Modeling and Visualizing Student Flow

Mohammad Raji, John Duggan, Blaise DeCotes, Jian Huang, and Bradley Vander Zanden

Abstract—In this work, we present a data science system to model and visualize student flow patterns based on electronic student data of a university. Our system is called eCamp. The datasets used by eCamp were previously disconnected and only maintained and accessed in a siloed manner by independent campus offices. At a campus-level, our models and visualization show how students make choices among hundreds of potential majors, as students gradually progress towards their sophomore, junior, and senior year. At a department-level, the student flow patterns revealed by eCamp show how each course plays a different role within a curriculum. eCamp further dives down to the granularity of the exact classes offered in each semester. At that level, eCamp shows how students navigate from one set of classes in one semester to another set in a subsequent semester. Previously, comprehensive information about student progression patterns at all of these level was simply unavailable. To that end, we also demonstrate how insights into such student flow patterns can support analytical tasks involving student outcomes, student retention, and curriculum design.

Index Terms—Big data applications, data analysis, data visualization

1 Introduction

In this work, we propose methods to model and visualize university student flow patterns on three levels: 1) on a campus level, where students flow through hundreds of potential majors, 2) on a department level, where students flow through various core and general parts of a degree program's curriculum, and 3) on a classes level, where students plan and flow through classes from one semester to the next, based on their academic goals and progress.

Insights about these student flow patterns can help faculty and institutions better design, support, deliver, evaluate and fund college education. The same insights also help students make better choices, and help advisers provide more targeted and more informed advices.

We gain these insights from electronic student records data. In particular, this work is based on data of 145,000 students over a period of 16 years from the University of Tennessee, Knoxville.

Using a data science approach, we address key traditional problems faced by decision makers. First, even though many individuals in an organization may own various pieces of the information, few have a clear, confident, and complete view of the situation. Second, when sophisticated designs and mechanisms are in place, continuous improvement and optimization can be hard because the measure of how well the intended outcomes are achieved lacks contextual specificity and quantitative rigor. Third, while population-scale or individual-level characteristics may be known, it can be hard to bridge those understandings under a common framework.

Furthermore, there is an increasingly wider gap between how a system, such as a university, should function versus how the system is functioning. On one hand, the design of how a university provides education has been influenced cumulatively over decades by many people holding different perspectives. The design was also shaped by priorities and challenges at different times. The intermixed effects of evolving philosophy, rules, policies, and practices are hard to quantify. On the other hand, students flowing through the university system will make choices and take actions based on subjective goals of their own. Their behavior is hard to predict. It seems very likely that data science could be the only feasible way to provide "the full picture".

In this work, we model and visualize student flow at different levels. These student flows were previously unknown, even though the raw data actually exists. For example, at a campus level, different degree programs are organized administratively into colleges, and the common hypothesis and advising guidelines are that students would choose their college first and then their major. However, as data would reveal, the flow of the student population over a typical 4-year span of college enrollment does not abide by college boundaries. At a department level, even though we can build a course prerequisite graph from the course catalog, and one might expect that the student flow correspond to that graph, data would also show otherwise. Similar examples abound.

We have implemented our methods in a system called eCamp. At the campus level, we show how college freshmen as a population start their college life taking general education courses together, and gradually divide and specialize into their chosen degree programs as they rise through sophomore, junior and senior years. At a department level, eCamp models course-course relationships in a degree program and visualizes the student flows of each curriculum. At the classes level, eCamp reveals how success and/or failure in a class correlates with academic performance in other classes in the same and following semesters.

• *The authors are with the University of Tennessee, Knoxville, TN 37996 USA. E-mail: {mahmadza, jduggan1, bdecoate, huangj, bvanderz}@utk.edu.*

Manuscript received 16 Dec. 2017; revised 14 Apr. 2018; accepted 20 May 2018. Date of publication 28 May 2018; date of current version 29 June 2021. (Corresponding author: Mohammad Raji.)
Recommended for acceptance by A. Perer and C. Scheidegger.
Digital Object Identifier no. 10.1109/TBDATA.2018.2840986

eCamp can help answer various questions about college education. Academic departments and faculty can use it to understand what courses are the most critical for students' long-term academic achievements in the major. Students can use eCamp to understand implications of their current grades so that they can better plan their next semesters. New students can better guide the search for their most suitable majors and know the time limits by which they need to commit to their choice to graduate on time.

We discuss the background of this work in Section 2. The application needs and the motivation of this work are in Section 3. Our modeling approach is described in Section 4. The visualization and analytics parts of eCamp are shown in Section 5. Finally, we discuss domain user feedbacks in Section 6 and conclude in Section 7.

2 Related Work

2.1 Application Background

The university is a "man-made" complex system that serves a fundamental role in shaping the modern society. How to make universities work better is a growing focus, especially in the United States [1], [2]. It's hard to gain a clear and comprehensive understanding of how a university functions, in part because universities produce the most valuable and also the most unique and complex product for the society—the human capital, the students.

It seems clear, however, without gaining a better understanding of the intrinsics of a university, simply making more monetary investment does not lead to significant improvements [3], [4]. To that end, the availability of electronic records, as captured by various university databases, has enabled using data science to gain the much needed insights. The development of the first version of eCamp [5] was motivated by this research premise.

Researchers have already developed many tools to analyze university databases [6], [7], [8], [9], [10], [11], [12], [13]. For example, DynMap models the topics in a course as a concept map to help inspect how well students understand the course content. [13]. CourseVis uses web log data from WebCT, a course management system, to track and visualize student progress in a course, especially for distance learning settings [12]; the system was later extended to help instructors see social, cognitive, and behavioral aspects of their students [11] within a course.

There are also previous works that studied course-course relationships. For example, Siirtola et al. showed student progress in the context as prescribed by the catalog [14]. Gama et al. visualized student grades in relation to which semesters the courses were taken [15]. Wortman et al. used graphs to explore patterns in how students' grades change as they retake courses in computer science [16].

eCamp aims to reveal student flow patterns on a university-scale. In [5], we first targeted administrators and faculty as users, and hence focused on campus-level and department-level flow patterns. In this work, we extend eCamp to include students as targeted users and added a novel per semester-level visualization of student flow.

Compared to other existing systems, the kind of data used in eCamp is the primary difference. To our knowledge, eCamp is the first data science research system built based on university-wide student population data.

2.2 Technical Background

There are three main entities in our dataset: majors, courses, and students. Major and courses have a natural and codified relationship in the university catalog. How the student population take overlapping courses that belong to different majors can suggest how strongly the majors are similar to each other, especially for new students such as freshmen and sophomores. As students progress through their degree programs, the courses they take become more specialized, thus gradually reducing the feasibility of switching majors.

At a campus-level, eCamp models major-major relationships as a hierarchy based on the above heuristics. Visualizing hierarchies has been well studied. Two traditional methods are to use Dendrograms and TreeMaps [17]. However, the hierarchies modeled by eCamp have a temporal aspect, which cannot be handled by Dendrograms nor TreeMaps. Due to that reason, we also consider Radial trees, which can effectively show how biological species evolve over time as in phylogenetic trees [18], [19]. However, radial trees have a limitation. They cannot easily visualize how different portions of entities go through the hierarchy.

Sunburst graphs can address this limitation of radial trees. For example, PathRings has used sunburst graphs for showing biological pathways [20]. Sunburst graphs cannot effectively show flow information, however. A popular method to address the need of showing flow information, especially temporal flow, is to use Sankey diagrams [21], [22] or Sankey-like structures [23], [24]. Based on the literature, we decided to use a variation of a radial tree which is also combined with Sankey-like edges to better convey student flow through time.

At a department-level, eCamp models student flow as represented by correlations between courses. We use Pearson's coefficient for course-course grade correlation. The resulting department-level curriculum graph for each degree program is a directional acyclic graph. As typically done in the field, we render the department-level curriculum graph as a node-link diagram [25].

At a per semester-level, eCamp looks at student flow through classes. A class is defined as a course taken in a specific semester. At this level, eCamp models temporal class relationships as a multilayer graph with interlayer edges [26]. Visualizing multilayer graphs are often challenging due to their multidimensional structure. DiffAni used a *diff* technique to compare timesteps of a dynamic graph [27]. Stein et al. created a pixel-based technique to show the changes in a dynamic graph [28]. Burch et al. combined a timeline with matrix cells to show graph changes. Although, matrix-based methods have the advantage of staying readable for large graphs [29], previous methods mostly focused on how edges change in a dynamic graph. In eCamp, we need to focus on class relationships within semesters, which characterizes a single graph layer, and also inter-semester relationships between classes. Hence, we employ a novel matrix-based visualization of the graph that incorporates interaction and mouse-hover gestures to show dynamic relationships. Interactive matrices have been used in similar contexts before such as in OnSet to show the relationship between elements across multiple sets [30]. We take the interaction component further by combining it with queries.

