Exploring the Interconnections of Programs, Departments, and Colleges at Utah State University through Graph Analysis

Bennett DenBleyker
Computer Science
Utah State University
Logan, UT
bennett.denbleyker@usu.edu
0000-0002-1369-6700

Bradley Leavitt
Computer Science
Utah State University
Logan, UT
A02082468@usu.edu

Rachael Fisher
Computer Science
Utah State University
Logan, UT
lA02022226@usu.edu

***Abstract*— This study shows the similarity between programs, departments, and colleges at Utah State University. With the data collected from the university's website, connections were made between the different programs and departments. Data was collected through the university's website and registrar. After the data was obtained and preprocessed a graph with all of the departments and associated programs and courses was created. Using a variety of algorithms like Jaccard, SimRank, and Panther Similarity, and Common Neighbor Centrality as well as the Louvain algorithm, we were able to analyze how similar the programs were at USU and how the majors and minors related to each other and to specific programs. And if the department of Computer Science has more connections with the College of Science or the College of Engineering. It**

Keywords—Graph Analysis, Educational, Course Network, Bipartite

I. INTRODUCTION

Utah State University (USU) is a university located in Logan, Utah. It has a rich history dating back to the late 1800s and has grown significantly over the years, now boasting 8 colleges, 52 departments, 165 majors, 647 programs, and almost 7,000 courses. With such a large and diverse academic offering, it can be challenging for students and faculty to understand the relationships and connections between different programs, departments, and colleges at USU.

In this research paper, we aim to explore the interconnections between programs, departments, and colleges at USU through the use of graph analysis. This will be done by creating a graph representation of the academic offerings at USU, with nodes representing colleges, departments, programs, courses, and course sections, and edges representing relationships between these entities. Through this analysis, the goal is to answer several important questions, including: (1) How similar are the distinct programs at USU? (2) How do majors and minors relate to each other and to specific programs at USU? (3) Is the placement of the computer science (CS) program at USU appropriate, and does it have more connections to the College of Science or the College of Engineering?

There has been limited work in this area, particularly at the level of granularity that we are examining. Previous research has focused on analyzing the relationships between courses and students but has not delved into the connections between programs and courses from every department and college. Additionally, there are challenges associated with data collection and analysis, including the need to comply with the Family Educational Rights and Privacy Act (FERPA) and the availability of reliable and up-to-date data. In this paper, these challenges will be addressed and present a comprehensive analysis of the interconnections at USU.

II. RELATED WORK

Applying data analysis techniques to education research can be beneficial to universities and can improve the educational experience for students. Several studies have been done to analyze the courses and programs at universities. Slim and Kozlick (2014) studied a complex network to detect and quantify the cruciality of university courses. They also applied the framework to study the complexity of curricula within colleges, which could lead to optimizing curricula to improve a universities graduation rate. In their research, they focused on the courses and their prerequisites. Their framework utilized data from courses, terms, and students' GPA. They measured the cruciality of courses at a university, college, and department level.

Another example of research being done on courses and more specifically their prerequisites was research done by Hriez and Al-Naymat (2021). They analyzed the dependency relationships between Course Learning Outcomes of prerequisites and advanced courses. They built a model to predict students' performance in advanced courses based on prerequisites. Their models were based on bipartite graphs. Uriah and McKay (2020) analyzed campus structures and student experiences with a bipartite network made up of courses in the students who take them. With their student-course network, they were able to reveal how students and majors can be strongly connected or dispersed and how social network analysis can be used to improve understanding of the learning environment at a university.

All of the relevant research discussed only analyzed the relationship between courses and students. Although this paper does not analyze the relationship between students and majors, or students and courses, or pre-requisite courses, courses were analyzed with programs and bipartite graphs were used for the models as was done in the relevant research. The network that

was used for this project is unique because it studies the relationship between different programs and courses in different departments and colleges. It also includes data about sections which has not been done before. Also, the department of Computer Science and its associated courses were a focus of this paper and the programs similar to it were analyzed.

