Spectral optimization uses the shortest-path betweenness score of all the edges in the network to remove the edge with the highest score, recalculating the scores after removal. Both methods repeat these processes over a determined number of iterations until they stop and the resulting communities are determined.

The three methods were tested with the dynamic network and compared against each other regarding number of communities found, execution time and similarity of resulting communities. To execute each method we used implementations available in Neo4J Graph Data Science library.

4.2.1. Graphs

To start we created Neo4J graphs with our network using the following Cypher command:

```
CALL gds.graph.create(
    'match-graph',
    'User',
    {
        MATCH: {
            orientation: 'UNDIRECTED'
        }
    },
    {
        nodeProperties: 'previous_community',
        relationshipProperties: 'weight'
    }
)
```

The command would vary depending on what relation we wanted to use, match, friendship or both. If we wanted to use both, we applied the following formula to weight, where f stands for friendship relation, w stands for relation weight and m stands for number of matches.

$$(\exists f \exists m \Rightarrow w = m * 2) \wedge (\nexists f \exists m \Rightarrow w = m) \wedge (\exists f \nexists m \Rightarrow w = 1)$$

These graphs would guarantee that all nodes had a seed property that would tell which was the previous community detected for a player. This property will be used in another section to describe how we tracked community events.

4.2.2. Execution

After the graph was created we used the following command to run the methods:

```
CALL gds.<method>.write(
    'graph',
    {
        writeProperty: 'community'
    }
)
YIELD communityCount, modularity
```

This command would write the resulting community of each player as a property of the node in Neo4J. The command would also output the number of communities, modularity and execution time.

4.2.3. Jaccard Similarity

To understand if methods were achieving similar results we compared resulting communities using Jaccard similarity score. This score is defined as the size of the intersection divided by the size of the union of two sets. The two sets used were the user IDs of the same community resulting from different methods. To achieve this we executed following Cypher command per resulting community:

```
RETURN gds.alpha.similarity.jaccard([1,2,3], [1,2,4,5]) AS similarity
```

After the the similarity scores were obtained for all communities we would calculated the average and use that as a similarity score between methods.

4.3. Community Life-cycle Analysis

As previously mentioned in another chapter, it's difficult to track changes that communities undergo over time. For this thesis, we defined the goal of observing two life-cycle events that could classify communities as growing or contracting. These events were:

- Growth: new nodes increase the size of a community;
- Contraction: some nodes are rejected by a community thus reducing its size;

Communities classified as growing would be the ones withstanding the event of growth, and on the opposite, communities with the contraction event would be classified as contracting.

To understand which events communities were experiencing we used a snapshot based comparison. We did this with an iterative process as shown in figure 13.

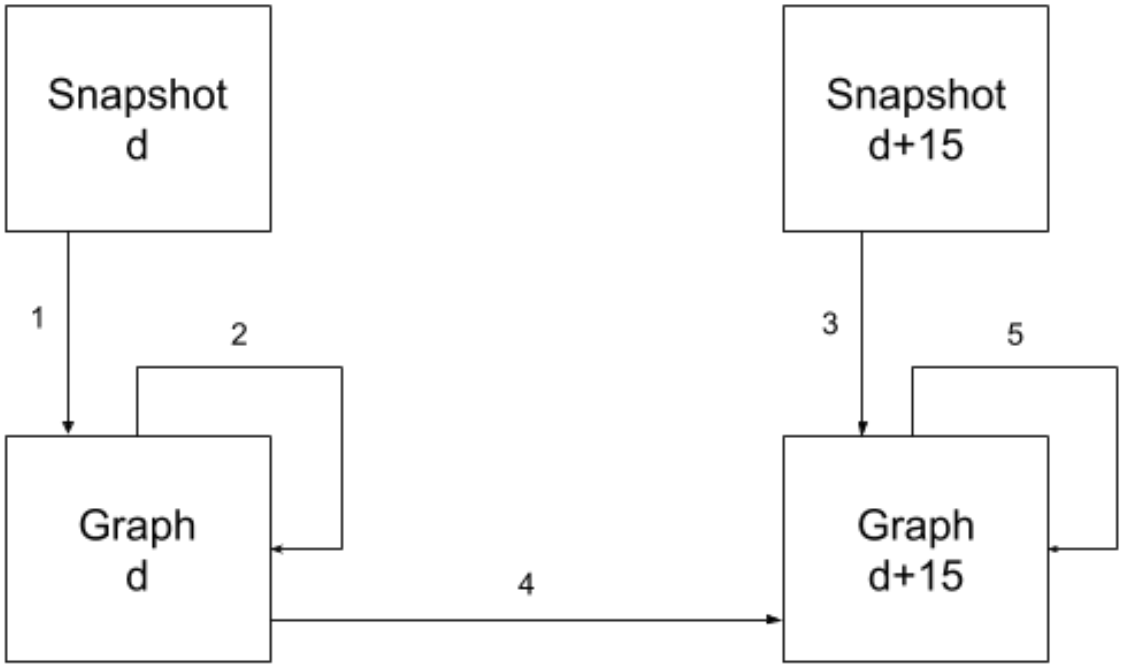


Figure 13. Life cycle analysis process: 1 - Generate graph; 2 - Community discovery process; 3 - Generate graph 15 days after; 4 - Copy resulting communities from previous iteration; 5 - Community discovery with seed property (previous_community)

