Università
Ca'Foscari
Venezia

Master's Degree

in Computer Science

Final Thesis

Modularity Based Community Detection on the GPU

Supervisor

Ch. Prof. Claudio Lucchese

Graduand

Federico Fontolan

Matriculation Number

854230

Academic Year

2019 / 2020

Abstract

Modularity based algorithms for the detection of communities are the de facto standard thanks to the fact that they offer the best compromise between efficiency and result. This is because these algorithms allow analyzing graphs much larger than those that can be analyzed with alternative techniques. Among these, the Louvain algorithm has become extremely popular due to its simplicity, efficiency and precision. In this thesis, we present an overview of community detection techniques and we propose two new parallel implementations of the Louvain algorithm written in CUDA and exploitable by Nvidia GPUs: the first one is based on the sort-reduce paradigm with a pruning approach on the input data; the second one is a new hash-based implementation. Experimental analysis conducted on 13 datasets of different sizes ranging from 15 to 130 million edges shows that the proposed algorithms have different efficiency based on the size of the graph. For this reason, we study also an adaptive solution.

Contents

1 Introduction

2 Community Detection State of the Art

The problem of community detection raises in many application scenarios from the necessity of finding groups of objects that have a large number of connections to each other. To represent problems where it is fundamental to empathize connection between objects, the graph theory is the main tool. A graph is a mathematical structure composed of nodes (or vertices) that denote the objects and edges (or links) that express some kind of relationship between objects and possibly having a weights that quantifies this relationship.

The Graph Theory born in 1736 when Euler used this mathematical abstraction to solve the puzzle of Königsberg's bridges. Since then, this tool was used in several of Mathematics, Social, Biological and Technological application. In recent time, the approach to this studies has been revolutionized to deal with bigger and more complicated challenges, supported by the increasing computing power. The necessity of finding this high-connected substructure in graph arises from real problems of the previous field: for example, the study of Protein-Protein Interaction (PPI) networks is very important because the interaction between proteins is the basis of all process in the cell. A study demonstrated that this type of network shown to be useful for highlighting key proteins involved in metastasis. [5]

Other examples can be found in the field of sociology: a historically well-know scenario is the Zachary's Karate Club. This dataset captures members of a Karate Club for 3 years.[3] An edge between two nodes represents an interaction between two members outside the club. At some point, a conflict between the administrator and a master led to split of the club into two separate groups. The question is if it is possible to infer who compose these two new groups basing on the information that this graph give to us. This small network of 1977 is famous because it has often been used as a reference point to test the detection algorithms used to analyze huge social web networks. In general this kind of problem, i.e. clustering people that belong to the same community base on interaction, it's useful not only in sociology but also in marketing: by knowing people with similar interests, it's possible to make better recommendation systems.

There are several of similar scenarios to apply this method in the real-world, all united by the fact that the data are unregular but it's present some well-defined topological structure that in a completely random graph are absent. A random graph is a fully disordered graph, firstly proposed by Erdös and Rényi [1] in 1959: it's a graph where the probability that there is an edge between two nodes it's equal for all pairs of nodes and, for this reason, the degree of the nodes (i.e. the number of edges incident to a node) is homogeneous. In real networks, this is not true, because they are often scale-free (fallow a power-law distribution). An example of this is the study about the citations in scientific papers made by Derek J. de Solla Price in 1965 [2] or the study about World Wide Web growing made by Albert-László Barabási et al in 1999 [4]. Furthermore, the degree distribution of the nodes is non-homogeneous not only globally but also locally, this due to the observation that there is a high concentration of edges within sets of nodes and a low concentration of edges between this sets. These two concepts are essential to formulate the formal definition of Community and Modularity. In this chapter will be presented some definitions of community and will be given an overview of some methods that are used to identify communities.

