Visualizing Degree Progress

Brad Reyes, *Undergraduate Student, Stanford University*
Mark Schramm, *Undergraduate Student, Stanford University*

Abstract—In this paper, we try to create an efficient and effective visualization for degree progression and course fulfillment for students at Stanford University. To do this, we created a system that has four features: cumulative unit counts and projections, percentage of remaining units that need to be taken within the major, a credit/no credit meter, and a course suggestion system based on previously taken courses. Once we built this, we tested our hypotheses with a user test, comparing our new system to the one that is used by Stanford students today. We found that a unit projection system (pace), a calculation for the percentage of remaining units being within the major, and a course suggestion system were popular among students trying to understand their degree progress situation. Ultimately, a system like this and the results we have recorded can be the foundation for other systems trying to solve this problem, and hopefully they can learn from our mistakes and use the effective aspects of our project in their visualization. Creating a visualization for degree progression is such a relevant topic, and helping students create healthy, balanced, and fulfilling schedules for graduation is a noble goal that can improve the lives of students during their college career.

Index Terms—Data Visualization, progression, curriculum, graph structure, course suggestion, color, saturation, gamification

1 Introduction

While there has been a fair (but not huge) amount of research with regards to visualizing course curriculum, little has been done with regards to visualizing overall degree progression. In this paper, we will discuss a system we created to help visualize degree progression, and user test this system to see how effective our hypotheses about visualizing degree progression are so that future systems can build off of this. We will also discuss related works in the field, and future work that needs to be done.

We will first break down why this is unfortunate and make our case as to why it's an important problem to tackle, as well as talk about our solution at a high level.

- *Brad Reyes is currently an coterminal student in the Computer Science department at Stanford University, CA*
  E-mail: breyes28@stanford.edu
- *Mark Schramm is an undergraduate student in the Computer Science department at Stanford University, CA, and is a developer at SimpleEmotion, CA*
  E-mail: mschramm@stanford.edu

Manuscript written for CS448b for June 5, 2016

1.1 Problem

Students, at least in the context of Stanford, often must sift through the clunky tables and degree worksheets that are given to determine whether they graduate in a timely manner. While some may argue that this is not a problem since little to no students actually fail to graduate because of a miscommunication of information, the bigger issue that this argument fails to see is that checking whether a student is going to graduate should not be a stressful ordeal that takes more than ten minutes.

Thus, the real problem that we are trying to solve is the cluttered and ineffective way of that course requirements and degree progression are visualized. We believe that we can use the data that other degree progression tools use much more efficiently, effectively, and artistically.

Likewise, a subset of this problem is that even during a student's undergraduate career, and even after giving students the tools they need to see degree progression from a very high level, selecting courses and seeing how choices

of courses can affect a student's undergraduate career later can help ensure that a student graduates in a timely manner, something that tools now do not adequately do (this will be discussed more in section 2: Related Works).

1.2 Motivation

Some might say that making degree progression easier to understand and see for students is not a worthwhile project, since the extra minutes that students take understanding and parsing the progression and requirement data does not outweigh the trouble of studying and creating an entire system. However, we feel that the splash effects of using requirement data to its full potential can have a load of benefits that will easily outweigh the costs of creating a system like this.

First off, let us dispel the notion that some people may have that these extra minutes students use to look at degree progression are inconsequential. We believe , along with many other Stanford students, that checking on degree progression should not be an ordeal that takes a large chunk of time. As we will present in Section 2: Related Works, the current tools for degree progression, Axess and major degree worksheets, are complicated and sometime redundant. Axess in particular is so clunky that students often turn to outside help to ensure that they will graduate.

Likewise, students may be unaware of where they stand at any given point because they do not check their graduation standing regularly due to inefficient and ineffective visualizations of degree progression. As a result, a student may be unaware that the pace he or she is going at will lead to unbearable and stressful quarters in an environment that always pushes for excellence.

