Louvain Community detection in a *Caveman* graph

July 24, 2017

Alaa BAKHTI (bakhti_a)

alaa.bakhti@epita.fr

Abstract

In this project, we tried to generate a very large Caveman graphs and thandetermine the community detection time of each graph. Louvain algorithm was used for the community detection.

Contents

1 Introduction

We start by defining some basic properties in graph theory such as *a clique, Characteristic Path Length, Clustering Coefficient and a Community* in a graph. Than we present the used features in the experiments. These features include *Caveman graph and Louvain community detection algorithm*. Finally we present the expreminents steps (Caveman graph generation, ...) and the results (Community detection execution time).

2 Basic Properties

2.1 Graph Clique

In the mathematical area of graph theory, a clique is a subset of vertices of an undirected graph such that every two distinct vertices in the clique are adjacent; that is, its induced subgraph is complete. [1]

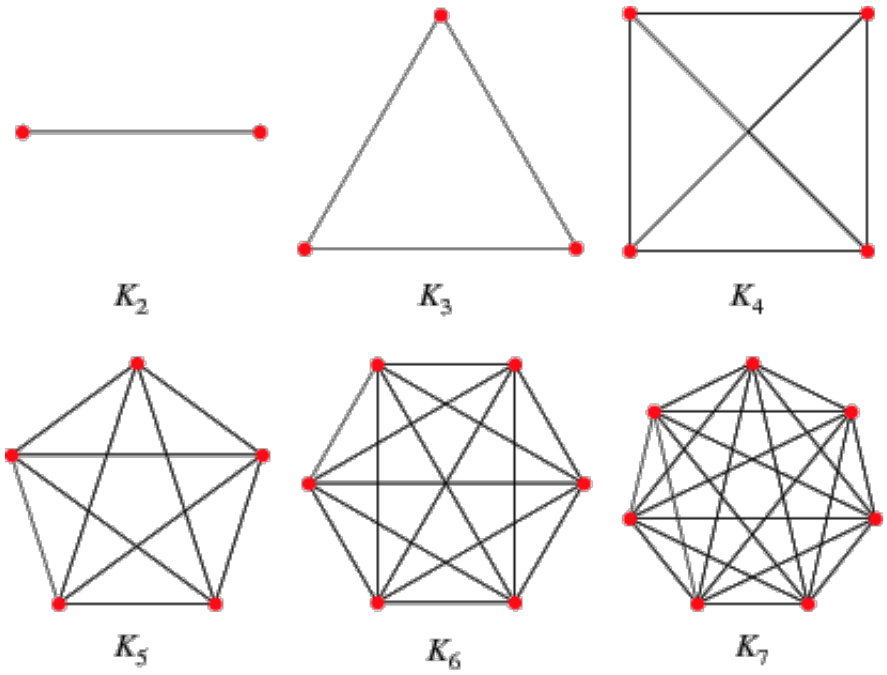

Figure 1: Example of cliques [3]

2.2 Characteristic Path Length (L)

Characteristic Path Length is simply defined as the average distance between any two nodes in the graph, or more precisely, the average length of the shortest path connecting each pair of nodes. All edges are unweighted and undirected, so the distance between nodes is simply the minimal number of edges that must be traversed to get from one node to the other. L is generally a measure of the global properties of a graph and, as we will see, can be quite independent of local structure. [2]

Characteristic Path Length is only a meaningful measure if a graph is fully connected – i.e. if there is a sequence of edges joining any two nodes.

2.3 Clustering Coefficient (C)

Clustering Coefficient is a measure of how clustered, or locally structured, a graph is. The Clustering Coefficient is an average of how interconnected each node's neighbors are. Specifically, if a node i has ki immediate neighbors, then we can define the Clustering Coefficient for the node i, Ci, as the fraction of the total possible ki*(ki-1) / 2 connections that are realized between i's neighbors. Loosely speaking, the Clustering Coefficient measures how close each agent's neighborhood is to a clique. [2]

2.4 Community

In the context of networks, community structure refers to the occurrence of groups of nodes in a network that are more densely connected internally than with the rest of the network. This inhomogeneity of connections suggests that the network has certain natural divisions within it.