III. Methods

The goal of this research paper is to understand the relationships between different colleges, departments, programs, and courses at Utah State University. To achieve this goal, we have collected data on the various academic units at the university and used it to create a graph representation of these units. In this section of the paper, we will describe the methods used to collect and analyze this data, including the methods used to create the graph, the similarity scoring methods used to measure the relationships between nodes in the graph, and the community analysis methods used to identify clusters of nodes that are closely related to each other. These analysis methods include similarity scoring techniques such as Jaccard, SimRank, and Panther, and grouping algorithms such as the Louvain algorithm to identify communities within the graph. We also look at how changing the graph changes similarity *within* these communities to better understand the relationships between different academic units at the university.

A. Data Collection

Two groups of data were used in this study: one received from the USU Registrar and the other created by web scraping the USU Catalog.

The first dataset was received as a large Excel sheet from the Registrar's office, and helped show a correlation between courses taken and program membership. For each program with more than five people enrolled, it listed all courses taken in any term by at least two of them. It gave the exact number of students in the program, and the number of them that took each class.

The second group of data was made up of multiple datasets: programs data, course data, and section data. The programs data was acquired by scraping the Catalog's "Programs by College and Department" webpage,[1] which lists all programs along with their college, department, and program type. Each program is represented as an HTML anchor tag, linking to a program page. From each program page, we then grabbed all hyperlinks, and filtered for those linking to courses, or creating popups describing them. We took the course IDs from these tags, but because courses were linked to or described in different ways for each program, it was difficult to programmatically determine the difference between elective, option, and required classes.

To obtain the course names and codes, we then web scraped the USU Catalog's "Courses" webpage[2] to correlate the course IDs with the course name and code. For example, the hyperlink for *CS 2810: Computer Systems Organization and Architecture* gives the course ID 286639. This will be used later to correlate programs' related course IDs with actual course codes and names. Finally, we scraped the sections offered, the number of students enrolled, and various metadata from the USU Banner system.

Web scraping was done using two different methods. Because the courses were paginated, they were collected and interpreted by JavaScript running in the browser's developer tools. Sections were collected using Python's *requests* via Banner's API endpoint.[3] The programs list and program data were also collected using Python's *requests* module, then parsed using BeautifulSoup.

B. Data Representation

This collected data was then coalesced into NetworkX graphs. One graph contained the data from the Registrar, connecting each program with the courses students in that program have taken (See Fig.). These edges were weighted by the number of students who took the course versus the number in the program for that term.

The second and third graphs both had the same nodes for colleges, departments, programs, and courses. The former had section nodes representing Fall of 2022, and the latter had the same representing Spring of 2023 (See Fig. and Fig. for Fall 2022 and Spring 2023, respectively). No edges are weighted in either of these two graphs.

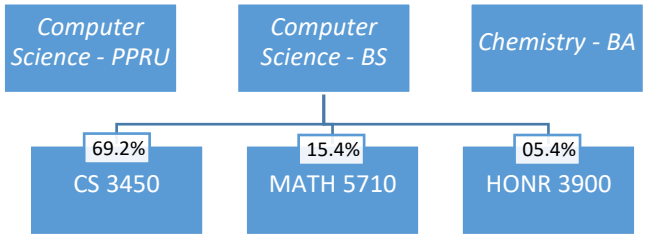

Fig. 1. Example of the data structure of the graph based upon the Registrar data, using Computer Science – BS as the exemplar.

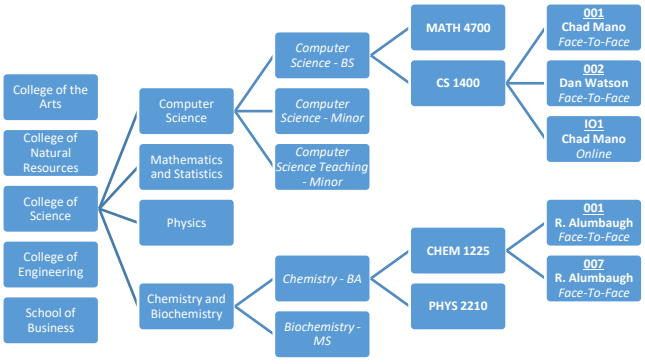

Fig. 2. Partial representation of each piece in the graph for Fall 2022. Left to right: Colleges, departments, programs, courses, sections.