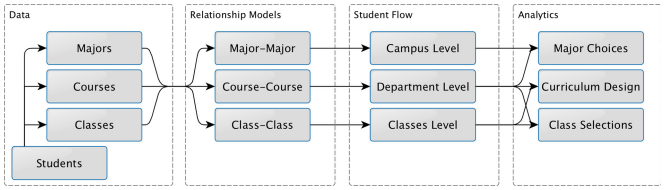


Fig. 1. Overview of eCamp's data pipeline. The data and relationship modeling are detailed in Sections 3 and 4 respectively. The student flow visualization and analytics components are discussed in Section 5.

3 Design Considerations

3.1 Overview

eCamp models how students flow through a university's degree programs in order to visualize actual population-scale patterns of how a university serves students. The visualized patterns show both confirmation and surprises in regard to how the university is designed and intended to work versus how the university is actually working.

Fig. 1 shows an overview of the data pipeline. We model relationships on three levels of abstraction. On a campus-level, we look at how different majors overlap and differ. On a per department-level, we look at how various courses in a curriculum have intended and unexpected effects on student success. Finally, on a classes level, in relation to each student's current academic performances, we look at how the availability and scheduling of classes in each semester can either enable or limit the student's choices. The resulting models are then visualized in ecamp through three types of visualizations (shown in Fig. 2).

3.2 The Data

Universities are similar to many large organizations. While there is a common system to maintain electronic data, many different campus offices maintain and use only some isolated components of the database. It is important that the organization apply a comprehensive data science approach to gain true situational awareness on multiple levels.

Specifically, many universities, including the University of Tennessee, use Banner [31] to maintain their central databases. Banner is an information system for universities that facilitates admission, registration, and curriculum management processes. We obtained an anonymized copy of the Banner database from the time span of 1996-2012. Table 1 shows the primary kinds of data in the dataset. Overall, there are records from 144,798 students in over 400 majors.

Graduate records show which students have successfully graduated from each major. Naturally, a portion of the student population has not graduated by 2012. While some of those students were dropouts, many of them remained in school after 2012.

In Banner, each major is identified by a major code. Graduation records reference major codes. When a degree program is revamped, a new major code may be issued to the degree program. Due to this reason, the names of each major included as part of major information may not be unique. Our models use major codes as the only identifier. However, our visualization use major names as labels for usability reasons.

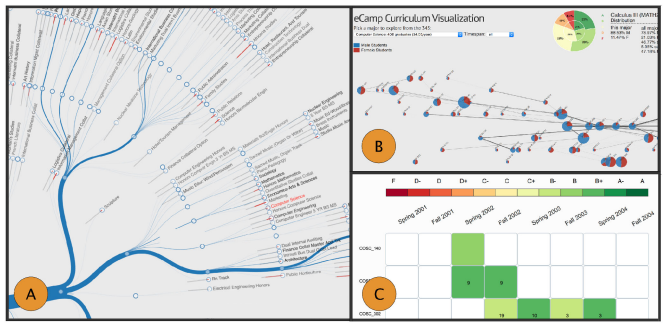


Fig. 2. An overview of the visualizations in eCamp is shown. (A) shows a view of the campus level radial tree depicting student choices with regard to their major. (B) shows the relationship between courses in a curriculum using a node link diagram. (C) shows student flow through classes using an interactive matrix.

Student grades data provides detailed information about all of the classes taken by each student, including the final grade as well as when the student took each class.

In eCamp, we draw a significant distinction between a class versus a course. A course is an abstract entry in the catalog of a degree program. A class is actually taught in a particular semester, by a particular instructor, and taken by a set of students. A class is the actual instantiation of a course. In fact, "courses" do not exist in our dataset. We have to aggregate all class offerings of the same course together to obtain the information about a course.

The distinction between course and class is important because of our multi-level modeling. When studying major-major relationships, we need to consider courses shared by the majors. When studying curriculum structure, we need to use courses as the finest granularity as well. When modeling student progression patterns on a per-semester level, we have to study classes and how students flow through classes.

3.3 Analytics Needs

While developing eCamp, we met with various potential users to understand how a better situational awareness may benefit them. We met students, faculty, as well as departmental and campus level administrators, including the former Vice Provost of the University of Tennessee, Knoxville.

The Provost's office seek analytics to better evaluate the effect of existing advising programs on improving student retention and reducing time-to-graduation. This would be especially beneficial because current advising programs are divided and executed by the campus as a whole, by individual colleges and by individual departments. Advisers from all of the advising programs may hold different and sometimes conflicting perspectives.

At the campus level, there is also a pressing need to understand how success in general education courses may affect a student's success in different majors. In addition, students can change majors. This kind of cross-discipline

TABLE 1
eCamp Dataset Informtion

Category	Number of Entries	Size (MB)
Graduation Records	100,239	33
Student Grades	4,723,835	461
Major Information	436	< 1

mobility is a natural occurrence on a college campus. There is very limited information on how to best provide transitional advising to such students at different stages of their decision and transition process.

At the department level, a recurring focus is to more accurately understand how students progress through the curriculum of each major. While the catalog can serve as a reference, it remains a subjective exchange of "lore" and "feeling" when needing to identify which courses truly play their intended roles in the curriculum. Examples of such roles are gate keeping courses, core courses that serve as the basis for other courses, and peripheral courses that diversify students' knowledge. In addition, all degree programs want to improve student success, retention, and diversity. There is a great desire for having a data-driven approach that can model and visualize the corresponding barriers.

At classes level, students and advisers face a routine task. That is, based on a student's past and current academic progress, what courses are the best for the student to take in the next semester? Currently, most of such decisions are made according to experience, gut-feel, and sometimes even just hear-say. In particular, based on low grades that a student has achieved in different courses, can the data objectively identify bad choices that will hurt a student's chance for long-term success?

As a result of analyzing these analytics needs, we have focused on modeling and visualizing three levels of student flow patterns: (i) student flow through all of the degree programs on a campus level, (ii) student flow through the curriculum structure within a degree program, and (iii) student flow through classes on a per semester level.

4 Modeling Student Flow

In this section, we describe the modeling part of our overall data pipeline (Fig. 1).

4.1 Modeling Goals

In our dataset, there are three main academic entities—majors, classes, and students. A fourth entity, course, can be created by aggregating classes information. In Sections 4.2, 4.3, and 4.4, we model a variety of relationships between these entities. We use these relationships to visualize student flow at campus, department, and classes level. In the following, we describe the details of each of these levels.

4.1.1 Campus Level

For a university as a whole, at first, new students fall in a single group where everyone has the option to choose any of the majors offered. As students take different courses each semester, the set of majors they can potentially choose from narrows quickly. At the same time, the student population also start to diverge, paths start to emerge and continue to become narrower and more specialized because the courses being taken are more and more specific to particular majors.

In reality, especially for new students, their first semester on campus may entail a lot of undecidedness. Some students may remain undecided for a longer period of time. They may also change their majors. Hence, even though a student may have an intended or declared major at the point, that information is not reliable for our purpose. We need to infer students' intended major based on how courses are shared among majors, and model student choices throughout their stay on campus.

Students can change their majors. These changes come with an overhead, because not all courses that a student has taken are relevant to their new intended majors. The further along one gets in an intended major, the more overhead there will be for the student, especially if the student plans to graduate on time. On a campus-level, some degree programs require that a student specialize in that program as early as possible. Some other majors would allow students much more time to sample other disciplines before deciding to specialize. Most administrators, faculty, and even students would assume that to be true, however, few truly know towards which end each major would fall in that spectrum. We intend that our model will be helpful for answering these kinds of questions.

4.1.2 Department Level

From outside each department, a degree program may be described by the set of courses required by the program. From inside each department, what defines a degree program is on a deeper level—the structure among the courses, i.e., the structure of each curriculum.

In eCamp, we model how courses relate to each other, not according to the pre-requisite or co-requisite structure specified in a course catalog, but instead by grade correlations as observed for the student cohort of the degree program, as well as the natural order in which the students actually take the courses.

The modeled structures are based on actual student outcomes. They may be different from the intended structure as specified in the catalog. When that happens, the differences will help reveal gaps between the "design intentions" and the "in situ realities".

The visualizations will help to change academic decisions from completely "experience-based" and "idea-driven" to "evidence-base" and "data-driven".

4.1.3 Classes Level

The time horizon of campus-level decisions is on decade-scale. For department-level decisions, especially those related to curriculum design and adjustments, the time scale is normally 2-3 years due to typical catalog update cycles.

A much faster and more personalized kind of decisions have to be made each semester. That is, next semester, what is the set of courses that will be the best for each student to take. There can be a rich set of optimization criteria. Clearly, the decision has to also consider each student's past and current academic progress.

In fact, many of the campus-level patterns are not relevant to these per semester decisions. Neither are the department-level considerations of the curriculum. The data analytics need to help answer questions like: "I got a C in COSC 140, should I attempt to take COSC 302 and COSC 311 at the same time next semester?"

We observe that there are common *groups* of courses that are often taken together, and there are natural *sequences* as well. These structures can be modeled on a per-semester level. In addition, we query the model to reveal how success

or failure in one class relates to success or failure in other courses. The visualization can serve as a data-driven advisor for each student's personalized scheduling questions.