2.1 Community Definitions

The informal definition of community is there are many more edges inside the community versus the rest of the graph, but there isn't a unique quantitative definition of community. This kind of freedom is necessary because the concept of community is strictly connected to the problem that will be analyzed: for example, in some cases, it's necessary that community overlap, but in other problems, this is not necessary. There is a unique key constraint that allows talking about community detection: the graph must be sparse. A sparse graph is a graph where the number of nodes has the same magnitude of the number of edges. In the unweighted graph case, if the number of edges is far greater than the number of nodes, the distribution of edges among the nodes is too homogeneous for communities to make sense [6]. In that case, the problem nature is little different: we aren't interested anymore on the edge density between nodes but we have to use some kind of metrics (like similarity or

distance) to clustering. In that case, the problem is more similar to data clustering. Despite this, assuming that a community is a subset of similar nodes it's reasonable, for this reasons some techniques (like spectral o hierarchical clustering) belonging to this field are adopted in community detection and will be shortly presented later on this thesis. Following this, Fortunato [6] defines three main classes of community's definitions: *local, global and based on vertex similarity*. Other types of definitions are still possible, but these three offers give a good summary of the problem. Now those classes will be presented to give an overview of the various approach that has been used to define this problem.

2.1.1 Local definitions

Considering that a community has a lot of interactions with the other nodes that are in it and few connections outside, it is fair to think about the communities as autonomous objects. The local definitions are based on this concept. Directly from this concept, we can think at the community as a clique, i.e. a subset whose vertices are all adjacent to each other. This type of definitions it's too strict: even if just one edge is not present, the subset is not a clique, but the subset has a very high concentration of edges. For this reason, the clique definition is often relaxed, using, for example, n-clique, i.e. a subset in which all the vertices are connected by a path of length less than n.

Anyway, this type of definitions ensure that there is a strong cohesion between the nodes in the subset, but not ensure that there isn't a comparable cohesion between the subset and the rest of the graph. For this purpose, other definitions were proposed. Given a graph $G(V, E)$, the relative adjacency matrix A and a subset of nodes C where $C \in V$, we define the internal degree k_v^{int} and the external degree k_v^{ext} for each vertex v that belongs to C as the number of edges that connect the node v with another node that belongs to C and not belongs to C, respectively:

$$k_v^{int} = \sum_{k \in C} A_{vk} \qquad\qquad k_v^{ext} = \sum_{k \notin C} A_{vk} \tag{1}$$

We also define the internal degree k_C^{int} and the external degree k_C^{ext} as the sum of all internal and external degree of nodes that belongs to C.

$$k_C^{int} = \sum_{i,j \in C} A_{ij} \qquad\qquad k_C^{ext} = \sum_{i \in C, j \notin C} A_{ij} \tag{2}$$

A strong community is a subset of nodes such that the internal degree k_n^{int} for each vertex n is greater than its external degree k_n^{ext} . This type of definitions once again very strict, for this reason we define as weak community a subset of nodes where the internal degree of the subset k_C^{int} is greater than its external degree k_C^{ext}. Many other variants of these definitions were presented in the literature.

2.1.2 Global definitions

The previous class quantify the community independently, considering every subset individually. Overturning the point of view, we can define communities in a graph-dependent way, considering them as an essential and discriminant part of it. There are many different interpretations of this approach in the literature, but the most important definitions are focused on this key fact: it's not expected to see a community structure in a random graph. For this reason, we define as *null model* of a graph another graph that have some features in common with the original one but it's generated randomly. This graph is used as a comparison term to identify if it's present a community structure in the graph or not and, if it is present, to quantify how it is pronounced. This approach, which is based the Modularity Optimization, is the main object of this study and is presented in detail in the next chapter.

