Abstract - CurricVis is a work-in-progress curriculum visualization application. The structure of a curriculum is modeled as a directed graph whose nodes represent courses and whose edges represent relationships between courses. Various graphical styles are used to encode details such as co-requisites and electives. Diagrams are automatically generated based on a curricular knowledge base. This approach has many benefits over manually-created visualizations: they require less maintenance and promote visual exploration of curriculum modifications. The CurricVis prototype has been used to generate visualizations of Computer Science curricula, including degree programs for majors and minors. The resulting visualizations have been valuable in advising current students as well as explaining the curriculum to prospective students. They have also been used to facilitate curriculum discussion in departmental committees. More user-friendly data entry and visualization generation interfaces are currently under development. Future work includes integration with Web-based course catalogs and student transcripts, path highlighting for computer-assisted advising, and the integration of more advanced information visualization techniques for large-scale curriculum visualization.

Index Terms – Curriculum visualization, information visualization.

INTRODUCTION AND MOTIVATION

Course catalogs are the tradition reference for course information, and whether printed or on the Web, they are efficient at conveying information about individual courses. However, their text-based descriptions are inefficient for imparting knowledge about the overall “shape” of a curriculum. The problems inherent in these textual descriptions manifest in different ways for different users, but they affect students, faculty, and administrators. New students are frequently overwhelmed with the number of options and are unable gain sufficient insight to make informed decisions. Upperclassmen have invested time in a curriculum, but they frequently need assistance identifying paths to satisfy graduation requirements within a set of constraints. New faculty members must invest significant time to understand a curriculum well enough to advise undergraduates, while faculty members who have been at an institution for many years may inaccurately remember revisions of courses and programs. Academic advisors benefit from having a consistent way of communicating fine-grained information about curricula, while administrators seek a high-level understanding.

Many of these problems can be alleviated through information visualization. Visualization can be used present information in a way that promotes understanding and insight for the human user [1]. Curriculum visualization is the application of information visualization to curricular data [2]. It is hoped that, through curriculum visualization research, educational institutions can be improved in a variety of ways. Visualizations can lead to improved communication, better decision-making by students and planners, and insight into patterns and conflicts.

APPROACH

CurricVis is a curriculum visualization application currently in development. The diagrams it generates are designed to highlight the flow of a curriculum, and so a directed graph representation is applied. The nodes of these graphs represent courses, and the edges represent relationships between courses. This leads to diagrams such as the one presented below.

FIGURE 1

The various graphical styles are used to encode information about the curriculum. The nodes representing elective courses are shaded, and those representing required courses are unfilled. The one rectangular node highlights a “hidden prerequisite”: it does not count on the minor, but it is a prerequisite of one of the electives (maths362). A solid arrow represents a prerequisite, and a dotted line indicates a prerequisite-or-parallel relationship. Not shown in Figure 1 are dashed lines, which are used to represent alternative prerequisites; these occur when there is a set of courses, any one of which satisfies the prerequisite requirement.

Using this visualization, a student can clearly see the paths to desirable courses. For example, students frequently ask how they can take cs345, a course on graphical user interface design. Figure 1 can be used to determine that a total of five courses must be taken first: cs120, cs121, cs124, maths161, and cs232. Furthermore, by finding the longest path between cs345 and its prerequisites, a reader can determine that satisfying all this course’s requirements can...
be done in three semesters, as long as cs121 and cs124 are taken concurrently. Anecdotal, gaining this insight from a purely textual description of the minor has proven to be quite difficult for students, but those who have used the visualization have quickly made this conclusion.

Program administrators can use the same visualization to analyze and improve curricula. For example, seeing the program as a whole can lead to desirable changes such as the elimination of hidden prerequisites. Using purely text-based descriptions of courses makes it more difficult to identify such structural problems.

SOFTWARE ARCHITECTURE

The CurricVis prototype adopts a three-tier software architecture and is implemented in SWI-Prolog (swi-prolog.org). The knowledge base represents curriculum data as a set of facts and rules. Using Prolog for the knowledge base simplifies the representation of complex information. The visualization generation layer constructs diagrams based on the content of the knowledge base. These graphs are drawn using the graphviz libraries [3]. Finally, the user interface layer constitutes the user interface. It allows the user to select a curriculum and then presents the generated visualization.

CURRENT STATUS AND FUTURE WORK

Diagrams generated with CurricVis have shown to be beneficial within my department. They have been useful in curriculum committee meetings to describe the structure of the undergraduate major, and the same diagrams have been used in posters, outreach events, and campus visits to explain the curriculum structure to prospective students. Furthermore, they have been useful during individual student advising sessions. However, two directions for improvement have been clearly identified to address the shortcomings of the prototype. These relate to the visualization methodology and the software interface.

There is important curricular information that is not yet incorporated into the CurricVis visualization methodology. For example, the number of credit hours for a course is not depicted, and the fact that some courses require departmental permission is not elegantly handled. There is also a problem introduced by “course equivalences” that are not explicit in the curriculum. For example, cs124 in Figure 1 is a course on discrete structures offered through Computer Science, but there is another course on discrete math offered by Mathematical Sciences. These courses are considered equivalent, but this rule is not explicitly stated in the individual course descriptions; it is only mentioned as an appendix to the academic programs. As a result, only some advisors are aware of the equivalence, and students are almost always ignorant of it.

The static diagrams generated by CurricVis are appropriate for use in printed materials, but the project goals would be better addressed by an interactive visualization. Currently, a single departmental major takes an entire printed page, and so juxtaposing it against another department’s infeasible. Augmenting the presentation and interaction modes has the potential to solve this problem. Research has begun on the use of dynamic graph drawing techniques and zoomable user interfaces [4] to manage the large amount of data.

The presentation layer is written in a pair of Prolog modules, but clearly, it is not reasonable to expect that end users should learn Prolog in order to use CurricVis. We are currently prototyping graphical user interfaces for interacting with the knowledge base, rule definitions, and visual semantics of the tool. These are being developed by following the usage-centered design process, specifically with the goal of redesigning CurricVis for deployment on the Web [5].

EXPECTED RESULTS AND ASSESSMENT PLAN

Once CurricVis has been redesigned in order to address the issues mentioned above, it will again be tested within my department. The knowledge base for this curriculum is already in place, and so the only required changes are in the visualization generation and presentation layers. The results of this testing will feed back into continued development of the tool, forming a spiral development model.

Once a limited-scale, single-department prototype passes internal tests, the tool will be shared with other departments. Usability testing will be employed to further refine the user experience. It is expected that such iterative development – design analysis, prototype development, and user testing – will lead to a tool that can be broadly disseminated outside of this tool’s host department and university.

REFERENCES