Visualizing Trends in Student Performance Across Computer Science Courses

Dana Wortman
University of Maryland, Baltimore County
1000 Hilltop Circle
Baltimore, MD 21250
danawortman@umbc.edu

Penny Rheingans
University of Maryland, Baltimore County
1000 Hilltop Circle
Baltimore, MD 21250
rheingan@cs.umbc.edu

ABSTRACT

Student retention is an important topic in Computer Science departments across the country. Keeping strong students and helping struggling students perform better are two fundamental components of improving retention. Isolating the cause(s) of students leaving the major is an important area of research. We endeavor to explore this problem using a visualization tool to probe student data within the beginning course sequence in Computer Science. We would like to see what patterns exist amongst students, focusing on success, failure, and repetition patterns throughout the first three courses. Identifying these patterns can help isolate some of the causes of decreased retention within the department, allowing us to address individual projects, courses, or exams that may be causing students exceptional difficulty or loss of interest. Due to the large amount of data and the variety of students' paths through their courses, it is essential that a visualization be developed to represent the data. Using graph layouts, parallel coordinates, color-mapping, and interactive selection, users can explore and query the data. Users can discover patterns within the data by selecting subgroups of students and examining the event sequences to find patterns of success, failure, and repetition amongst those students. Departments can use this information to isolate profiles of students for retention, remediation, and recruitment efforts as well as identify areas of the curriculum or instruction that can be improved.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education; I.3.8 [**Computer Graphics**]: Applications

General Terms

Human Factors

SIGCSE'07, March 7–10, 2007, Covington, Kentucky, USA.

Keywords

CS1, CS2, Visualization, Student Performance, Retention

1. INTRODUCTION

With enrollment numbers dropping and jobs increasing, educators are seeing a growing divide between the number of students beginning Computer Science programs and those completing them. The Computing Research Association collects enrollment data from Computer Science departments around the country. The 2004 survey noted that there was almost a 70% decrease in enrollment over the past two years [1]. The fundamental question is: "why are we driving students away?" Beaubouef and Mason identify several of the reasons why students leave Computer Science. They point out that students possess insufficient skills to succeed and that faculty use inappropriate instructional techniques[2]. Seymour and Hewitt cite a lack or loss of interest and poor teaching as two of the major contributing factors to students' decisions to leave[12]. Seidman found that early identification and intensive intervention for students likely to leave college has the greatest effect at improving retention[11]. In order to realize this improvement, we must introduce students to the discipline early in the curriculum, demonstrate enthusiasm for the subject [12], develop related skills early so that they can succeed [11], and use techniques in the classroom and beyond that will encourage learning and refine skills.

In order to improve retention, we are interested in uncovering patterns of success, failure, and repetition of students enrolled in our Computer Science department. Of particular interest is understanding what happens with the large number of students who repeat early courses in the major, if they continue to struggle or are more successful later. Due to the large number of students and the diversity of their progress through their courses, it is difficult to identify larger patterns without using visualization techniques to represent the data. Visualization allows us to see the entire dataset, use visual cues to identify important events, and summarize the data without looking at individual numbers or generated statistics. We use a basic structure borrowed from graph theory of nodes and edges, then build on that foundation by mapping performance to color and then provide a utility that allows the user to selectively filter groups of students to identify commonalities amongst students.

2. RELATED WORK

Using visualization to explore data is quickly becoming

a well-known technique for identifying patterns, especially in large datasets. Ware emphasizes that visualization provides the ability to comprehend massive datasets, perceive emergent properties, and understand large- and small-scale features[13]. Marcus et al. use a collection of 3-dimensional glyphs, color, and height to map a large dataset, enabling users to see patterns within the data based on several attributes[7]. Havre et al. developed a novel idea of using a river metaphor for visualizing topical patterns within news stories - incorporating color to represent similar topics and height to represent the frequency of a topic[4]. We map data to glyphs, color, and component widths to help the user identify interesting features within the dataset.