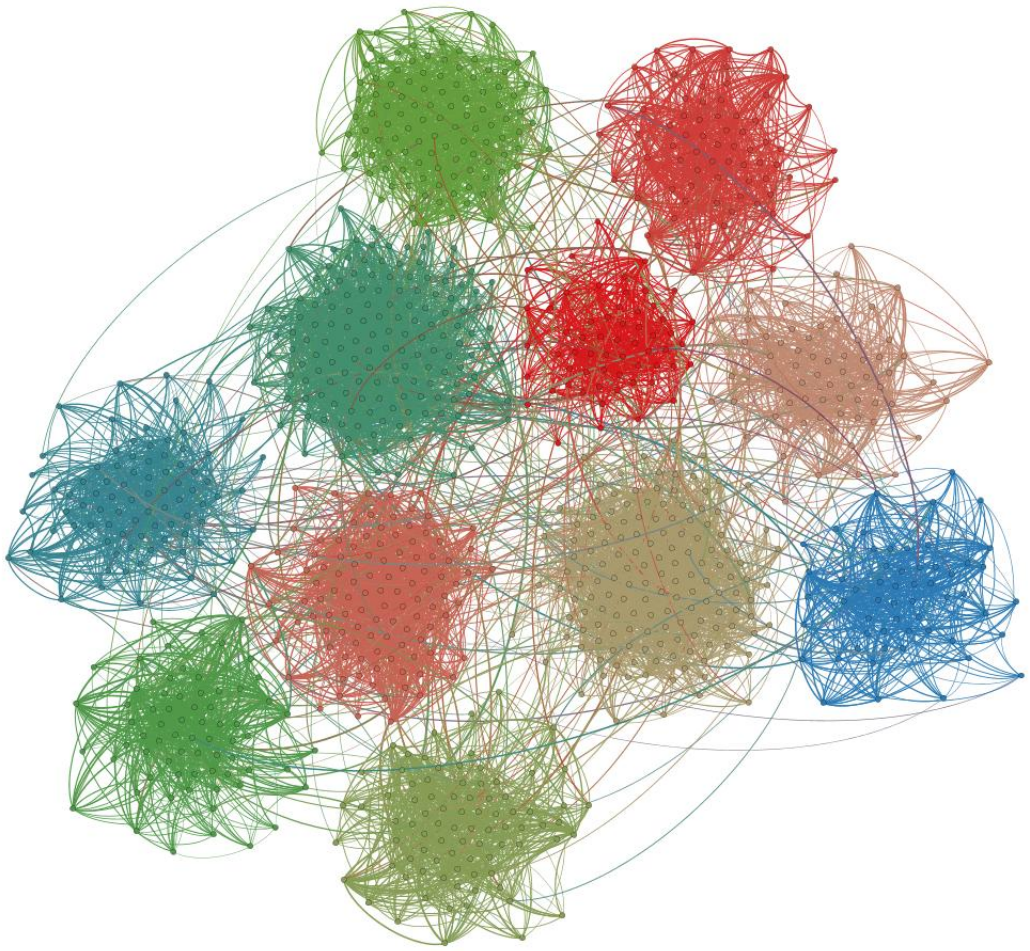

Figure 2: Example of communities in a graph [5]

3 Features

In this section, we will define the Caveman graph and the Louvain algorithm.

3.1 Caveman graph

The (connected) caveman graph is a graph arising in social network theory formed by modifying a set of isolated k-cliques (or "caves") by removing one edge from each clique and using it to connect to a neighboring clique along a central cycle such that all n cliques form a single unbroken loop (Watts 1999). A number of cavemen graphs formed in this manner from K_5-e are illustrated above. [3, 4]

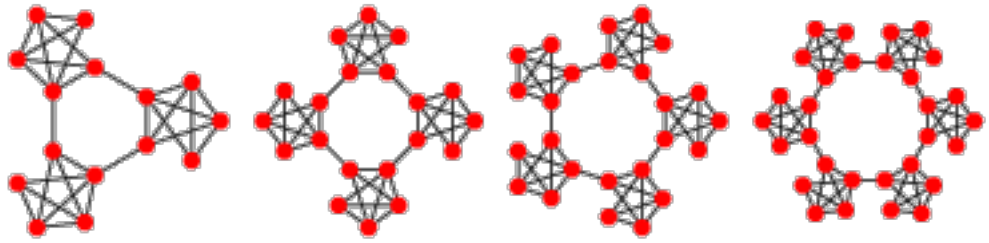

Figure 3: Example of ccaveman graphs [3]

The characteristic path length of a connected caveman graph is on the order of K/k where K represents the number of caves and k represents the number of connections a node has. Hence, as K gets large and the graph remains sparse, Characteristic Path Length (L) grows steadily. The caveman graph represents one type of extreme graph – high clustering coefficient and large characteristic path length. [2]

3.2 Louvain algorithm

The Louvain Method is a method for identifying communities in large networks. It was created by Vincent Blondel, Jean-Loup Guillaume, Renaud Lambiotte and Etienne Lefebvre. The method is a greedy optimization method that appears to run in time O(n log n). It has been used with success for networks of many different type and for sizes up to 100 million nodes and billions of links. The analysis of a typical network of 2 million nodes takes 2 minutes on a standard PC. The method unveils hierarchies of communities and allows to zoom within communities to discover sub-communities, sub-sub-communities, etc. It is today one of the most widely used method for detecting communities in large networks.
The original idea for the method is due to Etienne Lefebvre who first developped it during his Master thesis at UCL (Louvain-la-Neuve) in March 2007.

