A study of Core Periphery Algorithms

Social Network Analysis for Computer Scientists — Course paper — Group 14

Michail Athanasios Kalligeris Skentzos
s4398831@umail.leidenuniv.nl
LIACS, Leiden University

Ngoc Xuan Y Nguyen
s4372840@umail.leidenuniv.nl
LIACS, Leiden University

ABSTRACT

This should contain summary of our paper after we finish
Context: core periphery in networks, interesting because it allows to gain unique understanding of the.
Problem: overlapping cores + how do we visualize them properly
Solution: hierarchical core periphery paper if it works, ?
Data: Karate Club, hiv transmission
Experimental Results:

KEYWORDS

core periphery, hierarchical core periphery, social network analysis, network science

ACM Reference Format:
Michail Athanasios Kalligeris Skentzos and Ngoc Xuan Y Nguyen. 2024. A study of Core Periphery Algorithms: Social Network Analysis for Computer Scientists — Course paper — Group 14. In *Proceedings of Social Network Analysis for Computer Scientists Course 2024 (SNACS '24).* ACM, Leiden, NY, The Netherlands, 3 pages.

INTRODUCTION

In this paper we discuss the results of our group project for our course Social Network Analysis for Computer Scientists in Leiden University, 2024. The project was based on a study of the paper: Hierarchical core-periphery structure in networks by Polanco et. al. In our project we experimented with various core periphery algorithms apart from this one and present the results. More specifically in section 1 we give a quick introduction of the background of the topic followed by a review of the related work in section 2. In section 3 we present the core idea of the Hierarchical core-periphery algorithm for both a fixed and varying number of groups. Should we perhaps combine section 4, 5 and 6 into a cohesive Method section with the relevant subsections?In section 4 we present our own approach to evaluating these algorithms in order to compare the results followed by section 5 where we discuss our datasets and 6 where we describe our experiments. Finally in section 7 we discuss the findings of our

This note briefly explains the theory behind the method indicated in Paper [**?**]. Then, various experiments on the intended dataset would be performed on the aforementioned method, as well as three other older approaches for performance comparison and researching purposes. Some uncertainty about the core-periphery methodology that still persists nowadays would be shortly discussed at the end.

SNACS '24, Master CS, Fall 2024, Leiden, the Netherlands

1 BACKGROUND

The core periphery task is a task closely related to the well known community detection task. In traditional community detection given a graph the goal is to find a set of groups (communities) from the information about the nodes (initially just the edges) so that nodes within the same group are densely connected or have similar characteristics compared to nodes in different groups. The core periphery task also looks for groups but in a manner that it finds a group of densely connected nodes that is considered to be the core among the periphery group of more sparsely connected nodes. Many algorithms have been proposed throughout the years improving the tasks themselves to consider weighted edges, node attributes, try to estimate overlapping groups and capture hierarchical structure. Although community detection is widely recognized as an interesting task that yields usefull results, core periphery is considered to be of lesser importance and is often cosidered to be an edge case of community detectionMUST FIND SOURCES FOR THIS. In this paper we focus on core periphery task and more specifically hierarchical core periphery. In the next section we present related work in this field that led to the proposal of this algorithm.

2 RELATED WORK

Many core periphery algorithms have been proposed through the years with the concept first being introduced Raul Prebisch in 1950 to understand the economy of latin america considering the core to be the economically developed center and the periphery the periphery the undeveloped regions. then by John Friedmann in 1966 to understand urban development having the city as the core and the villages as the periphery in a growing environment. These models provided a mental concept of the world so as to study the various ways they might affect economic and urban growth but were only kept to a conceptual stage not applied on graphs.
ADD REFERENCES The first instance of a paper discussing core periphery structure in graphs is probably Models of core/periphery structures by Borgatti and Everett in 1999. In this paper an ideal core periphery is proposed where all the core nodes are connected among themselves and the periphery but none in the periphery are connected among themselves. This allows the algorithm to only be able to capture a core group and a periphery group as it tries to categorize a node as either core or periphery. This concept was improved by Alvarez et al proposing the k-core decomposition algorithm in 2006 that introduced a sliding scale based on node degree that categorized nodes as core or periphery.
The ability to capture more than two groups was introduced by

Kojaku and Masuda who in 2017 proposed a quality function to find nested cores in a network. This concept was also adopted by Gallagher et al in 2021 who applied Stochastic Block Models to capture a nested core periphery structure. Finally Polanco et. al proposed a new approach in 2024 in which they used a monte carlo method to enable finding overlapping core-periphery groups while also capturing a hierarchical structure. As this algorithm was the main focus of our group project it is presented in the section below.