Interactivity has also played a role in allowing users to detect patterns within data. Bederson and Shneiderman identify dynamic queries as useful tool for exploring large databases[3]. Timeboxes were introduced by Hochheiser and Schneiderman, enabling users to graphically map queries into the dataset via regular boxes[5]. De Pauw et al. [9] and Plaisant et al. [10] use a "details-on-demand" approach by providing user-interaction and querying at several hierarchical levels. Munzner et al. introduced the idea of "guaranteed visibility" within large datasets by providing a mechanism to ensure that the entire dataset is visible[8]. We exploit interactive querying and guaranteed visibility to allow users to visualize selected groups of students within the scope of the entire dataset.

3. APPLICATION

Our primary goal in developing this application was to explore the collection of grades from Computer Science I (CS1), Computer Science II (CS2), and Data Structures (CS3), identifying similarities and differences amongst successful and unsuccessful students. Most of the students taking these three courses are Computer Science or Computer Engineering majors. Our department has instituted a "Gateway" requiring students to earn a B or better in CS1 and CS2 to continue in the program. Before embarking on this project, we had a vague notion that there were a significant number of repeated attempts at these courses. However, we had no idea what patterns exist amongst these students as they retook these courses, such as whether they were more or less successful on subsequent attempts or even which topics in these courses prove problematic for them.

3.1 Data

Project grades, examination grades, and the final course grades were collected from instructors for each of these courses over four semesters. Each record in the dataset includes a unique identifier, grade in each of the course projects and exams, and a final grade for the course which included any laboratory assignments, quizzes, or curves applied to the class. In total, there are approximately 1700 individual students involved in the four semesters with over 20,000 individual grades. In visualizing these data, we hope to identify some of the problem areas for repeating students as well as some of the differences between successful and unsuccessful students. Ideally, we would like to see patterns that will help us improve the initial course sequence, helping students be more successful through a higher level of understanding.

3.2 Requirements

Since we are interested in eliciting patterns from the data using visualization, we must satisfy the following requirements. The visualization must allow the user to:

- Discover performance patterns amongst successful, struggling, and repeating students.
- Select groups of students based on performance criteria.
- Quickly and easily refine/expand a selection.

Supporting these three criteria is paramount to demonstrating the usefulness of the tool.

4. APPROACH

Our primary inspiration was a basic graph layout with nodes representing specific events, edges between those nodes representing students moving from one event to the next, and grades earned are mapped to color. We use line width to represent the number of students currently selected moving between two events and the width of a node represents the total number of students passing through that event.

4.1 Node Layout

We start with a temporal layout with some modifications inspired by parallel coordinates[6]. Starting with the basic idea of a temporal sequence of events from left to right, we then depict grades for each event from top to bottom (A to F). Figure 1 labels each of the components. Project, exam or course grades represent events. Each event is depicted by a single node. Related nodes are vertically aligned. Since the temporal relationship between exams and projects is unknown and differs between courses and semesters, we use two different event sequences to depict them. Each course has a "dummy" node that represents the beginning of the course. Previous course data link to the dummy node for the following course. Projects are horizontally arranged in an arc connecting the dummy node from the current course to the dummy node for the subsequent course. Projects are positioned vertically above the course grades and dummy nodes, exams are below. The grade each node represents is mapped to the color of that node. Users can specify the color for grades A and F and a linear interpolation generates the mapping in between. All figures in this paper map white to A and dark-blue to F, preserving the mappings in black and white images. The width of these nodes is linearly correlated to the number of students passing through that node:

$$\frac{NumberOfStudentsInNode \times MaxLineWidth}{TotalNumberOfStudents} \quad (1)$$

We incorporate guaranteed visibility of the nodes by ensuring that the width of a node does not decrease beyond a minimum threshold.

4.2 Event Mapping

Representing a single student's progress through the three course sequence begins with generating two paths, one through the projects, course grades, and dummy nodes and the other through the exams, course grades, and dummy nodes. Each edge along the path represents the transition of that student from one event to the next. The user may choose to map the grade from the starting or ending node to the color of the edge, emphasizing the transition into or out of the