4.2 Major-Major Relationship

Among all courses taken by students in any major, many of those courses are shared by multiple other majors. In other words, degree programs can and do have an overlapping relationship with one another. The overlaps can be large for new students, but then diminish as students start to take courses that are more specific to their own major.

We model the major-major overlapping relationships based on student records from those who have graduated, so that we can derive the set of courses that can be associated with each major.

4.2.1 M-Value

We first estimate the degree to which students in a single major will take a set of courses. Given a major A and a set of courses C, the estimate, M_A, is

$$M_A = \sum_{c_i \in C} \frac{s_A}{|S_{c_i}|_2 |A|}, \tag{1}$$

where s_A is the number of students from major A in the course c_i, $|A|$ is the total student population of major A. $S_{c_i} = [s_{m_1}, s_{m_2}, \ldots, s_{m_n}]$ is a vector of counts of students in c_i from all of the n different majors. $|S_{c_i}|_2$ is the Euclidean norm of the vector S_{c_i} and is computed as $|S_{c_i}|_2 = \sqrt{\sum_{k=1}^{n} s_{m_k}^2}$.

In Equation (1), $\frac{s_A}{|A|}$ corresponds to the probability that students in major A take course c_i. The per-course scores are then tallied up across the whole set of courses to form the overall M-Value for the major.

Some courses are taken by a much broader group of students than others. For example, introductory English courses have very little specificity in terms of majors, because they are shared by the entire student population. The additional term $|S_{c_i}|_2$ is introduced to reduce the weight of those general courses. This means that the final M-Value metric will be weighted towards courses which are shared between small sets of majors. A high M-Value means the given course set $Courses$ has a high specificity to a major. If the students in a major are not taking the courses in $Courses$, the M-Value will be low.

The M-Value essentially measures the affinity between a major and a set of courses. In other words, on the basis of a fixed set of courses, C, one can compute the affinity measure of all of the majors on campus with that set of courses, C. For example, if the course set C consists entirely of bio-engineering courses, the M-Values computed for each major can help to rank the similarities of all of the majors on campus with bio-engineering.

4.2.2 Major-Major Relationship Graph

Using the M-Value, we can capture the similarity between all majors on campus. This similarity for two majors, A and B is calculated as

$$M_{A,B} = \frac{M'_A + M'_B}{2}, \tag{2}$$

where M'_A is calculated for major A according to Equation (1), but using the course set taken only by students in major B. M_B is calculated for major B, but using the course set taken only by students in major A.

Suppose major A is computer science and major B is math. Then M'_A measures the affinity between the major of computer science and the math major's courses. M'_B measures the affinity between the major of math and the computer science major's courses. $M_{A,B}$ is an average of those two metrics and is the same value as $M_{B,A}$.

One can now gain a more precise control of the model by controlling which set of courses are used to compute the M-Values. For example, one can make major-major comparisons based on stages of a student's education, by including in the course set, C, only those courses taken typically by the student population during the corresponding stage (such as freshman year versus sophomore year or later). The resulting major-major similarities computed using M-Values will then vary from freshmen, sophomore, junior to senior year.

Conceptually, it is desired to then model and visualize majors gradually diverging from each other as time progresses for the student population on a per-semester basis, and observe how students move or dropout along the way.

The algorithm is essentially top-down clustering, beginning with all majors in a single group, with the result being a tree. At each stage (i.e., academic semesters during freshman, sophomore, junior, and senior year), an M-Value similarity matrix is calculated using each semester's courses.

The tree is initialized to have only the root node with all majors belonging to it. Then, the process proceeds step by step. Starting with the first semester of the freshman year—the courses typically taken during that semester are chosen, and a similarity matrix is produced. In each step, one new level in the tree is created.

The process then proceeds to the next stage-the second semester during freshman year. The above process is recursively repeated, treating the leaf nodes (first-semester division) as sets of majors to further divide, selecting courses typically taken by that group of students as the basis to determine how to make the division through clustering. This process continues through all eight semesters in a 4-year tenure of each student.

In each step, tree nodes are partitioned using a similarity matrix based on M-Values that are computed from course sets specific to that semester. In addition, when a tree node contains only one major, it is not further subdivided.

The hierarchy of majors that results from this algorithm has a very clear interpretation. Fig. 3 is a rendering of the hierarchy as a Sankey-like radial graph. Each leaf node corresponds to a single major, and each internal node represents that the set of majors below it were considered to be similar majors at that semester.

This hierarchy of majors also shows how earlier choices made by students lead, or limit, them to certain majors as they progress towards graduation. This effect of temporal bifurcation cannot be captured through traditional methods.

4.2.3 Student Dropout Patterns

While junior and senior students usually have a "declared" major, they can still change their major without going to the registrar's office to update their records. In addition,

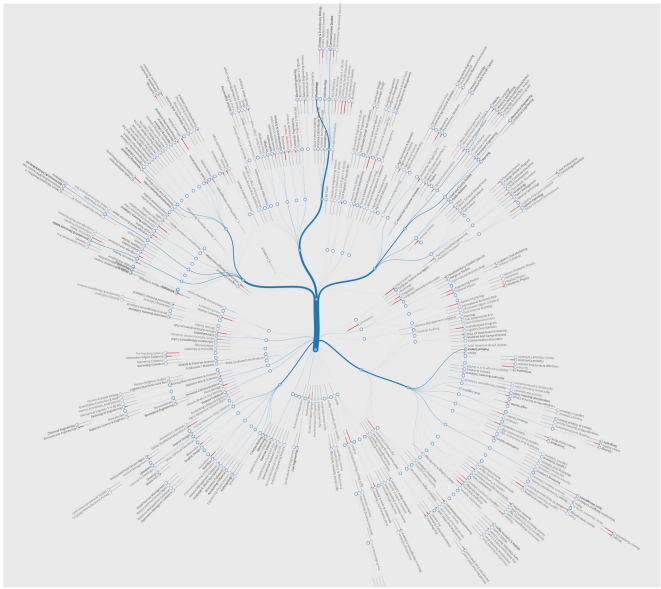


Fig. 3. Campus-level student flow. Each leaf node corresponds to a single major. The thickness of the paths corresponds to the number of students progressing through the corresponding nodes. The computer science major is highlighted in red towards the right of the visualization.

although freshman and sophomore students may also have a "declared major", many of them are in an exploration stage of their studies and they may be taking preparation courses that can lead to a few different majors. Effectively, their final major is unclear at that point.

Both of these situations can cause significant data quality issues if we analyze solely based on their "declared" majors. When it comes to analyzing for patterns of student dropouts, we need to make best-effort estimates of a student's intended major based on the data available.

For this, we look at the courses that the students have taken and measure the amount of overlap between those courses and the courses of each potential major. This overlap ratio represents a confidence value for our model. The higher this ratio is for a major, the more likely it is that they were pursuing that major.

Counting the number of estimated dropouts for a major can introduce uncertainties. For example, the intention of a student that has only taken two courses are more unclear than a student that drops out after having taken ten courses. Another potential caveat with this approach is a scenario where a student has changed major without updating his/her major in the university records, and then dropped out. By the data, it is difficult to not count the dropout as the previous major.

We account for these potential issues by showing the average confidence value for each major in a tooltip. The tooltip is shown when a user hovers over a major.

In Table 2, we show the top-5 majors in the database by number of graduates, and the average degree of overlap between courses taken by dropouts versus the full curriculum of the best-matched major. If the average overlap is high, then these are more likely to have intended to graduate from that major. If the average is low, then it is likely that these students are dropping out early in their studies. This could hint that general education courses are causing the dropout rather than specialty courses in the departments.

TABLE 2
Top Five Majors by Number of Graduates

Major Name	# of Graduates	Estimated # of Dropouts	Average Overlap
Psychology	3092	159	63.71%
Political Science	1263	52	61.40%
Journalism	1094	51	72.79%
Comm. Studies	1012	59	70.93%
Biochemistry	826	12	75.70%

The average overlap shows the average percent of overlap between courses of students that dropped out and the courses of the major.

The total number of estimated dropouts for each major is shown in the major-major graph using a red and gray bar. The percentage of the red bar over the gray bar represents the dropout percentage.

4.3 Course-Course Relationship

In academic departments, student progression is typically represented by pre-requisite relationships in course catalogs. However, many courses do not have pre-requisites. Additionally, some pre-requisite rules are not always enforced. Therefore, the actual progression of students cannot be captured effectively. In our available data, we found that student grades are the closest variable that when combined with temporal information about courses, can represent progression and success in a major. Specifically, we quantify how the courses taken by students in a major are structured with respect to when courses are taken by the students, as well as how courses are correlated in terms of student grades. With this knowledge, per-major curriculum structures can be determined. We believe other variables, such as instruction style, grading practices, rigor, etc. can help make the measure of student progression more accurate. However, these variables were not available.

For this purpose, we first calculate course-to-course correlation of student success. We then determine which courses are most-highly correlated with all other courses and at what point in time each course is taken.