To start we generated a graph from snapshot d where d represents an instant in time. We then performed community discovery and identified each node community. Afterwards we generated a new graph from the next snapshot $d + 15$ (15 days after) and copied the resulting communities from the previous iteration into the property ”previous_community” on this new graph. Finally we performed community discovery on this new graph with the property ”previous_community” as seed property. The Cypher command used for community discovery with seed property is the following:

```
CALL gds.<method>.write(
    'graph',
    {
        seedProperty: 'previous_community',
        writeProperty: 'community'
    }
)
YIELD communityCount, modularity
```

This life cycle analysis process was iterative and repeated through all snapshots. With the results we could compare communities size across time and understand the events that they went through, classifying them as growing or contracting.

4.4. Research Hypotheses

Contributing positively to social engagement in a mobile game in a non-intrusive way is a stiff challenge. Several factors can impact players engagement, and it is not easy to prove a cause-effect relation. A widespread approach when releasing a new game feature is to do AB testing [67]. Unfortunately AB testing involved time constraints that our schedule could not meet. As such, we opted for a predictive analysis approach that could prove our hypotheses and, if valid, any game could quickly implement as a full-fledged feature.

4.4.1. Friendship Recommendation

Friend recommendation engines usually recommended based on users similar lifestyles, habits, interests or behaviours. Since players on the same game already share a similar interest or behaviour, we thought that using the community discovery knowledge in that context could provide a different perspective for this problem.

Taking into consideration that our network is now composed of communities of players that are friends and/or play matches together, this thesis proposes friendship recommendations among players that belong to the same community but are not friends already as shown in figure 14.

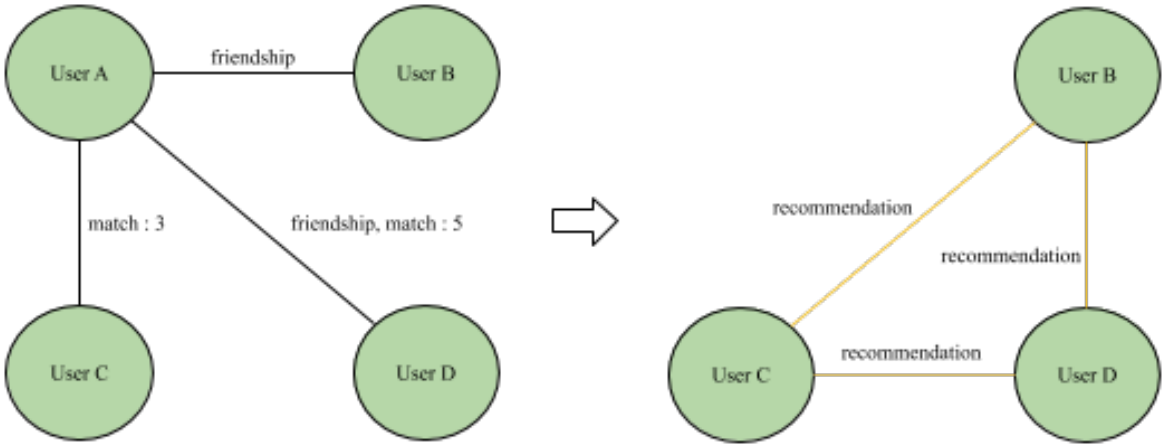


Figure 14. Friendship recommendation example

This hypotheses takes inspiration from past applications of friend recommendation in social graphs [68] [69] and uses community knowledge and game-specific data to improve on that.

CHAPTER 5

Results

This chapter presents the results obtained with our methodology. The first section will cover how real mobile game data was collected in collaboration with a mobile gaming company and what samples were selected and used. Secondly we will present the results obtained with the test of the different methodologies. Afterwards we will present the results of community life cycle analysis using the methodology that presented the most promising results. Lastly we will present the testing results of our hypotheses of friendship recommendation.

Across the chapter we'll present statistics measurements such as community size, number of relations or number of nodes. These measurements were obtained using simple querying Cypher statements.

5.1. Data Collected

The data used for this thesis was collected from a live mobile game during 60 days with an average of, approximately, 20M daily active users (DAU). All players data was collected anonymously and will not be available in an appendix or through a repository.

We sampled data based on different countries, resulting on three datasets that could represent different size of populations in the game. The countries chosen were France, Nigeria and Portugal, with the characteristics presented in table 4. In the table, we can see the number of players that each country has, together with the number of match and friend relations.

Country	Players	Matches	Friends
France	430479	2150708	456476
Nigeria	91348	337871	504462
Portugal	132829	754671	775121

Table 4. Countries statistics at d

In this game, France is the one of the biggest countries (430479 players), Nigeria is one of the smallest (91348 players) and Portugal is the averaged sized country (132829 players). Even with the highest player count, France presents the lower ratio of relations (matches and friends) per player with a value of 6.05 in comparison to Portugal 8.83 and Nigeria 9.22.