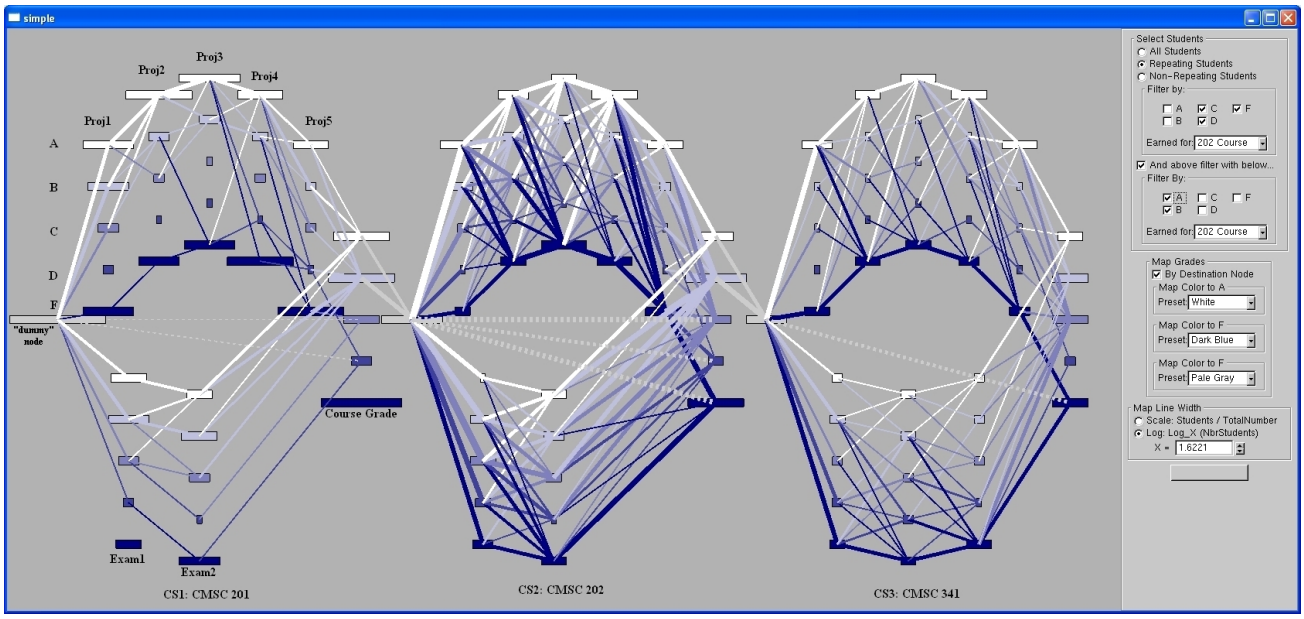

Figure 1: Repeating students who earned an D or F in CS2 and repeat it to earn an A, B, or C

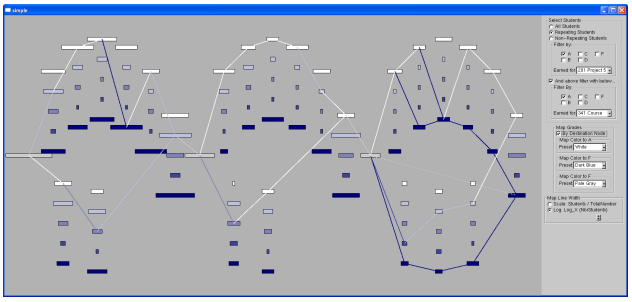

Figure 2: Single student progress through three courses, destination node used for color mapping.

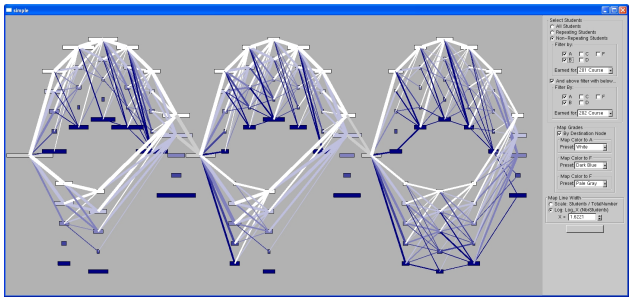

Figure 4: Non-repeating students who had an A or B in CS1 and an A or a B in CS2.

current event. Students who repeat a course are depicted by a stippled edge starting at the course grade and directed "backwards" (to the left) to the current course's dummy node. Figure 2 uses the destination node to color each edge leaving a node, showing the progress of a single student and her second attempt at CS3.