4.3.1 C-Value

The approach for determining course correlations is the C-Value metric. The C-Value, informally, measures the similarity between two sets of grades, while accounting for the size of these sets. To begin the formal discussion, the C-Value is heavily based upon the Pearson Correlation Coefficient (PCC), which is commonly used for studying linear correlation between variables.

Let X be a collection of grades for course A, and Y be the collection of grades for course B. For these sample populations the PCC, $r_{A,B}$, can be described as the sample covariance of X and Y divided by the product of the sample variance of X and the sample variance Y. This yields

$$r_{A,B} = \frac{\sum_{i=1}^{N^{A,B}} (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{N^{A,B}} (X_i - \bar{X})^2} \sqrt{\sum_{i=1}^{N^{A,B}} (Y_i - \bar{Y})^2}}, \quad (3)$$

where $\bar{X}$ and $\bar{Y}$ are the sample means for X and Y, respectively, $N_{A,B}$ is the number of students who took both courses A and B, and X_i, Y_i are specific student grades.

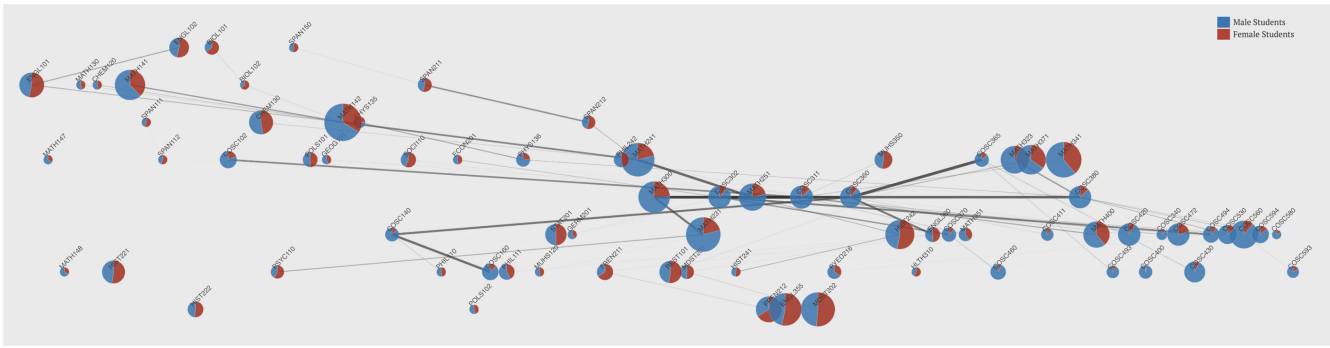


Fig. 4. Per curriculum student flow diagram. Thicker edges between courses show higher between-course correlation in student grades for students who took both classes. The major shown here is computer science. The "core courses" are MATH 300, COSC 302, MATH 251, COSC 311, COSC 360, and COSC 380. The size of the nodes corresponds to the percentage of students failing the class. Inside each node, the red portion corresponds to the ratio of female students, while the blue shows the ratio of male students in the class. The nodes show that the core courses in computer science tend to have less female students than peripheral courses.

By using PCC, one can see the correlation between courses based on how students performed within both of these courses. However, in this use case, PCC is insufficient without a final step. Consider a situation where only five people took a course typically unrelated to a major and then all went on to do well academically in the major. One might incorrectly determine that this course is highly correlated with success in the major. To correct for this, the final C-Value, is scaled by $N_{A,B}$, producing

$$C_{A,B} = N_{A,B} \cdot r_{A,B}. \tag{4}$$

In this work, we seek specific course-course relationships that are unique to a selected student population using the pairwise C-Value. For example, consider MATH 141, and PSYC 110 as two courses. The C-Value between those two courses could be different based on whether we are looking at the electrical engineering students who took them or the math majors. This can be the case even if the grades within each individual course could all be similar (e.g., follow a normal distribution), because the subset of students queried for C-Value calculation would differ.

Specifically, in Equation (3), the set of students used to query for grades in Course A and B is determined by the analytics. It is also possible that the set of students who have taken both courses is almost empty. In that case, the C-Value will become very small (after being weighted by Equation (4)), which then indicates, as far as the student population in question is concerned, that course A and B are likely to be unrelated because the students taking those courses are likely to be mutually exclusive.

Furthermore, it is likely that, two courses may exhibit a high correlation when exactly the same student population take two courses offered by the same instructor. In that case, grading habits of the instructor will affect the C-Value. Nonetheless, in this case, those correlations are usually not expected according to course catalogs, in which case, the C-Value does reveal data insights of previously unknown information (example in Section 6).

4.3.2 Per-Major Course-Course Relationship Graph

Calculating pairwise C-Value results in a similarity matrix in which each cell represents the C-Value between two courses. This information is most classically stored as a similarity matrix. To present the information visually, there are two common design choices. First, rendering the information as an adjacency matrix. Second, to render it as a node-link diagram or other variations of graph models. Because university administrators and students are primarily non-technical users of analytics, we opted for the node-link diagram (shown in Fig. 4). As shown in previous work, node-link diagrams are more easily understood by non-engineers and researchers [32], [33]. We also preferred node-link diagrams for their added flexibility [34], to encode additional measures (e.g., number of failures in a course, degree of being a representative course, and gender distribution) into the visualization to support more analytical tasks.

The graph helps capture success progression through a major in two stages. First, courses that represent overall success in a major can be defined as those that correlate most with all other courses of the major. With this in mind, we can sort courses based on how representative they are of success in a particular major. We call the most representative courses, "core courses". Returning to the similarity matrix notion of the C-Value results, this is done by finding the rows/columns with the highest sum. Second, we can calculate where a course fits temporally in the real-world curriculum. This is done by determining the average time, or semester, during which students take the course. Looking at core courses and their correlations, administrators can see if in practice the courses exhibit the logical organization that they had intended for them (visualized in Section 5.2).

4.4 Class-Class Relationship

The progression of students through a major is represented by the classes that they take. We define a class as a course taken in a specific semester. Every student takes a different set of classes each semester and therefore creates a unique path for themselves. Although students and faculty have a lore and feel of what some of these paths should look like based on the catalog, the real paths that students take as groups is mostly unknown.

Similar to how we build other models, again we use student population data to model course groups (in the same semester) and course sequences (across consecutive semesters). In this setting, a single path is a sequence of sets of courses taken by an individual. The aggregate of all paths show that some courses are "co-taken" by many students, and also some courses are "post-taken" after other courses.

These co-taken and post-taken patterns can be further augmented with information of student grades. For example, for students who have attained a low grade in one particular course, what are the most common grades achieved in all of the co-taken courses (i.e., the same semester) and the post-taken courses (i.e., in the subsequent semesters).

We model co-taken and post-taken relationships between sets of courses as a weighted multilayer graph, in which the layer dimension represents time, and inter-layer edges can exist [26]. As a multilayer graph, at each semester (timestep), courses may be taken together by the same students. In a graph, this relationship can be represented as weighted edges where the weights represent the number of students co-taking the classes. Inter-layer edges between timesteps represent post-taken relationships.

Mathematically, based on the definition from [26], our multilayer graph takes the form of $M = (V_M, E_M, W_M, V, L)$, in which V is a set of vertices, and L is a sequence of d layers. Then, $V_M \subseteq V \times L_1 \times \ldots \times L_d$, and defines which layers, each vertex is present in. Subsequently, $E_M = V_M \times V_M$ defines the edges between the vertices, and W_M is a function mapping edges from E_M to weights.

We now map this definition to our data. For each major, V defines the set of all courses for that major, while L represents the semesters. Thus, V_M tells us which courses are offered in the each semester. Finally, E_M defines intra-layer, and inter-layer edges between the courses. In this graph, a vector u takes the form of (u, α), where α is a canonical vector representing which layer u belongs to. Therefore, for (u, α) and (v, β), an edge is intra-layer (shows co-taken relationships), if $\alpha = \beta$. Otherwise, the edge shows a inter-layer relationship that depicts one course being taken after the other in a different semester.

Visualizing multilayer graphs is challenging due to clutter and the extra dimension that time entails. We have designed a novel visualization technique to show course-based relationships. Our technique is based on a dynamic matrix which we call a path matrix (detailed in Section 5.3).

5 Visualizing Student Flow

Based on the models constructed in Section 4, we describe how to visualize the student flow at campus-level, department-level, and classes-level.

The student flow at campus-level conveys student choices and mobility among all degree programs (Section 5.1). At the department level, the visualization of student flow reveals how courses within a department are structured and what role each course plays (Section 5.2). At the classes level, focusing on actual class offerings, we visualize the actual temporal paths taken by the student population and allow a variety of hypothesis-driven and interactive filters in the visualization (Section 5.3). We demonstrate analytics results using these visualizations in Section 5.4.

eCamp's user interface uses D3.js and is fully web-based. eCamp's backend system uses a regular Linux desktop. We implemented all of the data processing and modeling components in Python. Starting from scratch, the overall backend processing requires less than 10 minutes to run.