Finally, a solution to this problem would help specific groups more than others, specifically first generation students. When students do not have access to mentors and their parents cannot help them through the college process, things like checking on requirements can be all the more daunting. In fact, after six years, only 25 percent of first-generational students graduate, which is lowered to 11 percent for those students who are also low-income (Kahlenberg 2016). Eliminating information barriers is one way to alleviate this problem (Opidee 2015), and giving guidance through visualization to make sure these students set themselves up for success, we feel, is a worthwhile goal.

1.3 Solution and Hypotheses

The solution to this, which we will talk in much more detail in Section 3, is to create a system that more easily gives students a sense of their degree progression. We implemented four distinct features that we feel will help achieve separate but connected goals:
(1) Seeing cumulative unit counts while also projecting pace
(2) Giving a sense of how difficult future quarters will be
(3) Letting users know how many CR/NC (credit/no credit) classes they have used up to not exceed their limit
(4) Giving course suggestions based on requirements that need to be filled and classes already taken

We hypothesize that giving users these four features can give a student the confidence to say how well or poorly he or she is doing so far in his or her Stanford career. Likewise, we'll discuss how effective the visualization techniques or the algorithms/methods we used are towards this student confidence.

2 Related Work

First we can examine existing works that try and tackle this problem. The first is the system that Stanford students already used: Axess. Axess is a system that shows users their degree progress, including the requirements that they have already fulfilled for GERs (general education requirements). Axess is often criticized for having poor information density, having too many levels of indirection, and being a user experience nightmare. A screenshot of an entire page, as shown by Figure 1, shows how checking GER requirements can be more of a hassle than it should be.

To check major requirements, one has to go the the Stanford Engineering Handbook, which is a degree worksheet that a student can use to checklist all the requirements he or she has done. With this method, a student has to switch between manually entering his or her transcript information and looking up how to fulfill specific requirements.

Lastly, a project called Carta, a research project in the HCI department at Stanford, is hoping to alleviate this problem. After talking to one of its members, Tum Chaturapruek, we learned that Carta aims to display course prerequisites, student evaluations of classes, and degree requirements in one place. However, they are still early on in development, and we hope to help them with some user testing data with our system. They, in return, were able to get us prerequisite data, which we will use for our system.

Fig. 1. A screenshot showing how to see if a student has fulfilled a GER requirement. A student must click on the thinking matters dropdown to see if he/she has fulfilled the requirement.

In terms of research papers and articles, works related to visualizing degree progression and course suggestion can be broken down into seeing previous works on curriculum visualizations and progression visualizations. For example, curriculum can be visualized as nodes held together by edges that represent prerequisites and corequisistes, offering an easy way to see progression and recommend classes to take (Aldrich 2014). An example of this can be seen in Figure 2. It also describes how crucial it is to emphasize classes with higher edge degrees. This system, however, limits the freedom of students to see classes out of order (skipping prerequisites), something done by students fairly often. Dotted line and dashed lines can be used to show more information among relationships between the courses, which is a technique we can use to alleviate the problems with the first system we described (Gestwicki 2008).

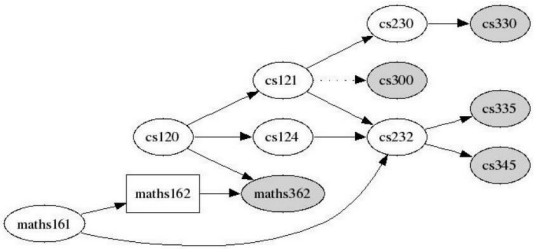

Fig. 2. A visualization for representing classes by prerequisites and corequisites. A similar structure will be used to parse similar data for our system

Almost all other visualization techniques use some sort of graph structure for curriculum, but do differ from the systems above. Some use a graph to link courses with topics. For example, courses appear on the left with related topics appearing in the center, with lines representing a relationship between them. The right side has courses related to those topics, and when selected, topics and related courses light up (Siirtola et al. 2013). This same system also describes a way to see this distance using a matrix, listing all classes on both rows and columns and using grayscale to signify how strongly those courses are linked. Other systems use advanced rendering. One displays a breakdown of all courses in each department in a 3D rendered world that looks like a 2D array with classes often given heights and colors to give more information to the data like number of units. A user can sift through this world to see the 3D rendered grid of course information (Sommaruga & Catenazzi 2007). While these

ideas have their own merits and advantages to displaying class information, they simply display too much information for the interactive visualization we are creating that will focus more on seeing degree progression rather than giving extensive information on each individual class. Because of this, these systems appear too cluttered since we must also show information on overall degree progression.