[1] https://catalog.usu.edu/content.php?catoid=35&navoid=26809
[2] https://catalog.usu.edu/content.php?catoid=35&navoid=26241
[3] https://ss.banner.usu.edu/StudentRegistrationSsb/ssb/searchResults/searchResults

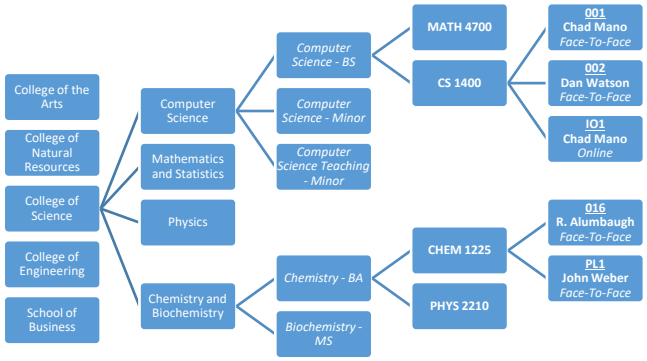

Fig. 3. Partial representation of each piece in the graph for Spring 2023. Left to right: Colleges, departments, programs, courses, sections.

C. Similarity Scoring

On the larger graphs, however, we did a high degree of analysis, using several different methods:

1) Jaccard Similarity

For each pair of programs, we calculated the Jaccard Coefficient to determine their similarity. This is calculated by dividing the number of similar neighbors by the number of overall neighbors:

$$J(u,v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|} \qquad (2)$$

$$\Gamma(x_i) = \{x_j \in \boldsymbol{A} \mid \boldsymbol{A}_{ij} > 0\} \qquad (1)$$

This similarity between programs is then used to create a graph (See Figure 4). If we take the average similarity of each program in a department with each program in another department, we can then get the similarity between departments, which can be used to create yet another graph (See Figure 5). If we then get the average similarity between departments in one college with departments in another, we can then also get the similarity between colleges, to make a third graph (See Figure 6).

One thing to take special note of in Figure 6 are the self-loops. From these, we can see that departments within the College of Science are typically highly similar to each other, while those within the College of Engineering typically are not. Because of this, a tiny bit of calculated similarity with the College of Engineering actually indicates far more true similarity than the same amount of calculated similarity to the College of Science. So, it may be more useful to look instead at the similarity score *relative* to the college's self-similarity score. We will call this *relative similarity*.

2) SimRank

SimRank Similarity is a NetworkX function that assigns a pair of nodes a similarity measure based on how similar their neighbors the algorithm is by nature cyclical so the function recurses until a certain tolerance is achieved. Similarity from one node affects the whole graph. Because of which we used the node types: colleges, departments, programs, and courses. This gives a read on how departments are related to colleges and other departments based on how many similar classes they share on the courses level.