5.1 Campus-Level Student Flow

Fig. 3 shows the campus-level student flow. The visualization is based on the temporal hierarchy of majors constructed in Section 4.2.2. The center of the visualization is the root of the tree of majors. It is the starting point for all of the new students: before their first semester on campus when they have the over 400 majors to choose from.

In this hierarchy, each leaf node is a major, and each tree level corresponds to a semester, i.e., the first semester of freshman year, the second semester of freshman year, the first semester of sophomore year, etc. The tree edges collectively map out the student flow from the root to leaf nodes. Every step along any particular path, the potential choices of majors become narrower. Eventually when a path reaches a leaf node, a student on this path will be virtually exclusive for that major due to his or her past coursework. The width of the path corresponds (in a logarithmic manner) to the size of the student population remaining on that path.

This visualization shows how degree programs are similar in advancing students towards graduation. The tree nodes show points in time where students encounter critical decisions with regards to which majors they would like to follow. At these critical decision points, the courses students take may significantly limit their future options.

This visualization also shows dropout patterns. When a student drops out, the student's intended major can be predicted using the method described in Section 4.3.2. Each blue node on the path signifies that the subgroup has "traveled" together through the major hierarchy and reached a new milestone, a new semester. The gray and red line segments shown for each major represent the percentage of students who have dropped out of their intended major. When the red line segment equals the gray line segment in length, it means 100 percent dropout. Correspondingly, when the red line segment is half of the gray line's length, it means 50 percent dropout. When the user hovers over a major, a popup tooltip displays the actual number of students that have graduated or dropped out, as well as the confidence percentage for the dropout estimation. The opacity of the red dropout bars also reflects the confidence value.

A large monitor is required to better use the radial graph visualization. Zooming and panning is also supported and assists users in viewing the different branches as well as the dropout patterns. Additionally, clicking on a major dives in and shows the student flow diagram for that major.

From a design perspective, other detail-on-demand techniques can be useful too. However, with this work focusing on the modeling aspect of student progression, we believe a thorough study of these techniques calls for future research and is out of the scope of this paper.

5.2 Student Flow Through Each Curriculum

There are many courses involved in the curriculum of a degree program. However, they are not necessary from the department that offers the degree program. For example, computer science students at the University of Tennessee normally take courses from almost 30 different departments.

From a curriculum design point of view, seeing how the students flow through all of the courses in a curriculum would be beneficial, especially to see those courses that are taken by the student population frequently.

Fig. 4 shows such a student flow visualization for the computer science major. The node link diagram shows for each course which courses it is correlated the most strongly with, as well as the order in which they are taken.

This visualization shows correlations between courses. In the node-link diagram, each node is a course and each link represents a grade correlation between two courses, modeled as the course-course relationship described in Section 4.3. To help users identify strong links, link thickness is scaled according to C-Values.

Since there is a C-Value between all course pairs, this has the potential to introduce significant visual clutter. To avoid clutter, we allow users to filter out weaker graph edges by thresholding based on C-Value. By default, without a determined threshold value, the visualization shows the minimum spanning tree with maximum edge weights of the node-link diagram. In other words, for each course (node), the chosen edge that connects it to the diagram is the one with the most correlation (weight).

This visualization also shows how much of a "core" role each course plays and the typical order in which students take the courses in the curriculum. These two pieces of information are encoded into the courses' horizontal and vertical positions in the diagram.

The horizontal position of a node indicates the average time at which the course is taken by the student population (from left to right). On the vertical dimension, courses that have the highest total correlation to all other courses in the major are placed at the center. Those courses are normally considered as the "core courses". Courses that have less overall correlation with the rest of the curriculum are considered peripherals. They are gradually placed away from the center. eCamp allows users to control the number of "core courses". In essence, this number represents the maximum number of courses that can be placed exactly at the vertical center of the diagram. Guided by these layout rules, the rest of the layout process is simply an automated spring-force approach.

The flexibility of the node-link diagram allows us to expose various course-related measures in the visualization, encoded within the nodes. Given the importance of gender diversity in the education system [35], we opted for showing gender distributions for each course. Each node in the graph shows student gender distribution as a pie chart, in which the blue color represents the percentage of male students and the red color represents females. To support discovery of bottleneck courses (i.e., those with the most failures), the size of each node represents the normalized percentage of failures in that course. For example in Fig. 4, we can see that math courses tend to be the bottleneck for computer science students. To help users see more contextual information about each course, when a user clicks on a node, extra information such as the exact number of students and grade distributions are also shown.

In this visualization, we can find insights to many curriculum design questions. For instance, one example is quality of student preparation—how students are progressing through the general education courses before they reach the department's core classes. Another example is whether the gate keeping courses, which will typically appear as bottleneck courses, are indeed appearing at appropriate locations as how the faculty has envisioned.

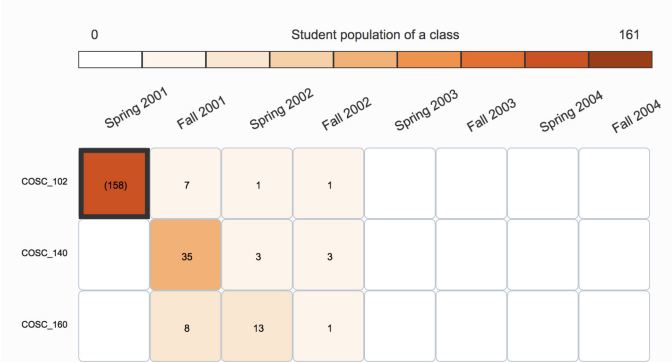


Fig. 5. A path matrix showing three courses of the computer science major in the span of 8 semesters. The path matrix changes color as users hover over different cells. In this example, the Data Structure and Algorithms course (COSC 140) is selected. Therefore, the color of all cells represent the number of students who took those courses and had previously taken the selected class.

5.3 Student Flow Through Classes

A course as defined in a curriculum only abstractly exists in the catalog. The actual instances of a course are those classes offered in different semesters. Although some core classes are offered every semester, most classes are not.

Student flow on a per semester granularity shows how a curriculum is executed. Since students take multiple classes per semester, the exact student paths on a per-semester time scale cannot be studied on a per-course level.

As described in Section 4.4, the collective paths that students take can be modeled as a weighted multilayer graph. We present a novel dynamic matrix to show the relationships within this graph. We call the resulting visualization a path matrix.

Fig. 5 shows a path matrix for three courses of the Computer Science major (from Spring 2001 to Fall 2004). The columns in the path matrix represent different semesters. Each row is a course. Each cell is a class in a semester.

Hovering over a cell selects the corresponding class. We call the selected class the *reference class*. Cells that correspond to the reference class' co-taken classes and post-taken classes are also highlighted. Again, co-taken classes are those classes that are typically taken in the same semester by students who are taking the reference class. Similarly, post-taken classes are classes that are typically taken in subsequent semesters by students in the reference class. The highlight colors in the co-taken classes and post-taken classes correspond to how many students in the reference class have taken the co-taken and post-taken classes. The number of students is also shown inside each cell.

As described in Section 4.4, the co-taken and post-taken relationships are modeled as edges of a multilayer graph, which are not easy to visualize because of their multidimensional structure. In a path matrix, we essentially use interactivity to substitute the complexity of visualizing an extra dimension. By hovering over different cells, users see varying groups of co-taken and post-taken classes. Note that visualizing the post-taken relationship is essentially showing interlayer edges in a multilayer graph.

The first outcome of the path matrix is helping administrators determine whether students are indeed taking classes according to intended requirements. However, in addition

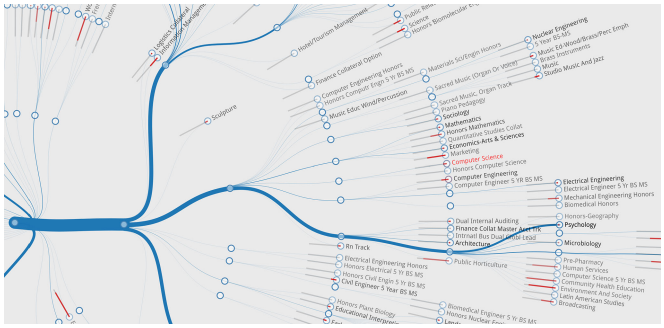


Fig. 6. This branch of the radial graph contains the university's computer and electrical engineering majors. These majors split apart from most other majors by the end of the 1st semester. Additionally, the psychology branch can be seen with a large number of students flowing through.

to showing raw student paths, the path matrix enables further queries on top of the model.

In the path matrix, we can filter by a particular group of students. For example, when hovering over a class, instead of selecting all of the students in that class, we can select students who got a low grade in that class, and further visualize those students' average grades in post-taken classes. The same notion can be applied towards filtering successful students. This filtering allows one to see how students' success/failure in each class affect the curriculum.

The student flow in the path matrix can be compared to the catalog. Although the catalog was not included in our original data, we manually collected pre-requisite and co-requisite information from the computer science catalog and added the information to the path matrix. When a reference class is selected, we annotate the rows that correspond to pre-requisite or co-requisite courses with a label.