5.2. Community Discovery

Our focus for this thesis was to be able to detect communities in large networks. However, due to the ambiguity in the definition of a community, extracting communities and evaluating their quality has proven to be very difficult. In order to find out which method was the most adequate for our use-case, we tested the three methods against our three datasets as presented in table 5 and table 6. The test was on a 4 CPU and 16GB memory computer hosting only Neo4J.

Method	Communities		
	France	Nigeria	Portugal
Label propagation	29066	19221	15432
Community folding	29520	19434	15614
Spectral optimization	29257	19372	15568

Table 5. Number of communities found per methodology

Method	Execution Time		
	France	Nigeria	Portugal
Label propagation	6 min 32 secs	2 min 11 secs	3 min 45 secs
Community folding	4 min 12 secs	1 min 35 secs	2 min 21 secs
Spectral optimization	5 min 23 secs	1 min 58 secs	3 min 13 secs

Table 6. Execution time per methodology

The number of communities found by the three methods was similar but the execution time was not. In fact label propagation has proven to be the slowest and it scaled almost linearly as node and relation size increased. Community folding was the fastest and it scaled well taking just 4 minutes and 12 seconds to process France dataset.

We compared the resulting communities for the Portugal dataset using Jaccard index as shown in table 7 and the results were very similar, specially Community folding and Spectral optimization.

Method 1	Method 2	Jaccard Similarity
Label propagation	Community folding	0.95
Label Propagation	Spectral optimization	0.96
Community folding	Spectral optimization	0.98

Table 7. Similarity between resulting communities in Portugal

Based on the scaling capacity to millions of vertices and edges [11] of Community folding and Spectral optimization, we excluded Label propagation, from further testing, since we wanted to make sure that this methodology could be propagated to more significant games. Due to our implementation of Community folding having some optimisations [40] the execution time was lower than Spectral optimization, so we chose Community folding as the most adequate to pursue the tests. All of the methods presented in table 7 have proven to be reliable options for this methodology and are suggested as possible future work.

5.3. Community Life-cycle

Using the Community folding methodology we pursued our analysis of the communities life cycles across the different datasets. To start off we evaluated the distribution of community sizes across the three countries on the snapshot $d0$ as shown in table 8.

In table 8 we can observe that almost half of the communities present in our datasets were composed of one or two players. The country with higher numbers of low size communities

Size	Communities		
	France	Nigeria	Portugal
1	10748	7849	6975
2	8600	5273	4192
3-5	5407	3687	2033
5-10	2303	1988	1470
10-50	1344	561	695
50-100	801	73	203
100-1000	305	3	44
+1000	12	0	2

Table 8. Communities size distribution at d

is Nigeria. These small communities are more exposed to death events since mobile retention rates are low after 15 and 30 days.

We conducted our life-cycle classification on the three countries across all time spectrum achieving the results presented in table 9. We excluded $d - 30$ since it served as the basis of our iteration process previously described in other chapter.

Country	Communities							
	d-15		d		d+15		d+30	
	G	C	G	C	G	C	G	C
France	3696	7179	3482	7296	4139	6362	4458	5873
Nigeria	2503	5082	4964	2842	2818	2973	3453	2612
Portugal	2341	1796	2675	1713	1593	2639	2792	1734

Table 9. Communities classification as growing (G) or contracting (C) over 60 days

In in table 9 we can observe that Portugal and Nigeria experienced an average growing classification of 23% and France around 13%. Regarding contraction, France was the country with the highest average (27%) while Portugal was the lowest (12%). During our analysis we also observed that the number of communities was stable across our whole analysis as shown in table 10.

Country	Communities				
	d-30	d-15	d	d+15	d+30
France	30291	29866	29520	29481	29947
Nigeria	18962	17987	19434	18269	19681
Portugal	14721	15026	15614	15342	15780

Table 10. Communities size over 60 days

5.4. Friendship Recommendation

Precision and recall are the most popular criteria used to evaluate a recommendation engine. In this section, we performed several offline experiments since we could not use the recommendations in a live game. To validate what was the acceptance of our recommendations we used

the snapshot d in our data to make recommendations and after that, we compared snapshots $d15$ and $d30$ to observe what recommendations were right or wrong.

We started by looking at the impact that each relation had in the performance of the recommendation. Lastly we compared results by looking at the performance by community sizes and by communities classified as growing or contracting.

5.4.1. Relations

In this subsection we'll start by evaluating the performance of the friendship relation by itself, meaning that the graph used for this test only had friendship type relations. Secondly we'll repeat the same test just for the match type relations. Lastly we'll combine both relations just like in previous sections using the formula mentioned in section 4.2.1. Precision and recall results presented in this subsection are cumulative, meaning that if a successfully predicted friendship was observed at $d + 15$ and not at $d + 30$ we still considered it as true positive.