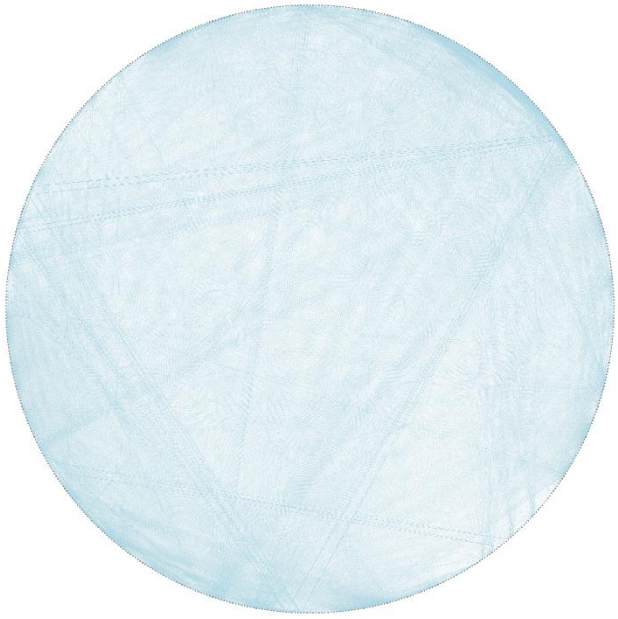

Fig. 4. Graph of the Jaccard similarity between all programs at USU. Programs in the same college are next to each other.

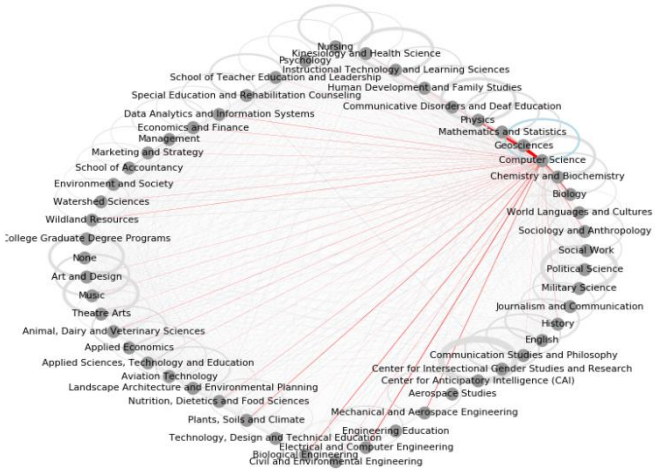

Fig. 5. Graph of similarity between all departments at USU. "None" refers to programs not in any department. Edge weights from Computer Science are multiplied by 8.

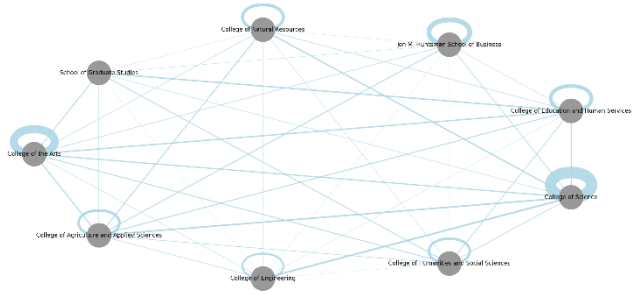

Fig. 6. Graph of similarity between all colleges at USU. "School of Graduate Studies" refers to all programs not in any college.

3) Common Neighbor Centrality

We also employed a function called Common Neighbor Centrality. It takes a graph and for all nodes it measures how many neighbors they share and what the shortest pathlength between them was. A constant α multiplied by the size of the common neighborhood and (1- α) multiplied by the degree of closeness to each node pair. Added together they make a similarity shown in this equation Where N is the number of nodes in the graph:

$$\alpha \cdot (|\Gamma(\mu) \cap \Gamma(\nu)|) + (1 - \alpha) \cdot \frac{N}{d_{uv}} \quad (3)$$

4) Panther Similarity

Panther similarity tells us how frequently two nodes are found on the same path. This takes as one of the inputs a result return size of k. This became a limitation when using all the graph nodes in our dataset, making finding all the colleges similarity to the CS department difficult. We ended up using it as a way to find the first college that was similar by path frequency.

D. Community Analysis

Community analysis is a method used to identify and analyze the relationships between different nodes in a graph. In the context of Utah State University's colleges, departments, and programs, community analysis can be used to understand how different departments and programs are related to each other and to the colleges they belong to.

1) Absolute and Relative Similarity

One approach to community analysis is to use measures of absolute and relative similarity. For any department, we can compute the relative similarity between colleges before and after adding it, as well as between the department and all other departments in the college. This can help us understand whether a department is a good fit for a particular college, based on how closely it is related to other departments in the college.