5.4 Analytics Results

5.4.1 *Major Exploration Advising*

Students who wish to explore different options before choosing a major must be made aware of how the courses they choose to take limit their options of which majors they may pursue. As an example, using campus-level student flow visualization, Fig. 6 shows that students who are potentially interested in electrical engineering have a very limited time in which to commit to it, otherwise their graduation might be delayed. As another example, Fig. 7 shows that students have five semesters to choose between industrial engineering and mechanical engineering, and still graduate on time. Many of the discoveries from the campus-level student flow visualization call into question the university's one-size-fits-all policy that require students to declare a major after 45 credit hours. It is obvious that the amount of time allowed should be tailored according to students' academic interests.

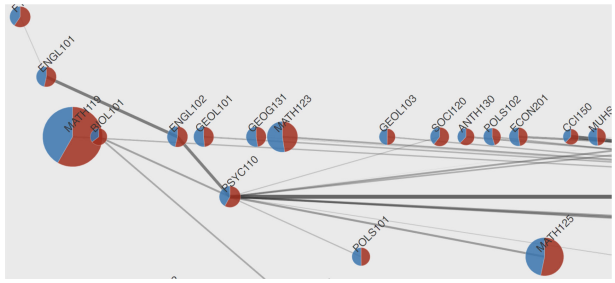


Fig. 8. The beginning of the communication studies curriculum. College Algebra (MATH 119), can be seen as the first bottleneck, although it has low correlation with the first core course (PSYC 110).

5.4.2 *Major Mobility Advising*

Another common advising task is helping students who wish to change majors. Consider a third-semester computer engineering student who comes to the adviser and expresses a desire to change majors due to a lack of interest in continuing computer engineering. Instead of just relying on experience, the adviser can use the campus-level student flow visualization to explore options along with the student.

First, the adviser locates the path from the root node to the node of depth 3 that contains computer engineering. Depth 3 corresponds to the third semester. Fig. 6 shows that path. Then, the adviser records each of the child majors from this node, and presents them to the student. Since at this point these majors have overlapping coursework, the student should be able to switch to any of them easily.

Consider another hypothetical situation. What if the computer engineering student cannot pass the second-semester physics course? In this case, it is likely in the student's best interests to change majors.

Even though the campus-level visualization can be used again to determine which majors would be a good fit for the student, the adviser must also consider the student's problems with Physics. Using the curriculum-level diagrams for each of the majors identified, the adviser notices that in the sociology major the physics course is not close to the core courses, and recommends to the student that he or she consider switching to it.

5.4.3 *Per-Major Dependency on External Courses*

eCamp's ability to show overall course correlations at a department level has led to discovering various patterns about their curricula.

When we look at a curriculum visualization, one of the first patterns that can be seen is the location of bottleneck courses. For example, Fig. 8 shows the initial courses in the communication studies major. We can easily see that most of the students in this major that take College Algebra

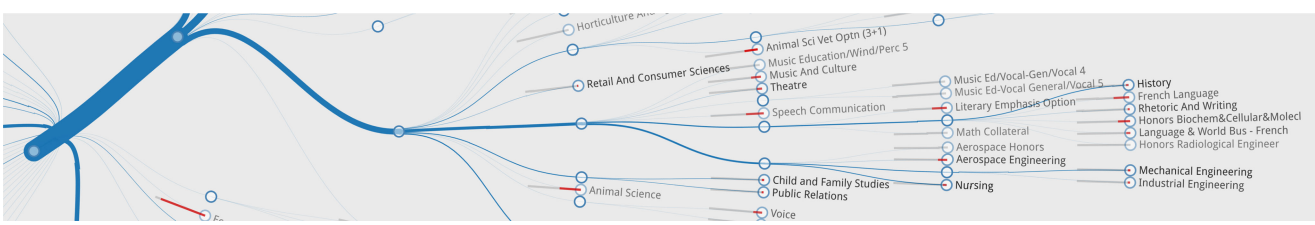


Fig. 7. This branch of the radial graph shows the path towards mechanical engineering and industrial engineering. Students have until the fifth semester to choose between the two.

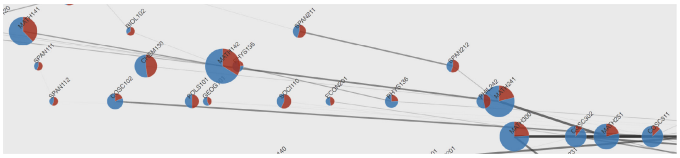


Fig. 9. Course correlations between the Calculus sequence (MATH 141, 142, 241) and the core courses. The core courses can be spotted by the straight horizontal line and high correlation between them (bottom-right).

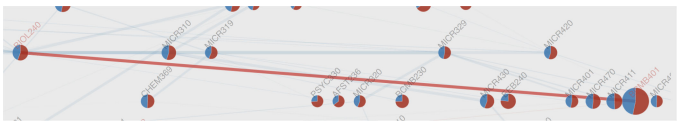


Fig. 10. A thick edge with high correlation is highlighted in red between Biochemistry I (BCMB 401) and General Genetics (BIOL 240). General Genetics is a core course in this curriculum, while Biochemistry I is a peripheral course that, is typically taught in later semesters.

(MATH 119), fail. However, we can also see that this course does not have a strong correlation with the core courses of the major (specifically, with PSYC 110). This calls into question the status of the course. If College Algebra is indeed important in the major, then maybe more introductory algebra classes should be required before taking it.

In the CS major, Fig. 9 shows that the Calculus sequence as a whole demonstrates correlation with two of the computer core courses, Linear Algebra (MATH 251), and Discrete Mathematics (COSC 311). When seeing general education sequences that affect student success in a major, the authors feel that it presents a good opportunity to encourage collaborations between the major-level and university-level student support infrastructures. We believe, whether these courses should be core courses in the CS curriculum is an interesting retention question, since they do not correlate with student success in non-theoretical CS courses. Also, many CS graduates will not engage in tasks requiring theoretical CS knowledge in their future jobs.

Another example of strong course connections is for microbiology students between two courses: Biochemistry I (BCMB 401) and General Genetics (BIOL 240). The connection between the two is highlighted in red in Fig. 10. These two courses seem different on surface, yet grades show a strong correlation in student performance in these two courses. Discussion with microbiology students revealed that although the two topics do differ in subject matter, they both require similar thought processes, placing emphasis on critical thinking and understanding over memorization. While this finding is practically just "how things are" to

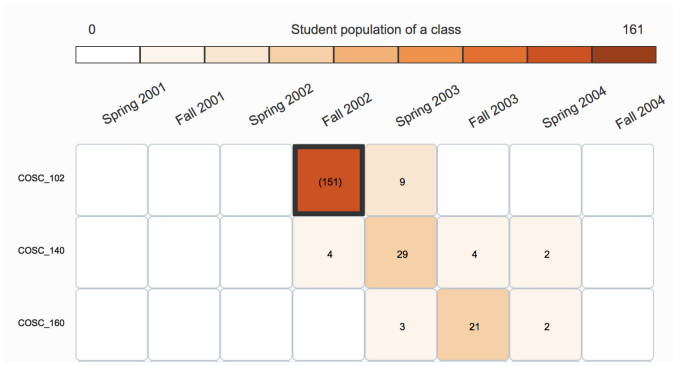


Fig. 12. Typical path from COSC 102 to COSC 160. Most students take these courses in consecutive semesters. However, some students have taken COSC 140 and COSC 160 in the same semester.

senior microbiology students, new students as well as campus-level advisers are unaware of these correlations.

5.4.4 *Comparing Curriculum Designs*

The department student flow also helps discover the why behind some campus-level patterns. For example, in the campus-level visualization (Fig. 6), we can see that students of many majors have the option of moving to the psychology major up until their fourth semester. In constrast, changing majors to computer science must happen by the second semester if students want to graduate on time.

The reason for this contrast cannot be determined from within campus-level student flow visualization alone. However, when we look at the department student flow visualization for the psychology major (Fig. 11), we can see that the core courses are towards the end of the major and students coming from other majors can fit in easily. In contrast, as we saw in Fig. 4, computer science curriculum have the gate keeping courses and core courses starting very early. This shows how the psychology major is more welcoming, and could be a potential reason for why so many students graduate as psychology majors.

