C-Explorer: Project Plan

Student: Kang Yunfan

Supervisor: Reynold Cheng

1. Background

Social networks have become increasingly popular and important in the present society. It can be modeled as large attributed graphs with vertices representing users and edges between vertices representing the friendship between two users. Each vertex has a set of keywords which are associated with the properties of that user (such as the user's interests or geographical location). Subgraphs whose vertices are densely connected suggests that this group of users share some common interests or features and we call this kind of subgraphs communities. Communities are extremely useful for some real-life applications such as event organization and advertisement injecting. A few CR algorithms have been developed but they are usually tested on limited number of datasets. The goal of this project is to develop a program to enables researchers to visualize the result of their CR algorithm. Meanwhile, it also provides APIs for researchers to plug in different datasets for testing purpose and compare different CR algorithms in detail. Furthermore, further research will also be carried out based on the existing ACQ CR algorithm to further enhance its performance.

2. Related works

There are some related works concerning graph queries. Fan et al. provided *ExpFinder* in [1] which uses graph pattern matching to find experts in social networks. In [2], Yi et al. introduced *AutoG* that facilitate users to iteratively compose graph queries by giving top-K queries suggestions. *VIIQ*, as is presented in [3], enables users to construct queries through interactive and iterative visual query formulation interface. However, as is pointed out in [4], these systems are designed to facilitate general graph queries but implementing CR algorithms by simple graph queries is indirect. Hence, we plan to develop *C-Explorer*, a web-based platform for users to formulate queries for community retrieval and it should also be easy to be extended in to real-life applications.

3. Objective and Features

- C-Explorer

The project is aimed to accomplish two objectives, the first one is to build *C-Explorer* program and the second objective is to do further research to enhance the existing community search algorithm with the help of the analysis function of C-Explorer.

C-Explorer is an open source web-based program. The program basically provides two functions for the general users.

The first function is community search. As is shown in the figure 1.1, user can formulate a query by adding query names, selecting keywords and the minimum degree. User can add query names by inputting a name and click the "+" button. Keywords sets for each query name will be loaded when

the name is added and the union of the sets all query names will be calculated and displayed as the keyword candidate list (the check list below the title keywords). New keywords can also be added to the keyword candidate list. After formulating the query, user can click on the “Submit” button to send the query to the server, and the server will return communities generated by the CR algorithm and the communities will be displayed in the right panel. After the display of the communities, functions that may facilitate the user to better explore the returned community are provided and user can click on the buttons or icons below the display of the community to use them. To restart a new query, user can click on the “Reset” button.

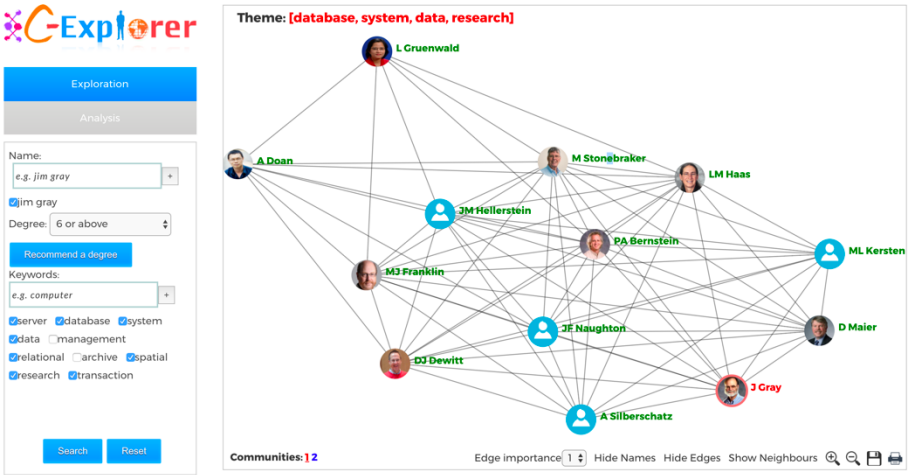

Figure 1.1 Query formulation and display of the communities

Furthermore, when a certain vertex is clicked, the profile of that people will be displayed (Figure 1.2). User can choose to start a new search to explore the communities of this people by clicking on the “Explore” button.