2) Louvain Communities

Another approach to community analysis is to use the Louvain algorithm, which is a fast and efficient method for detecting communities in large graphs. The Louvain algorithm works by iteratively optimizing a modularity measure, which is a measure of the strength of connections within a community relative to connections between communities. By maximizing this modularity measure, the Louvain algorithm can identify clusters of nodes that are densely connected within the same community, and relatively poorly connected to nodes in other communities.

3) Increase in Self-Similarity

One way to further refine the analysis of communities within the graph of Utah State University's colleges, departments, and programs is to focus on maximizing the similarity within communities. This can be achieved through various techniques such as clustering algorithms or network centrality measures, which can identify nodes that are highly connected and central within a community.

4) SimRank

Another approach to community analysis is to use hierarchical clustering based on SimRank. By using results from SimRank, we can use a hierarchical clustering algorithm to group the most similar departments first. This algorithm uses Single Linkage to decide which sets of nodes are combined, this can help us identify clusters of nodes that are closely related to each other within the graph of Utah State University's colleges, departments, and programs.

IV. Experimental Results

In this section of the paper, we present the results from our analysis of the graph of USU's colleges, departments, programs, and courses. Our analysis involved comparing absolute versus relative similarity between departments and colleges, using the Louvain algorithm to create departments out of related colleges, evaluating departments' community membership based upon increasing colleges' self-similarity, and using hierarchical clustering based on SimRank to find the most similar departments to Computer Science.

Because of some oddities with the Registrar dataset (such as the number of students in the *Computer Science – BS* program being only 13 people for Fall of 2021), we did not analyze it out of fear of getting false or unreliable results from dirty data.

A. Absolute and Relative Similarity

In Fig. , we compare three departments to the College of Science and the College of Engineering. Chemistry and Biochemistry (C&B) is a department that easily and obviously belongs within the College of Science, and Electrical and Computer Engineering (ECE) easily and obviously belongs within the College of Engineering. We can use these test cases to determine whether similarity is a good way to sort Computer Science into a college.

Using absolute similarity, all three departments fall within the College of Science, but using relative similarity, they all instead should be placed within the College of Engineering.

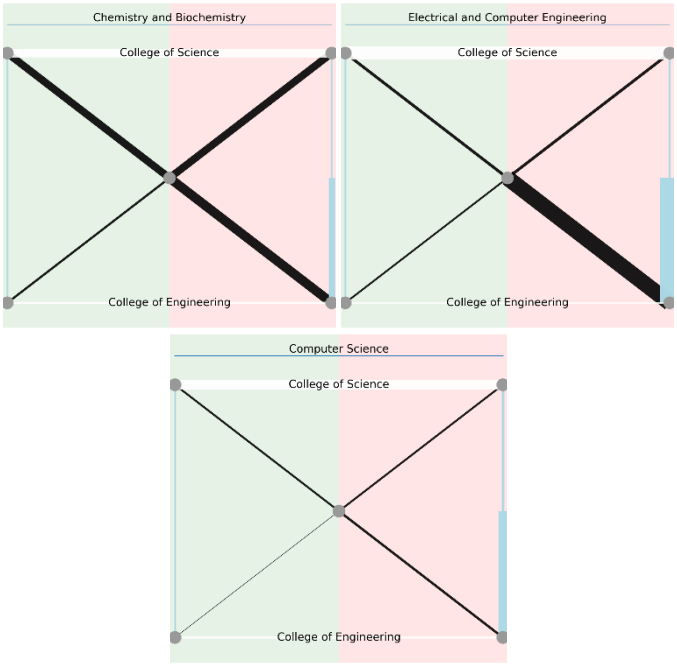

Fig. 7. Similarity of three departments with the colleges of Science and Engineering. Green is absolute similarity, and red is relative similarity.

B. Louvain Communities

The Louvain algorithm, as explained above, creates communities by optimizing modularity. When run on the graph of programs and courses, it creates approximately 100 departments when the resolution is between 0.5 and 4. Because smaller resolutions favor larger communities, the departments made at 0.5 are typically two 250-program departments and ninety-eight 1-program departments. A resolution of 4 typically creates a hundred 3-program departments. Using a resolution of 2 yields a number of actually interesting, useful departments, such as a whole Farming-based department or an Understanding-People department.