4.3 Aggregation

While it is interesting to see individual students' progress through the course sequence, in order to understand what trends may exist, we aggregate the student data based on similar event sequences. We use a single line to represent all the students who shared the same path between two events. The thickness of this line represents the number of students sharing this same event pair. The user can interactively select how this value is mapped to the width of the lines by choosing either the same linear equation as is used for the node-width mapping or by choosing a logarithmic relationship:

$$log_x(SelectedStudentsAlongEdge) \quad (2)$$

where the base x is specified as a floating point value greater than 1.0. In Figure 3, repeating students with a C, D, or F in CS1 and CS2 are displayed. The width of each line represents the number of students along each edge. Since the base of the log is close to one, it can be seen that only one student received an A in CS2 after "failing" either CS1 or CS2. It can also be seen from the width of the stippled lines that more students with C's repeated CS1 than CS2.

4.4 Selection

In order to select groups of students based on one or more points of similarity, we provide an interactive interface which supports primitive queries of the dataset. Users begin by selecting a category of student: all students, students who have repeated at least one course, or students that have not repeated any course. Then similarities within each category are queried by selecting an event (or two events) and the grade(s) to be included in the selection. Any students matching the query are displayed within aggregated lines. Students not matching the query are not displayed, but their presence is represented by the node widths which do not change.

Figure 4 is the result from a query selecting students who earned an A or B as the course grade for CS1 and either

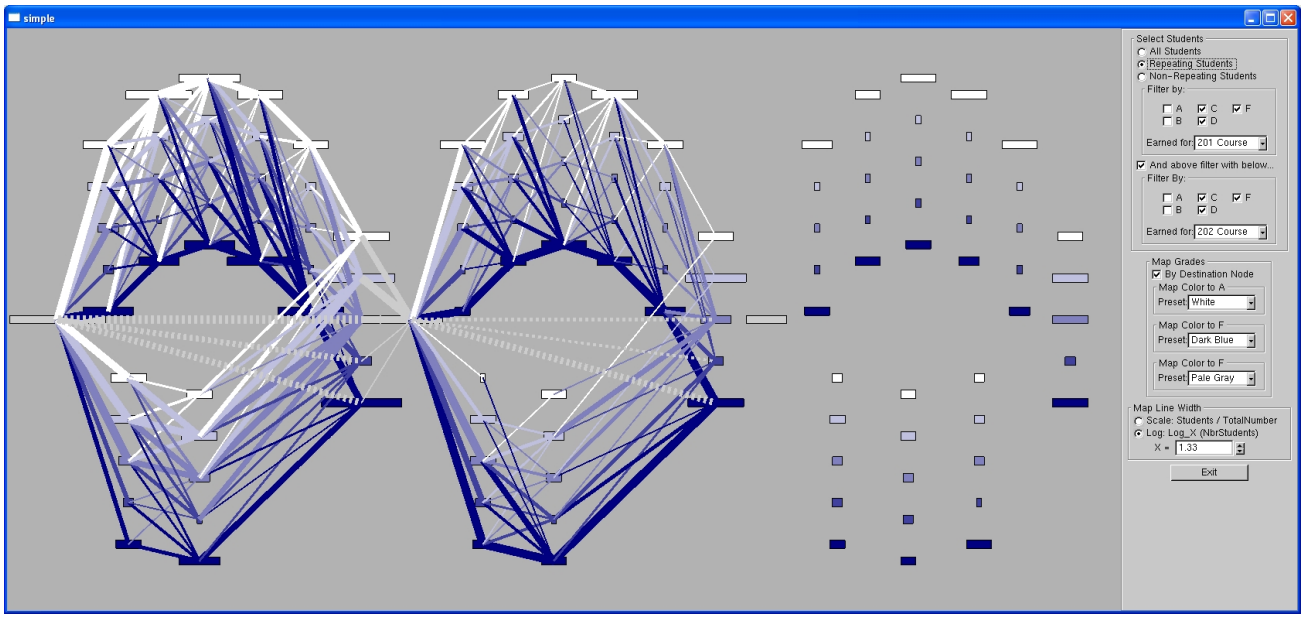

Figure 3: Repeating students who had a C, D, or F in CS1 and CS2.

an A or B for the course grade for CS2. These students are some of our strongest students, but it can be seen that at several points in their careers, some of them have had some difficulty. For example, Project 4 in CS1 seems more difficult than the other projects as there are more students earning low scores. Exam 1 in CS2 is also more difficult than Exam 2 as few students earned an A, most earning B's and C's and then improving their scores on the next exam.