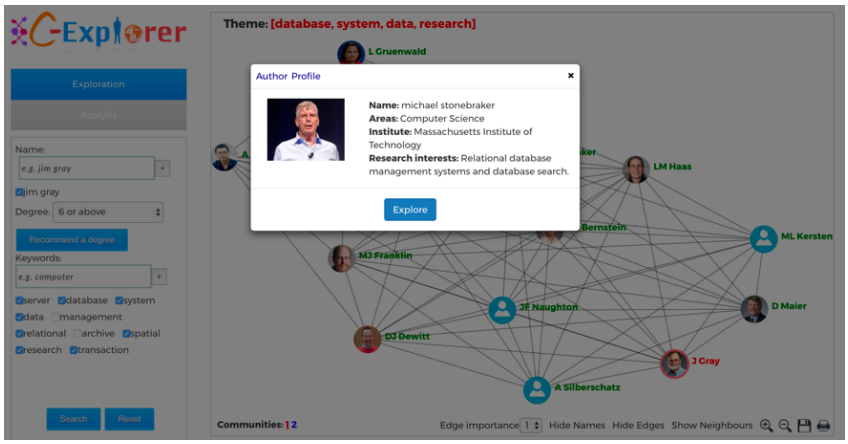

Figure 1.2 Explore a certain vertex

Secondly, a module for comparing the performance of different CR algorithms is provided (Figure 2.1). Query users are enable to compare communities retrieved using different CR algorithms by specifying the query name and keywords and click the “Compare” button. Two metrics measuring the average similarity over all pairs of vertices (CPJ) and the average frequency of keywords in the

keyword sets for all the vertices in the community (CMF) are proposed. Usually, communities with better cohesiveness will score higher in these two metrics. Meanwhile, statistics of the performance of different algorithms are provided. Communities retrieved by a certain algorithm will be displayed in a new window once the "view" link of that algorithm is clicked.

Researchers can use this platform to visualize the result of their algorithm as well as make comparisons with other algorithms. To facilitate this purpose, APIs will be provided so that dataset and algorithm used in these two functions can be replaced.

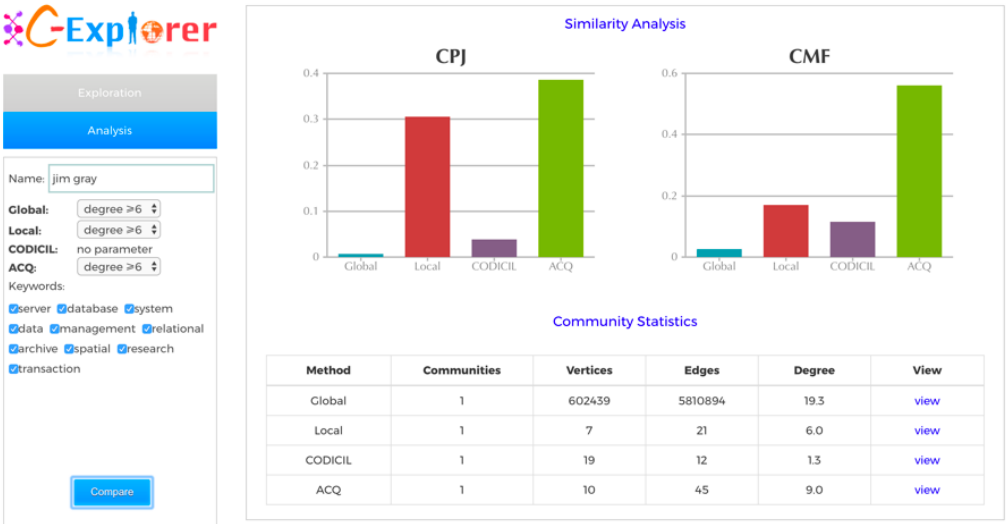

Figure 2.1 User Interface of "Analysis"

- **Algorithm enhancement**

Besides of the program, this project also aims to enhance the existing ACQ algorithm, an attributed community search algorithm raised by Yixiang Fang. Two directions will be explored in this part. The first one is to further explore the possibility of retrieving the community of highest degree with the constraint that the number of keywords shared by each vertices of that community is at least n (MaxKMinN). Secondly, instead of just consider the number of overlapping keywords between vertices, other similarity functions should be implemented and tested to see if we can further improve the performance of the algorithm, which can be demonstrated by the analysis function of *C-Explorer*. The deliverable for the algorithm part would be short essays or reports.

4. Methodology

- **Overview of C-Explorer and techniques used**

C-Explorer is a browser-server architecture program. JSP and Tomcat are used to implemented it. The program is originally designed to Demo the ACQ algorithm and the algorithm is written in Java. To facilitate this purpose, it is more straightforward to use JSP to implement the program. Meanwhile, the object-oriented feature of Java also makes it easy to provide interfaces for other users to plug in their own algorithms and take full advantage of this program. Besides, Java Servlet Technology is used to implement the back-end logic. The program will be deployed on the server provided by the CS department.