5.4.1.1. Friend relation

The friend relation presented very conclusive results. This methodology struggles to achieve decent recall values when the average communities size is small. In table 14, we can observe that the number of friends in the highest population country is lower than the others. The number of friend relations in France is so low that the average number of friends per player is between 1 and 2 while the other countries present an average of 5. As we can observe in table 12, this reduced friends situation in France contributed positively to the precision of the algorithm but not to the recall. Since the proposed methodology will only suggest friend recommendations to players that are not friends yet, a network with a smaller average community size will result in a reduced number of recommendations. With a reduced number of recommendations, we can expect the recall to drop in accordance, but precision will not necessarily be affected.

On this test, modularity seems to help France in achieving higher results, but it is dubious since the average community size is 1.

In general, the precision of this friend relation was two to three times higher than the match, which clearly states that friends-of-friends matter the most in this methodology. The difference in recall was not expressive, mostly due to the low number of resulting recommendations.

Country	Players	Friends	Communities	Modularity	Avg. Size
France	430479	456476	259308	0.5177	1
Nigeria	91348	504462	23980	0.2882	1
Portugal	132829	775121	35513	0.2834	1

Table 11. Friend relation statistics

5.4.1.2. Match relation

The match relation was tough to analyse. The relation in itself does not achieve good precision results but recall values are acceptable as shown on table 14. The average community size in this relation is between 5 and 7 players depending on the country, as shown in table 13. That average community size contributes to a high number of recommendations leading to closer

Country	Precision (d+15)	Recall (d+15)	Precision (d+30)	Recall (d+30)
France	0.2022	0.0202	0.2343	0.0234
Nigeria	0.1234	0.0575	0.1312	0.0611
Portugal	0.1544	0.0907	0.1761	0.1034

Table 12. Friend relation precision and recall

Country	Players	Matches	Communities	Modularity	Avg. Size
France	430479	2150708	97013	0.2831	5
Nigeria	91348	337871	45412	0.2342	4
Portugal	132829	754671	36267	0.35	7

Table 13. Match relation statistics

values between precision and recall. When the recall is the same as precision we can conclude that the number of recommendations is the same as the true positives. Even with lower precision values, the match relation revealed much better recall to precision ratio than the friend relation.

In multiplayer gaming, there is always solo players [70] that tend to have a reduced number of friends. The match relation proves useful to be able to create recommendations for players where a friend relation network would not result.

Country	Precision (d+15)	Recall (d+15)	Precision (d+30)	Recall (d+30)
France	0.0326	0.0289	0.0389	0.0345
Nigeria	0.0776	0.0622	0.0902	0.0723
Portugal	0.0570	0.0423	0.0785	0.0583

Table 14. Match relation precision and recall

5.4.1.3. Combined relations

The combination of match and friend relations shows potential for this methodology. By themselves, none of the relations achieved a similar result. As shown in table 15, the average community size in combined relations is between 8 and 10, which the highest in these tests. This shows that players play more games with friends of friends or unknown players than with their friends.

Modularity also plays an important role; throughout the test, we have seen a direct relation in modularity versus precision in 30 days, as shown in table 15 and table 16.

Country	Players	Matches	Friends	Communities	Modularity	Avg. Size
France	430479	2150708	456476	29520	0.3179	8
Nigeria	91348	337871	504462	19434	0.3130	8
Portugal	132829	754671	775121	15614	0.3379	10

Table 15. Combined relations statistics

Country	Precision (d+15)	Recall (d+15)	Precision (d+30)	Recall (d+30)
France	0.2112	0.1711	0.2652	0.2148
Nigeria	0.2438	0.1988	0.2708	0.2208
Portugal	0.2784	0.2562	0.3011	0.2771

Table 16. Combined relations precision and recall

5.4.2. Community Size

In table 17 we can observe the distribution of the precision at $d + 30$ by community size. The size with most precision is 10-50 followed by 50-100. The size with the lowest precision is +1000.

Size	Precision(d+30)		
	France	Nigeria	Portugal
1	0,0134	0,0103	0,0231
2	0,0145	0,0231	0,0213
3-5	0,0214	0,0507	0,0496
5-10	0,0322	0,0623	0,0545
10-50	0,0406	0,0721	0,0682
50-100	0,0582	0,0421	0,0532
100-1000	0,0604	0,0102	0,0231
+1000	0,0245	0,0000	0,0081

Table 17. Precision distribution over communities size at $d + 30$

5.4.3. Life Cycle

This subsection presents the results of measuring the precision of the recommendations considering just the growing and contracting communities. Precision results presented in this subsection are cumulative, meaning that if a successfully predicted friendship was observed at $d+15$ and not at $d + 30$ we still considered it as true positive. In table 18 we can observe that contracting communities have much lower precision values than growing communities across all countries. Portugal was the country with the highest precision at $d+15$ in growing communities with 77,31%. All countries had an average increase of 3% from $d + 15$ to $d + 30$ in growing communities with Portugal achieving 78,12% at $d + 30$. Contracting communities had an average of 3% precision results on $d + 15$ and 5% on $d + 30$. The lowest result was Nigeria with only 4,15% at $d + 30$.