5.4.5 *Student Progression*

When we viewed eight semesters of the computer science major's student flow through classes, we saw many important relationships between courses that were seldom noticed before. For example, we noticed that a typical path for students is taking COSC 102, COSC 140, and COSC 160 in consecutive semesters. This can be seen in Figs. 5 and 12 in different semesters. However, the matrix also tells us that some students have taken COSC 140 and COSC 160 in the

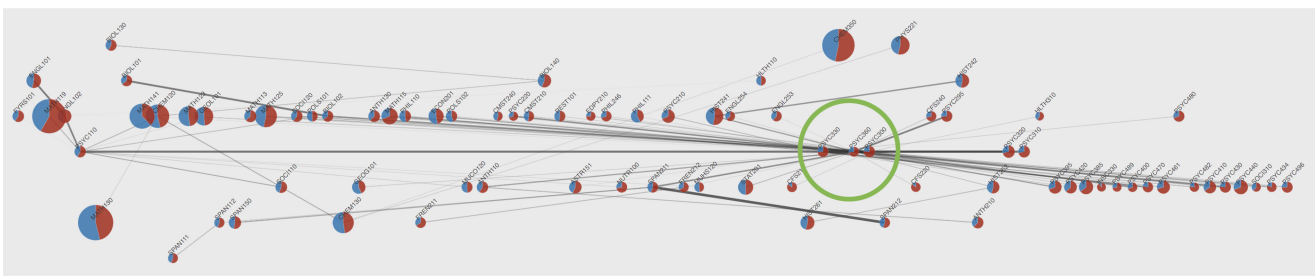


Fig. 11. Overview of the Psychology major. The majority of the core courses in the major are towards the end of the curriculum (circled in green). In contrast to the computer science major, most of the classes in Psychology comprise of female students.

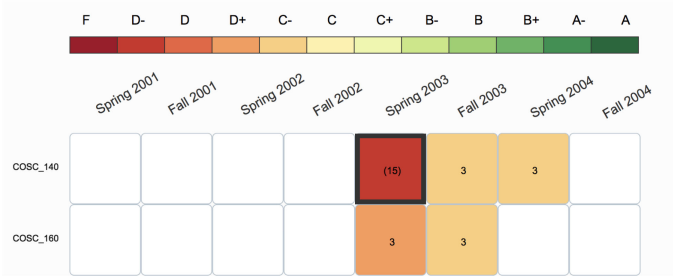


Fig. 13. A section of a path matrix, showing average grades for those who failed in COSC 140. In this filtered path matrix, the red color corresponds to an average grade close to F. The colormap can be seen at the top of the image.

same semester. When we filter the matrix to show failures and average grades (Fig. 13), we can see that students who failed at COSC 140 but had taken COSC 160 in the next semester, generally did better than those who took both classes in the same semester. This suggests that perhaps COSC 140 should be a pre-requisite of COSC 160.

Similarly, Fig. 14 shows that, students who failed in Data Structures (COSC 302), also performed poorly in Systems Programming (COSC 360). Looking at other semesters, we saw that failing Data Structures was one of the most catastrophic events in the path matrix, suggesting its importance as a pre-requisite. By looking at the left hand side labels of the matrix, we can see that Data Structures is now a pre-requisite of COSC 340, and COSC 360. However, at the time that our data was collected, this was not the case and students had taken the courses in the same semester.

We can also filter the path matrix based on success. Fig. 15, shows the average grade of students who got a grade of C or better in COSC 140, in Spring 2002. We can again see that students who took the 100 level classes consecutively, generally performed better. Hovering over other semesters, we noticed that students who do well in COSC 140 (Data Structures and Algorithms I), generally perform better in most other courses. However, we also observed that COSC 311 (Discrete Structures) had lower grades compared to other courses, regardless.

6 Domain User Feedback

Here we present more detailed observations derived by two domain experts and feedback from graduate students.

The first domain expert was our department head, who is intimately familiar with Electrical Engineering's

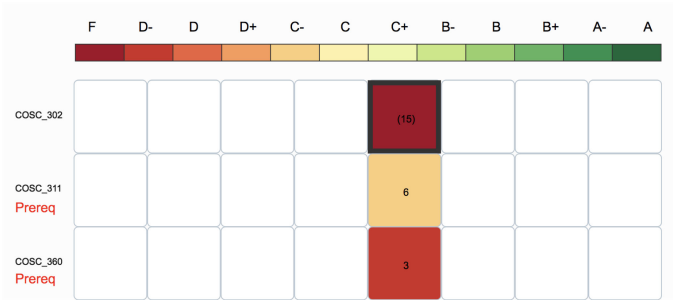


Fig. 14. A path matrix showing how three students who took COSC 302 and COSC 360 and failed in the first, also failed in the latter. While this happened in 2003, the new curriculum has indeed made COSC 302 a pre-requisite of COSC 360.

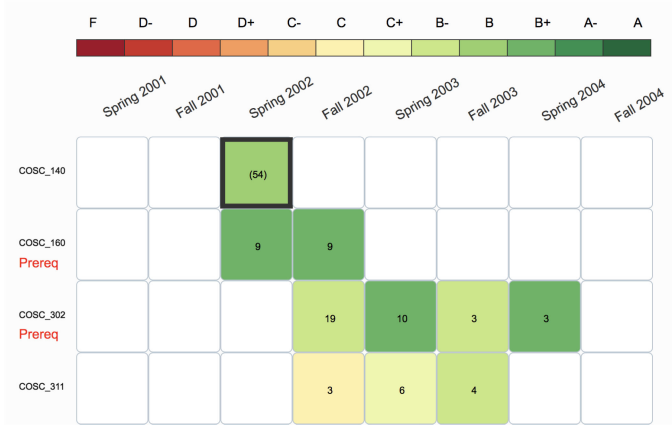


Fig. 15. A path matrix filtered by success. The matrix shows that successful students mostly take COSC 140, COSC 160, COSC 302, and COSC 311 in a sequence, while students who take the two latter courses in the same semester usually get lower grades.

curriculum. The second is a faculty member who recently served as Vice Provost of the university, whose priorities are improving student retention and time-to-graduation. Both began to ask questions on a university-wide and per-major basis, that they had not previously considered.

The former Vice Provost felt that the per curriculum visualizations are useful for evaluating majors in terms of how welcoming they are to students switching to them. For example, she mentioned that Classical Civilization is often considered a found major, where students who graduate with this major did not enter the university planning to do so. In such a major, core courses would ideally be very late in the curriculum, which was the case for Classical Civilization. In contrast, engineering departments prefer for students to commit very early, so it would be best for their core courses to be much earlier in the curriculum.

She thought that the radial graph was interesting to university-wide administrators as it showed major branches of study available in the university. Specifically, she saw four main arms of majors and noted that the number of students graduating from each of these arms was very uneven. This led her to ask questions regarding how the university is distributing resources, and whether or not this distribution matched the goals of the university.

She also felt that first-year advisers' work would benefit from access to the campus-level student flow visualization. Specifically, she saw that the Journalism and Communications majors maintained shared curricula until very late in the student career. This means that students are likely to choose one of these majors before they have taken courses which would help them determine which major is best suited to their interests. Hence, it is important for first-year advisers to make sure that students are informed that they should keep an open mind about which major they like the most, as it should still remain possible to switch between these majors very late into the curriculum.

Our department head examined the curriculum student flow diagram for Electrical Engineering, and saw patterns that he had expected and patterns that he had not. One finding was that some of the courses showing strong grade

correlations had the same instructor. He expressed a preference that course success be independent of instructors and instead be driven by the course's material. Additionally, he expressed surprise that the courses meant to serve as gatekeeping courses for Electrical Engineering did not seem to significantly affect the remaining curriculum, which led him to wonder why that was the case.

Based on feedback from students, we learned that personal experience seems to play a role in helping a user better identify, and validate patterns in the visualizations. To students, the ability to validate seems so important that we feel familiarity with the university affects how valuable a user may find the student flow visualizations.

For example, a student in computer science used the path matrix and quickly noticed that successive takes of the first three courses in computer science lead to better grades over time, which corroborates with his personal experience that those skills can be learned as long as the proven structure is followed. Another graduate student in computer engineering noticed that computer engineering has a curriculum structure that is more spread-out and less organized than that of computer science, however. The visualizations also showed her that the gender diversity problem in computer engineering is significantly worse than that in computer science. One can then hypothesize how well structured a curriculum is, may affect a department's ability to retain a more diverse student population.

Regarding improvements to eCamp, the former Vice Provost noted that it would be useful to see how flow differs between students with different financial backgrounds. She noted that students from low-income backgrounds are considered to be at higher risk of not graduating, and it would be interesting to be able to see where these students are typically struggling. Additionally, she suggested building tools to predict when students are changing majors and what majors they are changing to, as this could hint at why so many students take longer than 4 years to graduate.

7 Conclusion and Discussions

In this paper, we have taken a data science approach to integrate and make sense of previously disparate electronic student records. Our framework models relationships amongst multiple types of entities, in order to visualize student flow at campus, departmental, and per-semester levels. Our system, eCamp, enables university personnel and students to ask and answer complex questions using the data.

As future work, we'd like to build a deeper set of analytics using more contextual information, by expanding eCamp to incorporate additional data sources such as instructor information of each class, student financial aid information, and student admission information (such as SAT scores).

In addition, based on our visualization results, for future work one can consider more graph-centric measures and graph comparison algorithms to unravel even more complex relationships at a university. We believe the use of *time* has played a key role in our models and suggest that future work consider including that aspect as well.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments, encouragement, and recommendation. The authors are supported in part by US National Science Foundation Awards OCI-0906324, CNS-1629890, and the Engineering Research Center Program of the US National Science Foundation and the DOE under NSF Award Number EEC-1041877.