C. Increase in Self-Similarity

Like the Louvain algorithm, we will place the Computer Science department into a college by trying to optimize modularity. This optimization attempts to maximally increase similarity within a community and maximally decrease similarity outside of the community. As seen in Figure 8, adding the Computer Science department to either college increases its self-similarity by approximately the same amount. Notable, however, is the change in similarity to other colleges. By placing it in Engineering, similarity increases marginally to Art, and reasonably to the College of Agriculture and Applied Sciences (CAAS). In the College of Science, however, has the same increases as Art and CAAS, but also increases similarity with Engineering. Placing Computer Science in Engineering would therefore optimize modularity.

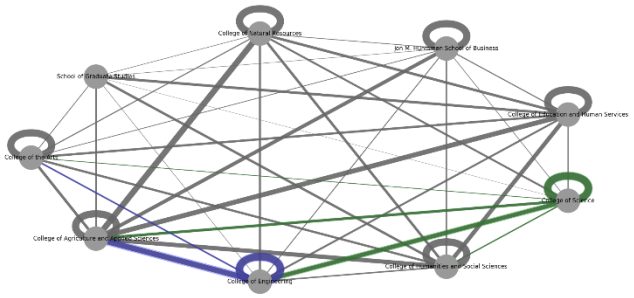

Fig. 8. 1 Change in relative similarity between colleges when the Computer Science department is added to the College of Science (green) or the College of Engineering (blue). Gray displays similarity where the department is not in any college.

D. SimRank Similarity

Using the SimRank function gave us expected results of what we had previously assumed were similar departments. The most significant are found in the table in Figure 9. By SimRank similarity the most similar department to CS is the Math department which there are plenty of math classes that CS students need to take. Notably The math department is in the college of science.

Other departments highly similar to CS are all within the College of Science and Engineering. There are more Science departments listed but this also includes the information that CS is currently listed under the College of Science. This adds an amount of bias towards the Science department. With the data given, there may be more Science departments close to CS yet amongst those the engineering ones are very close.

E. Clustering by SimRank

Using hierarchical clustering we were able to find what communities would emerge based on what departments are most similar. The first pair of departments being the most similar to each other were “Electrical and Computer Engineering” and “Mechanical and Aerospace Engineering”. The first group of 4 were “Watershed Sciences”, “Environment and Society”, “College Graduate Degree Programs”, “Wildland Resources”. Eventually CS was picked up by the first two engineering departments along with the math department and the physics department.

Sim-rank	SimRank CS and other departments	
	Department	***SimRank score***
1st	Mathematics and Statistics	0.0280
2nd	Electrical and Computer Engineering	0.0263
3rd	Physics	0.0214
4th	Mechanical and Aerospace Engineering	0.0202
5th	Geosciences	0.0175
6th	Chemistry and Biochemistry	0.0163
7th	Biological Engineering	0.0157

[a.] Scores are normalized over all nodes in the graph not just departments

Fig. 9. Top similarity ranking of other departments with the CS department

V. Conclusion

A. Answers to Research Questions

The first question we posed was, Does the Computer Science (CS) department more naturally reside in the College of Engineering or in the College of Science? After all the experiments and similarity measures were collected it is still hard to say definitively if there is a “correct” College that CS belongs to. The results show evenly that the CS department has a place within both the colleges of Science and Engineering. Looking at the programs within each department as well as the courses required by both, there are a few that are CS courses. Within the Computer Science program, you will find more science course requirements than engineering. However, both Engineering and Computer science require science courses. These requirements make each college similar to the other. It goes to say CS as a department and program can fit into either Science or engineering. It can be observed that the different colleges have put the department either college for their own reasonings.