5. RESULTS

Our goal in developing this project is to help students succeed within the Computer Science curriculum. In order to do this, we need to understand how "successful" students move through these three courses and then examine how other students perform.

5.1 Picture of Success

Figure 4 shows successful students who passed CS1 and CS2 on the first try with either A's or B's. Interestingly, with the exception of one or two individuals, these students also passed CS3 with a C or better on the first try. In general, these students perform well across the board, but there are several events that may require examination.

In CS1, Projects 4 and 5 show a decrease in scores from the previous projects, this is reasonable as Project 4 introduces simple dynamic memory and Project 5 uses a dynamic data structure. In CS2, these students continue to perform well, but score poorly on Exam 1 and struggle with Project 3 (which is dynamic memory, again). Even Project 5 (templates and exceptions) is more difficult than Project 4 (inheritance). In CS3, these students struggle with Projects 2 and 4. Project 1 is a warm-up project introducing no new material, so Project 2 tends to be a shock to some students. Project 4 usually introduces n-ary trees or similar dynamic-allocation, sorting, and searching problems.

These projects represent difficult concepts for every student, and obviously, even our "top" students struggle with some of them. However, since our "top" students are struggling, we are not providing students with the necessary background through previous projects, exams, laboratory activities, and instructional materials. We must also examine these projects for motivational content, developing "real-world" problems for students to solve keeps students interested and improves retention.

5.2 Perils of Defeat

Figure 3 depicts students earning a C, D, or F in CS1 and CS2, therefore repeating the course. Given that the dataset only covers four semesters, these students' CS3 data were not included. However, looking at the course grades for CS2, only a few (probably 2 or 3) students had the ability to move on to CS3 in the fifth semester, others would be required to repeat CS2 again. This indicates that repeating a course does not provide sufficient knowledge and understanding to succeed at the next level. The high number of repeats from the course grade of F indicates that quite a few of these students desired to try again, but have met with a similar fate. Based on these results, we should focus on developing a remediation program for students repeating these courses, identifying what difficulties they experienced and addressing those issues by providing additional structure or scaffolding to help them succeed.

5.3 Portrait of a Repetition

Figure 5 tells an interesting story of students repeating CS1 and then succeeding in CS2 (with or without repeating). These students successfully used their repetition of CS1 to improve the skills required to succeed in CS2. Upon repeating, most of them earned B's in CS1 and CS2. Another observation is that there are no students who earned an F in CS1 or CS2 in this selection, indicating that students who receive an F in CS1 find it difficult to pass CS2 within the next 3 semesters. Figure 5 also indicates that students who earn a C in CS1 have a good chance of being successful in the future, while students earning D's or F's

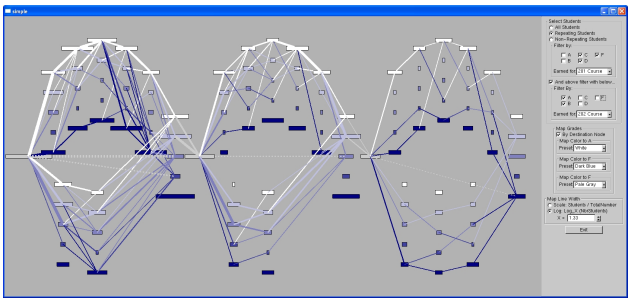

Figure 5: Repeating students who earned a C, D, or F in CS1 and then went on to earn an A or B in CS2

do not. This may also imply that the Gateway policy (of B or better) is not really necessary as students who earn a C move through the following courses successfully. Similarly to successful students, Projects 4 and 5 from CS1 and Exam 1 from CS2 seem to be significantly more difficult than the surrounding events, indicating that additional materials and preparation are necessary.