3 HIERARCHICAL CORE-PERIPHERY

The Hierarchical core-periphery algorithm takes an undirected, unweighted graph and through a monte carlo group selection process yields a set of groups that represent the core structure of the graph. In the initial state all nodes belong to group 0 and each new state is proposed by picking a random group and adding or removing a random node. The likelihood of transitioning to that state is calculated according to the ratio of the probabilities of the previous and next state. By repeating these steps long enough the algorithm converges into the final most probable state. The purpose of this section is to discuss the hierarchical core periphery algorithm in detail.

3.1 Proposed Monte Carlo Scheme - fixed k

make pseudocode from our presentation notes Monte Carlo scheme - make pseudo code

- choose group uniformly
- choose to remove or add node 50-50
  - If Remove
    - choose uniformly at random from current group
    - if no nodes, continue
  - If Add
    - choose uniformly at random from nodes not in current group
    - if group is full, continue
- accept move with Metropolis-Hastings style acceptance probability - fancy way of saying fraction of probability between new state and previous state (obviously maxed at 1)

3.2 Proposed Monte Carlo Scheme - varying k

- explain type 1 and type 2 moves
- explain altered monte carlo for varying k - use pseudocode
  - choose move type
  - if 1
    - choose group number uniformly
    - **Add group** at index and update all upper group memberships
  - if 2
    - choose group uniformly
    - choose to remove or add node 50-50
      - If Remove:
      - choose at random from current group
      - if no nodes, **Delete group**
      - If Add:
      - choose randomly from nodes not in current group
      - if group is full, continue
- Finally accept move with Metropolis-Hastings style acceptance probability - fancy way of saying fraction of probability between new state and previous state (obviously maxed at 1)

Process: With probability 1/2k(n+1) choose Type 2 else Type1 Calculate Likelihood Accept or reject the move with probability a

3.3 Calculation of Graph probability

Give an explanation and understanding only on how the probability of the graph is calculated. The proof of the monte carlo ergodicity and detailed balance are not of importance. We need the reader to understand the calculation of the probability for each state so that the likelihood of the next state is understood. Also we are using this as a metric so it's properties will be better understood if it's explained here.

- Network probability

$$P(A \mid \omega, k, g) = \prod_{u<v} \omega_{h(u,v)}^{a_{uv}} \left[1 - \omega_{h(u,v)}\right]^{1-a_{uv}} \tag{1}$$

$$= \prod_{r=0}^{k-1} \omega_r^{m_r} (1-\omega_r)^{t_r - m_r} \tag{2}$$

- Network probability over number of groups k and node participation in groups g

$$\begin{aligned} P(A \mid k, g) &= \int P(A, \omega \mid k, g)\, d\omega \\ &= \int P(A \mid \omega, k, g) P(\omega)\, d\omega \\ &= \prod_{r=0}^{k-1} \int_0^1 \omega_r^{m_r} (1-\omega_r)^{t_r - m_r}\, d\omega_r \\ &= \prod_{r=0}^{k-1} \frac{m_r!(t_r - m_r)!}{(t_r+1)!}. \end{aligned}$$

where t_r is the total number of nodes in the group and m_r is the total number of connected nodes in the group

$$t_r = \sum_{u<v} \delta_{r,h(u,v)}$$

$$m_r = \sum_{u<v} a_{uv} \delta_{r,h(u,v)}$$

4 APPROACH

Perhaps we dont need to explain the studying part of the approach. just the selection of the algorithms and dataset, the blocking issues with the code maybe? and then our experimentation process

- studied hcp paper and followed it's references
- studied relevant papers
- Blocking issue: code not compiling: had to manually pick files from previous commit to make make work (with indexed lists not mvectors)
- find datasets & transform in gml format
- run hcp & older algorithms on datasets
- result evaluation - graph probability as a metric for traditional core periphery structure
- visualize results - potential blocking issue multiple cores - Frank suggested the plugin from this article: https://medium.com/@vespinozag/position-ranking-for-gephi-563bb9deb0bf

5 DATA

In this section we should discuss the data, what they represent (nodes, edges) and what structure we aim to capture from them

- terrorist: https://networks.skewed.de/net/terrorists_911
- urban streets: https://networks.skewed.de/net/urban_streets

6 EXPERIMENTS

Populate this while doing the experiments so as to not lose track

- traditional core periphery
- k-decomposition
- stochastic block models (sbm)
- Hierarchical core-periphery (hcp)

As there is no accepted metric to evaluate core periphery we aim to evaluate the graph likelyhood proposed in Hierarchical core periphery as a metric to assess the performance of the above algorithms in the traditional 2 group core periphery problem. Then we will try to do the same for sbm vs hcp for multiple groups

7 CONCLUSION

ACKNOWLEDGMENTS

REFERENCES