Country	Precision			
	d+15		d+30	
	G	C	G	C
France	0,5828	0,0342	0,6433	0,0509
Nigeria	0,6521	0,0293	0,6801	0,0415
Portugal	0,7731	0,0711	0,7912	0,0937

Table 18. Precision at $d + 15$ and $d + 30$ for growing (G) or contracting (C) communities

CHAPTER 6

Conclusions and future research

As mentioned in the introductory chapter of this thesis, social interaction plays a vital role in the development of mobile games. Backed by scientifically validated community discovery methodologies, it is possible to make a concrete contribution.

So how can we develop a methodology that is easily reproducible across other mobile games? An easily reproducible methodology must use common mobile gaming interactions and academic valid methods. Friendships and matches are common in most mobile games just like modularity optimisation methods are in community discovery literature. We also suggest in the future work section other interactions widely present in the industry.

To evaluate performance and quality in real use cases, without user feedback, we concluded that predictive analysis is a viable solution. The results presented in this thesis clearly show that we can use dynamic networks and community knowledge to better understand players and their social attributes.

To deal with temporal data we concluded that snapshots provide a good basis for analysis. With snapshots, we could observe growth and contraction events in the community life-cycle analysis and measure the performance evolution of our friendship prediction.

Small communities (2-5 players) were predominant in our mobile game representing an average of 44,4% of the total communities. Even with that, solo players still represented a great part of the population with an average of 40,4%.

Community evolution was very volatile in our mobile scenario. Growth and contraction events occurred on 35,3% of the communities on average, with Nigeria achieving 42,1%. Contraction events represented almost two times the number of growth events on all our datasets.

Friendships were predictable with some precision at $d+30$ (average of 27,5%) just by using community knowledge. When we applied the proposed methodology just to growing and contracting communities, we observed that contracting communities were not worthy of predicting with results as low as 4,15% at $d + 30$. On the opposite, growing communities showed significantly higher results, with the Portugal dataset achieving 79,12% precision at $d + 30$. Results over time were similar from $d + 15$ to $d + 30$ which showed that the prediction was particularly effective in the short term but limited in the long term.

Lastly, we concluded that this methodology can be boosted in the future by bringing other relations to the equation since the results of the combination of relations were higher than each relation by itself.

6.1. Contributions

This thesis serves the academic community as a hypotheses for the application of community discovery methodologies in mobile games. The comparison of different methodologies with a real mobile gaming scenario is also relevant and useful.

Community detection methodologies keep evolving and have already proven to be capable of handling large volumes of data. This thesis also demonstrates that through a simple recommendation system we can predict friendships until 79% in growing communities. If those recommendations were presented to the user and possibly adopted that could help boost communities growth. In games with 20 million DAU, increases in community size can mean a lot of revenue and business value.

6.2. Limitations and Future Work

As mentioned previously, this thesis explored how communities behave in the mobile gaming world. With that knowledge, we wanted to prove that a substantial contribution could be made. That focus and time constraints limited decisions and left some interesting questions and paths to be explored. To conclude this thesis, some recommendations to this effect are presented below.

- The results of relations by themselves have proven to be lacking when compared to the combination of them. That result has lit our interest in exploring further interactions and their possible combinations:
  - Gifting - Gift sent from one player to another.
  - Chat - Message sent from one player to another.
  - Challenge - Challenge sent from one player to another.
- Relation weight was a field that we left unexplored. The formula used on this thesis was simple; if the user had played more than one match with another, then the weight would be the number of matches between them. A linear regression or some optimisation algorithm would be very interesting to explore in order to increase the model precision and recall.
- The community detection algorithm used in this thesis was Louvain. Other options were considered but discarded due to scalability or performance. It would be interesting to see if other algorithms could achieve more accurate results or even be more suited in smaller populations.
- Due to time constraints, this thesis results were validated through a confirmation analysis perspective. However, it would be interesting to do proper AB testing of the recommendations and measure its impact on user retention and engagement.
- In the life cycle analysis it would be interesting to pursue the detection of other life cycle events such as birth or death. It would also be very interesting to explore an hypotheses that could relate birth or death of communities in mobile games.

References

[1] Newzoo, *Global Games Market Will Reach $102.9 Billion in 2017*, 2014. [Online]. Available: https://newzoo.com/insights/articles/the-global-games-market-will-generate-152-1-billion-in-2019-as-the-u-s-overtakes-china-as-the-biggest-market/.

[2] Activision Blizzard, *Activision Blizzard Announces First-Quarter 2019 Financial Results | Activision Blizzard, Inc.* 2019. [Online]. Available: https://investor.activision.com/news-releases/news-release-details/activision-blizzard-announces-first-quarter-2019-financial.

[3] C. Chapple, *Call of Duty Mobile Breaks Record with 100 Million Downloads in Its First Week*, 2019. [Online]. Available: https://sensortower.com/blog/call-of-duty-mobile-first-week.

[4] Adjust, *User acquisition strategy: Best practices for marketing mobile games*, 2019. [Online]. Available: https://www.adjust.com/blog/guide-user-acquisition-mobile-games/.