Second, if a student decided they wanted to change their major, to what program could they easily switch to? The answer does depend on a case-by-case basis and the preferences of each student. But as a general suggestion students should change to a major within the current college or department that they have already put work into to maximize their current credit usage. Naturally that’s what people are often seen doing to minimize credit waste. As an interesting note, there are many similarities between the science and engineering departments. Scientists and engineers alike will find switching between these colleges will be less of a hassle than outside their scope. They all reside within

the scope of S.T.E.M. Often in practice it is seen that Math, CS, and engineering majors often will switch to another major within S.T.E.M.

What about minors? For similar reasons as switching departments minors can be taken from other departments in S.T.E.M. without too much trouble. CS majors At USU don't get to enroll in any minors so if they do choose a minor, it is usually a math minor that only requires a few more math classes.

B. Lessons Learned

1) Data is Messy

For the purposes of in-class learning our data was provided to us in a clean straightforward format. It weas easy to load in and set graphs up accordingly. Collecting it ourselves proved to be more challenging but manageable.

2) Algorithms, new and revisited

There were many algorithms that were covered in class. However, a select few we were able to apply to our graph structure that would give us meaning full results specifically similarity measure was what we were looking for. We found new ones that gave interesting results and insights.

3) Graph Structure is Important

The whole experiment rested on how courses, programs, colleges, etc. were connected in the graph to reveal similarities and run algorithms to find implicit relations to each node. This because of the flow from colleges to courses were specific it took multiple passes explicitly adding certain edges to the correct nodes. In the end the graph was well put together.

C. Future Work

There are plenty of options to take this work beyond what we have completed at this time some of those being:

Including idea survey data to courses. There could be some sentiment analysis of classes and generalizing programs, departments, and colleges. This could lead to revealing information on how students view their courses and what pains students can avoid if desired.

Taking a machine learning approach. Even with simple models there are many opportunities to explore machine learning with this graph. Graph Neural networks could be something to look into. Things such as path finding for prerequisites in classes, optimal graduation course paths, and next semester schedules suggestions. There are plenty more applications that could be useful with the data aggregated as such.

ACKNOWLEDGMENT

We would like to acknowledge Kristi Swainston for helping us obtain the registrar data.

REFERENCES

"Programs of study," *Programs of Study - Utah State University - Acalog ACMS™*. [Online]. Available: https://catalog.usu.edu/content.php?catoid=35&navoid=26598. [Accessed: 18-Dec-2022].

R. F. Hriez and G. Al-Naymat, "A framework to capture the dependency between prerequisite and advanced courses in higher education," *Journal of Computing in Higher Education*, vol. 33, *(3)*, pp. 807-844, 2021. Available: https://login.ezproxy.lib.utah.edu/login?url=https://www-proquest-com.ezproxy.lib.utah.edu/scholarly-journals/framework-capture-dependency-between-prerequisite/docview/2596178771/se-2. DOI: https://doi-org.ezproxy.lib.utah.edu/10.1007/s12528-021-09292-0.

I. Uriah, B. P. Koester and T. A. McKay, "Campus Connections: Student and Course Networks in Higher Education," *Innovative Higher Education*, vol. 45, *(2)*, pp. 135-151, 2020. Available: https://login.ezproxy.lib.utah.edu/login?url=https://www-proquest-com.ezproxy.lib.utah.edu/scholarly-journals/campus-connections-student-course-networks-higher/docview/2387228331/se-2. DOI: https://doi-org.ezproxy.lib.utah.edu/10.1007/s10755-019-09497-3.

A. Slim, G. L. Heileman, W. Al-Doroubi and C. T. Abdallah, "The Impact of Course Enrollment Sequences on Student Success," 2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA), 2016, pp. 59-65, DOI: 10.1109/AINA.2016.140.

A. Slim, J. Kozlick, G. L. Heileman, J. Wigdahl, and C. T. Abdallah, "Network analysis of University Courses," *Proceedings of the 23rd International Conference on World Wide Web*, 2014.

T. Zhang, "Mining Course Groupings from Student Performance." Order No. 27964975, Fordham University, Ann Arbor, 2020.