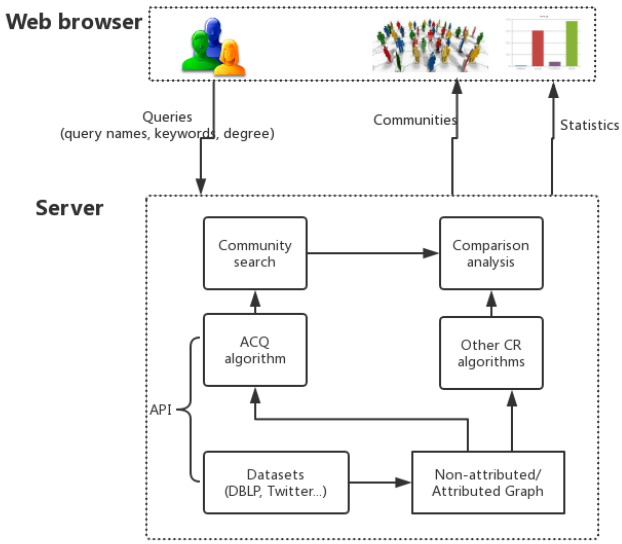

Figure 3 The framework of C-Explorer

An overview of the framework is shown in *Figure 3*. The browser side provides interfaces for users to formulate queries, view communities and analysis.

Queries issued by the *Exploration* page (Figure 1.1) are sent to the *Community Search* module on the server side. The module calls the ACQ algorithm to retrieve and return to the browser a set of communities. The communities are displayed using Scalable Vector Graphics (SVG) at the front end. SVG is an XML-based vector image format and it helps to achieve features such as zoom-in and zoom-out to facilitate users to better view the communities.

For the *Analysis* page (Figure 2.1), its queries are received by the *Comparison analysis* module. The module analyzes the quality of communities found by different CR algorithms and return the result and the statistics to the browser. The comparison results are shown using the chart and the statics will be put into a table. The two charts (CPJ and CMF) are drawn using ConvasJS, which provides beautiful themes and simple API to display rich content and make the charts interactive.

Test cases will be design to test the performance and the stability of the system when multiple queries are send to the server at the same time. After the system is made online, user acceptance test will also be carried out by asking about 10 researchers to try this platform and call the APIs.

- **Methodology for the algorithm**

Dataset used is based on the XML files released by DBLP*. Raw data is processed to retrieve the whole graph and the keywords of each vertex. The graph is stored using adjacency matrix and the keywords are stored using linked lists. Currently the preprocessing of data has been done for the DBLP dataset. More datasets will be included with the assumption that methods can be designed to extract the graph structure and the keywords.

ACQ algorithm will be extended to solve the MaxKMinN problem. There might be some mathematical prove of the correctness of the algorithm. Different similarity functions to evaluate the similarity

between vertices will be designed. The comparison of the similarity functions will be made based on CMF and CPJ and can be visualized using the "Analysis" module of the C-Explorer program.

5. Schedule

1 October 2017	Deliverables of Phase 1 (Inception) • Detailed project plan • Project web page
22 October 2017	Solve Efficiency problem of the program and deploy the program with ACQ embedded online
November 2017	Design and testing of APIs for researchers to plugin algorithms and datasets; Do literature collection and review of CR algorithms.
December 2017	Learn about key concepts and related algorithms in CR algorithms.
8-12 January 2018	First presentation
21 January 2018	Deliverables of Phase 2 (Elaboration) • Preliminary implementation • Detailed interim report
January – February 2018	Design and test the algorithm
March – 15 April 2018	Combine the algorithm with the program for demonstration, write reports about the algorithm.
15 April 2018	Deliverables of Phase 3 (Construction) • Finalized tested implementation • Final report
16-20 April 2018	Final presentation
2 May 2018	Project exhibition

Reference:

[1] W. Fan, X. Wang, and Y. Wu. Expfinder: Finding experts by graph pattern matching. In *ICDE*, pages 1316–1319, 2013.

[2] P.Yi,B.Choi,S.S.Bhowmick,andJ.Xu.Autog:Avisualquery autocompletion framework for graph databases. *PVLDB*, 9(13):1505–1508, 2016.

[3] N. Jayaram, S. Goyal, and C. Li. Viiq: auto-suggestion enabled visual interface for interactive graph query formulation. *PVLDB*, 8(12):1940–1943, 2015.

[4] Y. Fang, R. Cheng, S. Luo, J. Hu, K. Huang. C-explorer: browsing communities in large graphs. *Proceedings of the VLDB Endowment*, *10*(12), 2017.