[5] Megacool, *The future of sharing for mobile games*, 2018. [Online]. Available: https://megacool.co/blog/the-future-of-sharing-for-mobile-games/.

[6] N. Ducheneaut, N. Yee, E. Nickell, and R. J. Moore, "The life and death of online gaming communities: A look at guilds in world of warcraft," in *Conference on Human Factors in Computing Systems - Proceedings*, 2007, pp. 839–848, isbn: 1595935932. doi: 10.1145/1240624.1240750.

[7] M. McGlohon, L. Akoglu, and C. Faloutsos, "Statistical Properties of Social Networks," in *Social Network Data Analytics*, Springer, 2011, pp. 17–42. doi: 10.1007/978-1-4419-8462-3{_}2.

[8] V. Batagelj, "Efficient Algorithms for Citation Network Analysis," *arXiv preprint cs/0309023*, 2003. [Online]. Available: http://arxiv.org/abs/cs/0309023.

[9] M. Fatemi and L. Tokarchuk, "An empirical study on IMDb and its communities based on the network of co-reviewers," in *Proceedings of the 1st Workshop on Measurement, Privacy, and Mobility, MPM'12*, 2012, p. 7, isbn: 9781450311632. doi: 10.1145/2181196.2181203.

[10] C. C. Aggarwal, "An Introduction to Social Network Data Analytics," in *Social Network Data Analytics*, Springer, 2011, pp. 1–15. doi: 10.1007/978-1-4419-8462-3{_}1.

[11] M. Coscia, F. Giannotti, and D. Pedreschi, *A classification for community discovery methods in complex networks*, 2011. doi: 10.1002/sam.10133.

[12] G. Rossetti and R. Cazabet, "Community discovery in dynamic networks: A survey," *ACM Computing Surveys*, vol. 51, no. 2, p. 35, 2018, issn: 15577341. doi: 10.1145/3172867.

[13] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 69, no. 2 2, pp. 1–16, 2004, issn: 1063651X. doi: 10.1103/PhysRevE.69.026113.

[14] T. Murata, "Detecting communities from social tagging networks based on tripartite modularity," *Other*, 2011. [Online]. Available: http://www.kddresearch.org/Workshops/IJCAI-2011-HINA/Papers/murata11detecting.pdf.

[15] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers and Security*, vol. 28, no. 1-2, pp. 18–28, 2009, issn: 01674048. doi: 10.1016/j.cose.2008.08.003.

[16] M. E. Newman, A. L. Barabási, and D. J. Watts, *The structure and dynamics of networks*. 2011, vol. 9781400841, pp. 1–582, isbn: 9781400841356. doi: 10.1007/s10955-006-9267-8.

[17] S. Fortunato, *Community detection in graphs*, 2010. doi: 10.1016/j.physrep.2009.11.002.

[18] C. Aggarwal and K. Subbian, "Evolutionary network analysis: A survey," *ACM Computing Surveys*, vol. 47, no. 1, p. 10, 2014, issn: 15577341. doi: 10.1145/2601412.

[19] A. Clauset, "Finding local community structure in networks," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 72, no. 2, p. 26 132, 2005, issn: 15393755. doi: 10.1103/PhysRevE.72.026132.

[20] F. Luo, J. Z. Wang, and E. Promislow, "Exploring local community structures in large networks," *Web Intelligence and Agent Systems*, vol. 6, no. 4, pp. 387–400, 2008, issn: 15701263. doi: 10.3233/WIA-2008-0147.

[21] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, 2005, issn: 00280836. doi: 10.1038/nature03607.

[22] S. Gregory, "Finding overlapping communities in networks by label propagation," *New Journal of Physics*, vol. 12, no. 10, p. 103 018, 2010, issn: 13672630. doi: 10.1088/1367-2630/12/10/103018.

[23] J. Chen, O. R. Zaïane, and R. Goebel, "A visual data mining approach to find overlapping communities in networks," in *Proceedings of the 2009 International Conference on Advances in Social Network Analysis and Mining, ASONAM 2009*, 2009, pp. 338–343, isbn: 9780769536897. doi: 10.1109/ASONAM.2009.15.

[24] J. J. Chen, V. L. Huang, J. Liu, and J. M. Chen, "PSCAN: A parallel structural clustering algorithm for networks," *Proceedings - International Conference on Machine Learning and Cybernetics*, vol. 2, pp. 839–844, 2013, issn: 21601348. doi: 10.1109/ICMLC.2013.6890400.

[25] J. Scripps, P. N. Tan, and A. H. Esfahanian, "Node roles and community structure in networks," in *Joint Ninth WebKDD and First SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*, 2007, pp. 26–35, isbn: 9781595938480. doi: 10.1145/1348549.1348553.

[26] S. Papadopoulos, Y. Kompatsiaris, A. Vakali, and P. Spyridonos, "Community detection in social media performance and application considerations," *Data Mining and Knowledge Discovery*, vol. 24, no. 3, pp. 515–554, 2012, issn: 13845810. doi: 10.1007/s10618-011-0224-z.