References

[1] E. G. Gee, "An open letter to college and university leaders," (2013). [Online]. Available: http://www.acenet.edu/news-room/Documents/An-Open-Letter-to-College-and-University-Leaders.pdf, Accessed on: June 2, 2018.

[2] L. Soares, "Post-traditional learners and the transformation of postsecondary education: A manifesto for college leaders," *Amer. Council Educ.*, vol. 118, pp. 1–18, 2013.

[3] W. S. Swail, "A different viewpoint on student retention," *Higher Learning Res. Commun.*, vol. 4, no. 2, pp. 18–25, 2014.

[4] V. Tinto, "Promoting student completion one class at a time," in *Proc. Symp. Retention 360 Conf.*, 2010.

[5] M. Raji, J. Duggan, B. DeCotes, J. Huang, and B. V. Zanden, "Visual progression analysis of student records data," in *Proc. Workshop Vis. Data Sci.*, Oct. 2017.

[6] P. Attewell, S. Heil, and L. Reisel, "Competing explanations of undergraduate noncompletion," *Amer. Educational Res. J.*, vol. 48, no. 3, pp. 536–559, 2011.

[7] M. R. Clark, "Negotiating the freshman year: Challenges and strategies among first-year college students," *J. College Student Develop.*, vol. 46, no. 3, pp. 296–316, 2005.

[8] J. Grann and D. Bushway, "Competency map: Visualizing student learning to promote student success," in *Proc. 4th Int. Conf. Learning Analytics Knowl.*, 2014, pp. 168–172.

[9] G. D. Kuh, J. Kinzie, J. H. Schuh, and E. J. Whitt, *Student Success in College: Creating Conditions That Matter*. Hoboken, NJ, USA: Wiley, 2011.

[10] M. J. Lutz, J. R. Vallino, K. Martinez, and D. E. Krutz, "Instilling a software engineering mindset through freshman seminar," in *Proc. Frontiers Educ. Conf.*, 2012, pp. 1–6.

[11] R. Mazza and V. Dimitrova, "Generation of graphical representations of student tracking data in course management systems," in *Proc. 9th Inf. Vis. Conf.*, 2005, pp. 253–258.

[12] R. Mazza and V. Dimitrova, "Visualising student tracking data to support instructors in web-based distance education," in *Proc. 13th Int. World Wide Web Conf. Alternate Track Papers Posters*, 2004, pp. 154–161.

[13] U. Rueda, M. Larrañaga, M. Kerejeta, J. A. Elorriaga, and A. Arruarte, "Visualizing student data in a real teaching context by means of concept maps," in *Proc. Int. Conf. Knowl. Manage. I-Know*, 2005.

[14] H. Siirtola, K.-J. Raiha, and V. Surakka, "Interactive curriculum visualization," in *Proc. 17th Int. Conf. Inf. Vis.*, 2013, pp. 108–117.

[15] S. Gama and D. Goncalves, "Visualizing large quantities of educational datamining information," in *Proc. 18th Int. Conf. Inf. Vis.*, 2014, pp. 102–107.

[16] D. Wortman and P. Rheingans, "Visualizing trends in student performance across computer science courses," *ACM SIGCSE Bulletin*, vol. 39, no. 1, pp. 430–434, 2007.

[17] B. Johnson and B. Shneiderman, "Tree-maps: A space-filling approach to the visualization of hierarchical information structures," in *Proc. IEEE Vis.*, Oct. 1991, pp. 284–291.

[18] D. H. Huson, D. C. Richter, C. Rausch, T. Dezulian, M. Franz, and R. Rupp, "Dendroscope: An interactive viewer for large phylogenetic trees," *BMC Bioinf.*, vol. 8, no. 1, 2007, Art. no. 460.

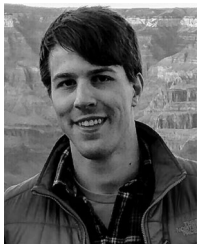
[19] H. Zhang, S. Gao, M. J. Lercher, S. Hu, and W.-H. Chen, "Evolview, an online tool for visualizing, annotating and managing phylogenetic trees," *Nucleic Acids Res.*, vol. 40, no. W1, pp. W569–W572, 2012.

[20] Y. Zhu, L. Sun, A. Garbarino, C. Schmidt, J. Fang, and J. Chen, "PathRings: A web-based tool for exploration of ortholog and expression data in biological pathways," *BMC Bioinf.*, vol. 16, no. 1, 2015, Art. no. 165.

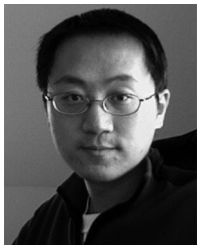
[21] J. M. Cullen and J. M. Allwood, "The efficient use of energy: Tracing the global flow of energy from fuel to service," *Energy Policy*, vol. 38, no. 1, pp. 75–81, 2010.
[22] H. Alemasoom, F. Samavati, J. Brosz, and D. Layzell, "EnergyViz: An interactive system for visualization of energy systems," *Visual Comput.*, vol. 32, pp. 403–413, 2016.
[23] P. Xu, Y. Wu, E. Wei, T.-Q. Peng, S. Liu, J. J. Zhu, and H. Qu, "Visual analysis of topic competition on social media," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 12, pp. 2012–2021, Dec. 2013.
[24] Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu, "OpinionFlow: Visual analysis of opinion diffusion on social media," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 12, pp. 1763–1772, Dec. 2014.
[25] E. M. Reingold and J. Tilford, "Tidier drawings of trees," *IEEE Trans. Softw. Eng.*, vol. SE-7, no. 2, pp. 223–228, Mar. 1981.
[26] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, "Multilayer networks," *J. Complex Netw.*, vol. 2, no. 3, pp. 203–271, 2014.
[27] S. Rufiange and M. J. McGuffin, "DiffAni: Visualizing dynamic graphs with a hybrid of difference maps and animation," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 12, pp. 2556–2565, Dec. 2013.
[28] K. Stein, R. Wegener, and C. Schlieder, "Pixel-oriented visualization of change in social networks," in *Proc. Int. Conf Advances Social Netw. Anal. Mining*, 2010, pp. 233–240.
[29] F. Beck, M. Burch, S. Diehl, and D. Weiskopf, "The state of the art in visualizing dynamic graphs," in *Proc. Eurographics Conf. Vis.*, 2014.
[30] R. Sadana, T. Major, A. Dove, and J. Stasko, "OnSet: A visualization technique for large-scale binary set data," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 12, pp. 1993–2002, Dec. 2014.
[31] Ellucian, "Banner student information system for higher education," 2018. Accessed on: Apr. 8, 2018, [Online]. Available: https://www.ellucian.com/Software/Banner-Student/
[32] D. V. Steward, "The design structure system: A method for managing the design of complex systems," *IEEE Trans. Eng. Manage.*, vol. EM-28, no. 3, pp. 71–74, Aug. 1981.
[33] P. J. Clarkson, C. Simons, and C. Eckert, "Predicting change propagation in complex design," *J. Mech. Des.*, vol. 126, no. 5, pp. 788–797, 2004.
[34] R. Keller, C. M. Eckert, and P. J. Clarkson, "Matrices or node-link diagrams: Which visual representation is better for visualising connectivity models?" *Inf. Vis.*, vol. 5, no. 1, pp. 62–76, 2006.
[35] B. A. Barres, "Does gender matter?" *Nature*, vol. 442, no. 7099, 2006, Art. no. 133.

Mohammad Raji received the BS and an MS degree in computer engineering from Razi University, Iran, in 2008 and 2012 respectively. He is working toward the PhD degree at the University of Tennessee, Knoxville. His research interests include web-based data visualization systems, large scale visualization and deep learning.

John Duggan received the BS degree in computer science from the University of Tennessee, Knoxville, where he is currently working toward the PhD degree. His research interests include large scale visualization, collaborative visualization, and mixed reality visualization.



Blaise DeCotes received the master's degree in computer science from the University of Tennessee, in 2014. His research focused on the analysis and visualization of large and unique datasets, including national park species locations, power grid sensors, and collegiate university records. Currently, he works as a developer for PhishLabs, a cyber security threat detection and mitigation company located in Charleston, South Carolina.



Jian Huang received the BEng degree in electrical engineering from the Nanjing University of Posts & Telecom, China, in 1996, and the MS and PhD degrees in computer science from the Ohio State University, in 1998 and 2001, respectively. He is a professor with the EECS Department, University of Tennessee, Knoxville. His research is on data analytics, visualization, and parallel systems. He was a recipient of DOE Early Career Principal Investigator Award.

Bradley T. Vander Zanden received the BS degree in computer science/accounting from the Ohio State University and the MS and PhD degree in computer science from Cornell University. He is a professor with the EECS Department, University of Tennessee, Knoxville. His research focuses on instructional technology and power-efficient software techniques for mobile devices. Earlier in his career, he focused on human-computer interaction.