2.1.3 Based on Vertex Similarity

The last class of definitions assumes that edges in the same community are similar to one another. All the definition used in the classic clustering methods belongs to this class because they calculate a distance (similarity) between object and aren't based on the edge density like the previous definitions. This distance can be calculated in various ways: if it is possible to embed the vertices into a n-dimensional Euclidean space by assigning a position to them, one method consists to calculate the distance between two nodes, considering that similar vertices are expected to be close to

each other. To calculate the distance, one could use a norm. Three norms often used in the literature are the following. Given two points $A = (a_1, ..., a_n)$ and $B = (b_1, ..., b_n)$ that belongs to the n-dimensional euclidian space E, we define the norms l_1 (Manhattan distance), l_2 (Euclidian distance) and l_3 (Maximum distance) as:

$$l_1(a, b) = \sum_{k=1}^{n} |a_k - b_k| \tag{3}$$

$$l_2(a, b) = \sum_{k=1}^{n} \sqrt{(a_k - b_k)^2} \tag{4}$$

$$l_3(a, b) = \max_{k \in [1,2]} |a_k - b_k| \tag{5}$$

Another option is the cosine similarity $\cos(a, b)$, that is very popular in literature:

$$\cos(a, b) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} (a_i)^2} \sqrt{\sum_{i=1}^{n} (b_i)^2}} \tag{6}$$

If it is not possible to embed the graph in a Euclidean Space, it is possible to infer the distance from the adjacency matrix. If it is not possible to embed the graph in a Euclidean Space, it is possible to infer the distance from the adjacency matrix. One idea is to map the distance in order to assign smaller values at nodes with the same neighbourhood. Given an adjacency matrix A we define the distance between two nodes a and b as:

$$d(a, b) = \sqrt{\sum_{k \neq a,b} (A_{ak} - A_{bk})^2} \tag{7}$$

Many other variants of that definition (but based on the same principle) were presented in the literature, for example considering the overlap between neighbourhood respect to the union.

Other alternative measures consider the number of independent paths between nodes, i.e. path that does not share any common edges, or they are based on random walk on a graph: for example, the average number of steps needed to reach one vertex from another by a random walker.

2.2 Community Detection Algorithms

We now present some techniques used in the field of community detection: graph partitioning methods, hierarchical clustering, spectral clustering. Moreover, the

Girvan and Newman algorithm will be presented in the next chapter: this is because this method firstly introduced the modularity function and it is presented separately. The goal of this chapter is to give a useful overview in order to get the differences with the Modularity optimization and empathize the motivations that make the Louvain algorithm one of the most used nowadays. For this reason, all the methods that are presented in this thesis find non-overlapping community, as the Louvain methods. For the sake of completeness, we remark that in Fortunato's report [6], that was mainly used to write this chapter, is also present an analysis of those overlapping algorithms.

3 Modularity Optimization

Historically, the modularity function Q was introduced as a stop criterion for the Girvan and Newman algorithm in 2002. This is a quality function, i.e. a function that allows distinguishing from a "good" cluster and a "bad" one. The function assigns to a partition a score that is used to compare partitions. This is not a trivial goal, because define if a partition is better than another is an ill-posed question: the answer may depend on the particular concept of community that it is adopted. Nevertheless, this sometimes is necessary, for example in the case of hierarchical clustering, where it's necessary to identify the best partition in the hierarchies. A simple example of this kind of function are the sum of the difference between internal degree k_v^{int} and the external degree k_v^{ext} [2.1.1].

The modularity function became very popular and a lot of methods based on this quality function were created. In this chapter we present the functions and its limits in details, the algorithm in witch it was firstly used, some optimization techniques based on modularity and indeed the Louvain algorithm, that is the main subject of this thesis.

3.1 Function

The function is based on the idea that we did not expect to see a graph structure in a random graph. We define as a *null-model* of a graph another one that it's generated randomly keeping some structural proprieties of the original one. Comparing the graph with its null model, we can quantify how much is well defined the community structure. Therefore, the modularity function is dependent on the choice of the null model. Given a graph $G = (V, E)$, a partition of nodes C and a function $c(x)$ that assign each nodes x to its community, we define a generic modularity function as :

$$Q = \frac{1}{2|E|} \sum_{i,j \in V} (A_{ij} - P_{ij}) \delta(c(i), c(j)) \tag{8}$$

where A is the adjacency matrix of G, P is the matrix of expected number of edges between nodes in the null model and δ is an filter function: its yields one if $c(i) == c(j)$, zero otherwise.