The inspiration for this method of community detection is the optimization of Modularity as the algorithm progresses. Modularity is a scale value between -1 and 1 that measures the density of edges inside communities to edges outside communities. Optimizing this value theoretically results in the best possible grouping of the nodes of a given network, however going through all possible iterations of the nodes into groups is impractical so heuristic algorithms are used. In the Louvain Method of community detection, first small communities are found by optimizing modularity locally on all nodes, then each small community is grouped into one node and the first step is repeated. [1, 6]

4 Experiments

4.1 Caveman graph generation

A graph is written in a text file using this format.

- The first line in the file contains the number of nodes in the graph.
- The following lines contains the connection between nodes. If a node a is linked to the nodes b and c. Than in the file we will write these 2 lines (a b, a c) that presents the connections of the node a.

To generate a Caveman graph with K caves and k nodes in a cave, we follow 2 steps:

- Create the caves: Create K caves and link all the nodes in the cave except the 2 final nodes.
- Link the caves between each other: Use the 2 final nodes in each cave to link the current cave with the previous cave and the cave after.

4.2 Community detection

The louvain community detection algorithm was used for this purpose. The source code that can be found here [7].

4.3 Results

A script was created that takes as argument:

- –nbr_graphs=X : The number of graphs to generate is X.
- –cave_size_start=X : The starting size of the cave size is X.
- –nbr_caves_start=X : The starting number of the caves is X.
- –cave_size_step=X : The step used to get the next cave size is X.
- –nbr_caves_step=X : The step used to get the next number of caves is X.

This script do followings:

- Generates a graphs with all the combinations between the cave size (k) and the number of caves (K).
- For each graph, determines the time needed to detect the communities using Louvain algorithm.
- Saves the results (K, k, exec_time) in a file "results.txt" in the root directory of the project.

To do the benchmark I used a K $\in$ [1..250] and k $\in$ [1..1000]. The following figure shows the results.
We notive that the time Louvain algorithm takes to detect comminities follows an exponential growth as K increases.

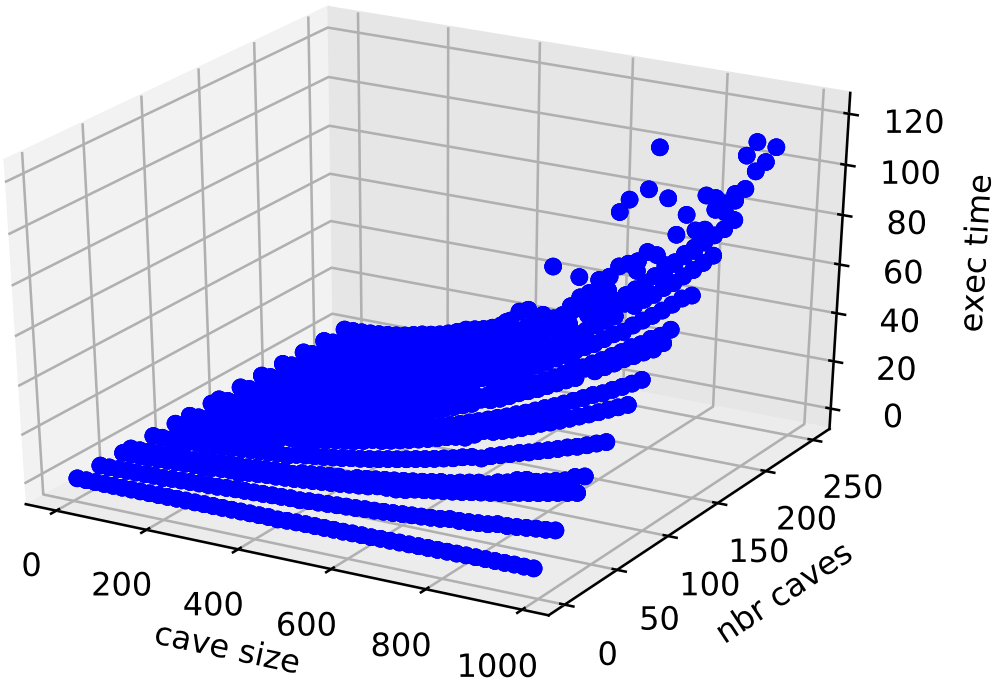

Figure 4: Execution time of Louvain algorithm

5 Conclusion

After the generation of Caveman graphs with different values for the cave size and the number of caves, we notices that Louvain algorithm follows an exponential growth as the number of caves increases.

References

[1] Wikipedia https://en.wikipedia.org

[2] Indigosim http://www.indigosim.org/tutorials/smallworld/t2s1.htm

[3] Wolfram Math World http://mathworld.wolfram.com

[4] Watts, D. J. Small Worlds: The Dynamics of Networks between Order and Randomness. Princeton, NJ: Princeton University Press, 1999. http://press.princeton.edu/titles/6768.html

[5] Stackoverflow

[6] Fast unfolding of communities in large networks https://arxiv.org/abs/0803.0476

[7] Louvain source code https://sites.google.com/site/findcommunities/

List of Figures