Figure 1 shows a selection of students who have repeated CS2, these students earned C's, D's or F's the first time around and have repeated it to earn an A or B. Most of these students did well in the previous courses, but fewer earned A's in CS1 than B's. Few of these students ended up with an A in CS2, either, the majority of them earning B's. Also, almost all of these students earned a C in CS3 (with and without repetition). Together with Figure 1, Figure 3 indicates that most students who repeat a course are either unsuccessful or are barely successful in the next course. This brings to light the necessity for a remedial program for repeating students.

6. FUTURE DIRECTIONS

This visualization technique only begins to address the "big" question of retention and how to improve it, but it does take a step forward. In the future, we plan to add techniques to provide automated pattern recognition, advanced queries, and increased dataset information including race, gender, and a more complete listing of educational events (like major histories and complete curriculum progress). With these components, we plan to examine when and how students enter the program, whether transfer students are at a disadvantage or advantage when transferring credit, whether gender or race play a role in course selection or success, and how course progress affects changing majors or leaving the school entirely. We also specifically ignored differences in teaching styles, testing techniques and instructor identity, but these are all areas where future work could identify additional patterns.

7. CONCLUSIONS

We developed a visualization technique to examine the introductory three-course sequence within our Computer Science department: CS1, CS2, and CS3. Our primary goal was to attempt to discover patterns within the successes, failures, and repetitions of students within these courses. By using a basic graph-structure enhanced by aggregation of data, color-mapping, interactive selection and data-mapped glyphs, we have identified some interesting patterns within the courses, indicating several problem areas to address. With this information, we can explore alternative strategies for improved educational practices, hopefully leading to an increase in retention and success of our individual students.

8. ACKNOWLEDGMENTS

The authors would like to thank Dr. Tim Oates, Dr. Yun Peng, Ms. Dawn Block, Mr. Mitch Edelman, Ms. Susan Evans, Mr. Dennis Frey, Mr. Daniel Hood, and Mr. Sa'ad Raouf for supplying the student grade data.

9. REFERENCES

[1] Computing Research Association. CRA taulbee survey. web, May 2004.

[2] T. Beaubouef and J. Mason. Why the high attrition rate for computer science students: some thoughts and observations. *SIGCSE Bull.*, 37(2):103–106, 2005.

[3] B. Bederson and B. Shneiderman. *The Craft of Information Visualization: Readings and Reflections*. Morgan Kaufmann, 2003.

[4] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. Themeriver: Visualizing thematic changes in large document collections. In *IEEE Transactions of Visualization and Computer Graphics*, Jan- Mar 2002.

[5] H. Hochheiser and B. Schneiderman. Visual queries for finding patterns in time series data. In *University of Maryland, Computer Science Dept. Tech Report, CS-TR-4365*, 2002.

[6] A. Inselberg and B. Dimsdale. Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *VIS '90: Proceedings of the 1st conference on Visualization '90*, pages 361–378, Los Alamitos, CA, USA, 1990. IEEE Computer Society Press.

[7] A. Marcus, L. Feng, and J. I. Maletic. 3d representations for software visualization. In *SoftVis '03: Proceedings of the 2003 ACM symposium on Software visualization*, pages 27–ff, New York, NY, USA, 2003. ACM Press.

[8] T. Munzner, F. Guimbretière, S. Tasiran, L. Zhang, and Y. Zhou. Treejuxtaposer: scalable tree comparison using focus+context with guaranteed visibility. *ACM Trans. Graph.*, 22(3):453–462, 2003.

[9] W. D. Pauw, D. Lorenz, J. Vlissides, and M. Wegman. Execution patterns in object-oriented visualization. In *Proc. Conference on Object-Oriented Technologies and Systems*, 1998.

[10] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, and B. Shneiderman. Lifelines: Using visualization to enhance navigation and analysis of patient records. Technical Report CS-TR-3943, 1998.

[11] A. Seidman, editor. *College Student Retention: Formul for Student Success*. American Council on Education and Praeger Publishers, 2005.

[12] E. Seymour and N. M. Hewitt. *Talking About Leaving: Why Undergraduates Leave the Sciences*. Westview Press, 1997.

[13] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann, 2004.