In principle, the choice of a null model is arbitrary, but we have to consider carefully the graph properties to keep in the null model because they determine if the comparison is fair or not. For instance, it's possible to choose as a model that keeps only the nodes and edges numbers, assuming that an edge is present with the same probability for each pair of nodes (in this case P_{ij} is constant). For this reason, The standard null model of modularity imposes that the expected degree sequence(after averaging over all possible configurations of the model) matches the actual degree sequence of the graph [6]. In this scenario, the probability that two vertices i and j are connected by an edge is equals to the probability to get two stubs (i.e. half-edges) incident to i and j.

This probability p_i of piking a stub from the nodes i is $\frac{k_i}{2|E|}$ where k_i is the degree of nodes i. The probability that two stub joining is $p_i p_j = \frac{k_i k_j}{4|E|^2}$. Therefore, the expected number $P_i j$ of connections between the nodes i and j is:

$$P_{ij} = 2m p_i p_j = \frac{k_i k_j}{2|E|} \tag{9}$$

Replacing P_{ij} from (9) in (8) we obtain:

$$Q = \frac{1}{2|E|} \sum_{i,j \in V} (A_{ij} - \frac{k_i k_j}{2|E|}) \delta(c(i), c(j)) \tag{10}$$

that is the standard modularity function.

3.2 Resolution Limit

3.3 Girvan and Newman algorithm

3.4 Modularity Optimization Techniques

3.4.1 Greedy Techniques

3.4.2 Extremal Optimization

3.4.3 Simulated Annealing

3.4.4 Spectral optimization

3.5 Louvain Algorithm

3.5.1 Parallel Implementations

4 Nvidia GPUs architecture and CUDA

5 Implementation

5.1 Overview

5.2 Fast-Move Pruning

5.3 Sort-Reduce Pattern

6 Performance and Analysis

7 Conclusion

References

[1] A.Rényi P.Erdös. "On Random Graphs". In: *Publ. Math. Debrecen 6* (Dec. 1959), pp. 290–297.

[2] Derek J. de Solla Price. "Networks of Scientific Papers". In: *Science* 149.3683 (1965), pp. 510–515. ISSN: 0036-8075. DOI: 10.1126/science.149.3683.510. eprint: https://science.sciencemag.org/content/149/3683/510.full.pdf. URL: https://science.sciencemag.org/content/149/3683/510.

[3] W.W. Zachary. "An information flow model for conflict and fission in small groups". In: *Journal of Anthropological Research* 33 (1977), pp. 452–473.

[4] Albert-László Barabási and Réka Albert. "Emergence of Scaling in Random Networks". In: *Science* 286.5439 (1999), pp. 509–512. ISSN: 0036-8075. DOI: 10.1126/science.286.5439.509. eprint: https://science.sciencemag.org/content/286/5439/509.full.pdf. URL: https://science.sciencemag.org/content/286/5439/509.

[5] Pall F. Jonsson et al. "Cluster analysis of networks generated through homology: automatic identification of important protein communities involved in cancer metastasis". In: *BMC Bioinformatics* 7.1 (Jan. 2006), p. 2. ISSN: 1471-2105. DOI: 10.1186/1471-2105-7-2. URL: https://doi.org/10.1186/1471-2105-7-2.

[6] Santo Fortunato. "Community detection in graphs". In: *Physics Reports* 486.3-5 (Feb. 2010), pp. 75–174. ISSN: 0370-1573. DOI: 10.1016/j.physrep.2009.11.002. URL: http://dx.doi.org/10.1016/j.physrep.2009.11.002.