[27] G. Palla, A. L. Barabási, and T. Vicsek, "Quantifying social group evolution," *Nature*, vol. 446, no. 7136, pp. 664–667, 2007, issn: 14764687. doi: 10.1038/nature05670.

[28] R. Cazabet and F. Amblard, *Encyclopedia of Social Network Analysis and Mining*, 2018. doi: 10.1007/978-1-4939-7131-2.

[29] S. E. Schaeffer, "Graph clustering," *Computer Science Review*, vol. 1, no. 1, pp. 27–64, 2007, issn: 15740137. doi: 10.1016/j.cosrev.2007.05.001.

[30] C. Bron and J. Kerbosch, "Algorithm 457: Finding All Cliques of an Undirected Graph [H]," *Communications of the ACM*, vol. 16, no. 9, pp. 575–577, 1973, issn: 15577317. doi: 10.1145/362342.362367.

[31] L. Donetti and M. A. Muñoz, "Detecting network communities: A new systematic and efficient algorithm," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2004, no. 10, P10012, 2004, issn: 17425468. doi: 10.1088/1742-5468/2004/10/P10012.

[32] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007, issn: 09603174. doi: 10.1007/s11222-007-9033-z.

[33] R. L. Breiger, S. A. Boorman, and P. Arabie, "An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling," *Journal of Mathematical Psychology*, vol. 12, no. 3, pp. 328–383, 1975, issn: 10960880. doi: 10.1016/0022-2496(75)90028-0.

[34] E. Lazega, S. Wasserman, and K. Faust, *Social Network Analysis: Methods and Applications*, 4. Cambridge university press, 1995, vol. 36, p. 781. doi: 10.2307/3322457.

[35] P. Pons and M. Latapy, "Computing communities in large networks using random walks," *Journal of Graph Algorithms and Applications*, vol. 10, no. 2, pp. 191–218, 2006, issn: 15261719. doi: 10.7155/jgaa.00124.

[36] 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, issn: 01628828. doi: 10.1109/34.868688.

[37] R. Kannan, S. Vempala, and A. Vetta, "On clusterings: Good, bad and spectral," *Journal of the ACM*, vol. 51, no. 3, pp. 497–515, 2004, issn: 00045411. doi: 10.1145/990308.990313.

[38] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, vol. 70, no. 6, p. 6, 2004, issn: 1063651X. doi: 10.1103/PhysRevE.70.066111.

[39] 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, P10008, 2008, issn: 17425468. doi: 10.1088/1742-5468/2008/10/P10008.

[40] H. Lu, M. Halappanavar, and A. Kalyanaraman, "Parallel heuristics for scalable community detection," *Parallel Computing*, vol. 47, pp. 19–37, 2015, issn: 01678191. doi: 10.1016/j.parco.2015.03.003.

[41] J. Duch and A. Arenas, "Community detection in complex networks using extremal optimization," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 72, no. 2, pp. 2–5, 2005, issn: 15393755. doi: 10.1103/PhysRevE.72.027104.

[42] C. P. Massen and J. P. Doye, "Identifying communities within energy landscapes," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 71, no. 4, p. 46 101, 2005, issn: 15393755. doi: 10.1103/PhysRevE.71.046101.

[43] J. Zhao, H. Yu, J. H. Luo, Z. W. Cao, and Y. X. Li, "Hierarchical modularity of nested bow-ties in metabolic networks," *BMC Bioinformatics*, vol. 7, no. 1, p. 386, 2006, issn: 14712105. doi: 10.1186/1471-2105-7-386.

[44] I. A. Kovács, R. Palotai, M. S. Szalay, and P. Csermely, "Community landscapes: An integrative approach to determine overlapping network module hierarchy, identify key nodes and predict network dynamics," *PLoS ONE*, vol. 5, no. 9, pp. 1–14, 2010, issn: 19326203. doi: 10.1371/journal.pone.0012528.

[45] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, pp. 7821–7826, 2002, issn: 00278424. doi: 10.1073/pnas.122653799.

[46] S. Fortunato, V. Latora, and M. Marchiori, "Method to find community structures based on information centrality," *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, vol. 70, no. 5, p. 13, 2004, issn: 1063651X. doi: 10.1103/PhysRevE.70.056104.

[47] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Paris, "Defining and identifying communities in networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 9, pp. 2658–2663, 2004, issn: 00278424. doi: 10.1073/pnas.0400054101.

[48] I. Vragović and E. Louis, "Network community structure and loop coefficient method," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 74, no. 1, p. 16 105, 2006, issn: 15393755. doi: 10.1103/PhysRevE.74.016105.

[49] G. W. Flake, S. Lawrence, and C. L. Giles, "Efficient identification of web communities," in *Proceeding of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000, pp. 150–160, isbn: 1581132336. doi: 10.1145/347090.347121.

[50] H. Ino, M. Kudo, and A. Nakamura, "Partitioning of Web graphs by community topology," in *Proceedings of the 14th international conference on World Wide Web*, 2005, p. 661. doi: 10.1145/1060745.1060841.

[51] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 76, no. 3, p. 36 106, 2007, issn: 15393755. doi: 10.1103/PhysRevE.76.036106.

[52] I. X. Leung, P. Hui, P. Liò, and J. Crowcroft, "Towards real-time community detection in large networks," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 79, no. 6, 2009, issn: 15393755. doi: 10.1103/PhysRevE.79.066107.

[53] A. Arenas, A. Díaz-Guilera, and C. J. Pérez-Vicente, "Synchronization reveals topological scales in complex networks," *Physical Review Letters*, vol. 96, no. 11, p. 114 102, 2006, issn: 00319007. doi: 10.1103/PhysRevLett.96.114102.

[54] S. v. Dongen, "Graph clustering by flow simulation," Ph.D. dissertation, 2007, pp. 27–64, isbn: 90-393-2408-5. doi: 10.1016/j.cosrev.2007.05.001. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1574013707000020%5Cnhttp://linkinghub.elsevier.com/retrieve/pii/S1574013707000020.

[55] J. Reichardt and S. Bornholdt, "Statistical mechanics of community detection," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 74, no. 1, p. 16 110, 2006, issn: 15393755. doi: 10.1103/PhysRevE.74.016110.

[56] D. Chakrabarti, "AutoPart: Parameter-free graph partitioning and outlier detection," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3202, 2004, pp. 112–124, isbn: 3540231080. doi: 10.1007/978-3-540-30116-5{_}13.

[57] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 78, no. 4, 2008, issn: 15393755. doi: 10.1103/PhysRevE.78.046110.

[58] E. A. Akkoyunlu, "The Enumeration of Maximal Cliques of Large Graphs," *SIAM Journal on Computing*, vol. 2, no. 1, pp. 1–6, 1973, issn: 0097-5397. doi: 10.1137/0202001.

[59] V. Batagelj and M. Zaversnik, "An O(m) Algorithm for Cores Decomposition of Networks," *arXiv preprint cs/0310049*, 2003. [Online]. Available: http://arxiv.org/abs/cs/0310049.

[60] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 4, 2008, issn: 10916490. doi: 10.1073/pnas.0706851105.

[61] C. A. F. Gonzalez, S. R. Villaverde, J. L. G. Barroso, and J. M. Aguado, "Mobile gaming: Industry challenges and policy implications," *Telecommunications Policy*, vol. 36, no. 3, pp. 212–221, 2012. [Online]. Available: http://oa.upm.es/15635/.

[62] D. Vohra and D. Vohra, *Apache Kafka*. 2016, pp. 339–347, isbn: 9781782167938. doi: 10.1007/978-1-4842-2199-0{_}9.

[63] T. Ivanov and M. Pergolesi, "The impact of columnar file formats on SQL-on-hadoop engine performance: A study on ORC and Parquet," *Concurrency Computation*, vol. 32, no. 5, 2020, issn: 15320634. doi: 10.1002/cpe.5523.

[64] D. Vohra and D. Vohra, "Apache Avro," in *Practical Hadoop Ecosystem*, 2016. doi: 10.1007/978-1-4842-2199-0{_}7.

[65] J. J. Miller, "Graph database applications and concepts with Neo4j," in *Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA*, vol. 2324, 2013.

[66] N. Francis, A. Green, P. Guagliardo, L. Libkin, V. Marsault, S. Plantikow, M. Rydberg, P. Selmer, N. Francis, A. Green, P. Guagliardo, L. Libkin, T. Lindaaker, N. Francis, A. Green, P. Guagliardo, L. Libkin, T. Lindaaker, V. Marsault, S. Plantikow, P. Selmer, and A. Taylor, "Cypher : An Evolving Query Language for Property Graphs To cite this version : HAL Id : hal-01803524 Cypher : An Evolving Query Language for Property Graphs," 2018.

[67] S. W. H. Young, "Improving Library User Experience with A/B Testing: Principles and Process," *Weave: Journal of Library User Experience*, vol. 1, no. 1, Aug. 2014, issn: 2333-3316. doi: 10.3998/weave.12535642.0001.101.

[68] N. B. Silva, I. R. Tsang, G. D. Cavalcanti, and I. J. Tsang, "A graph-based friend recommendation system using genetic algorithm," *2010 IEEE World Congress on Computational Intelligence, WCCI 2010 - 2010 IEEE Congress on Evolutionary Computation, CEC 2010*, pp. 18–23, 2010. doi: 10.1109/CEC.2010.5586144.

[69] J. Naruchitparames, M. H. Gunes, and S. J. Louis, "Friend recommendations in social networks using genetic algorithms and network topology," *2011 IEEE Congress of Evolutionary Computation, CEC 2011*, pp. 2207–2214, 2011. doi: 10.1109/CEC.2011.5949888.

[70] M. Mora-Cantallops and M.-Á. Sicilia, "Player-centric networks in League of Legends," *Social Networks*, vol. 55, pp. 149–159, 2018, issn: 0378-8733. doi: https://doi.org/10.1016/j.socnet.2018.06.002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378873318301229.