Other works show different visualizations to show curricula, including hierchical graphs (like in the first system we described), TreeMaps, and nested graphs, making the point that each project working with curriculum visualization might take parts from these three and that none of the three are correct in every situation (Kriglstein 2008). Class curriculum visualization is a very important feature of the degree progression system since it gives solutions on how to fill in the requirements that need to be met. However, visualizing the requirements that need to be met is also important yet not researched a lot. One way to show progression on a high level is using color gradient (Lee & Pea-Mora 2006). We will build off of these findings and see if color gradient is an effective way to show progression in terms of degree progress.

Finally, progression can be broken down into three important qualities for a good visualization: cumulative progress, pace of progression, and if that pace is good or bad to reach the final milestone (Harand 2014). These three qualities seem like good starting points, and we hope to extend what they found here to degree progression specifically and to see how these qualities can be morphed to fit our system.

Ultimately, we found lots of building blocks for visualizing degree progression and relevant and personalized classes. We hope to bring together all the this information to create an original visualization to be a solution to helping students graduate with little frustration thanks to how we represent degree requirement and course data, especially with degree progression.

3 METHODS

In this section, we will talk about the system we created and also the testing methods we used to analyze how effective our hypotheses were regarding our visualization choices for degree progress.

3.1 System

Here we describe the system that we tested users on. A picture of the system as a whole can be seen in Figure 3. First off, take note that these four features can be broken up into two groups: the first three subsections (3.1.1, 3.1.2, 3.1.3) are features we visualized in our system while the last one (3.1.4) is a feature that we created methods, algorithms, and data parsing techniques for so that it can be visualized later. We will present these findings in their corresponding subsection. In each subsection, we will discuss how we visualized the data, why we did it, and our hypotheses of why it will be effective (the results of this test is later in the paper, of course). Look at Figure 3 to see the system as a whole.

It is important to note that our system is generalizable. By that, we mean that our system can accept many different variations of schedules and produce a visualization just for them. It is not a system that has only one set of static data and only applies to that one set.

3.1.1 Cumulative Unit Count and Pace

The first feature we created for our system was a line graph that shows students their cumulative units they have achieved towards graduation. Students need 180 units to graduate, so that is set as a goal line for the user to see. The line graph can be split up into three areas: GER units, major units, and other units. The user can hover over these to see their GPA for a specific area, and in the future, more in depth information about each area. Figure 4 shows this in more depth.

Users can toggle the areas they want shown, allowing a user to filter out any section of their units they do not want to see. In this way, a user can break down unit counts and fully understand the situation he or she is in.

Fig. 3. The system as a whole. The top left shows the first feature we will describe: the cumulative units and pace line. The bottom left shows the percent of remaining units that have to be taken in the major. The bottom left also has the credit/no credit bar. The right column shows the number of units needed to fulfill your major requirements. When clicked, it gives you suggestions for classes to take based on a prerequisite algorithm

An important and novel feature of this graph is the pace or projection line. An example of the projection line can be seen in Figure 5, which shows different states that the projetion line can be in. The line itself is a projection of what quarter the user will reach the goal line of the number of units needed to graduate based on the pace he or she is currently going at. In this, way, we hope to either let the user know that he or she is going at a good pace and should not worry, or can potentially let a user know that he or she should take action and start taking more units in certain areas before it is too late to fix.

An important aspect of the pace line is the color it is. As seen in Figure 5, the color of the line tells a lot about what the user should do in the short run and long run. The color indicates how good or bad the pace projection is. For example, a red line indicates that the person is not going to graduate on time if the same pace is kept. Saturation is used for urgency. If a student takes twelve units in his or her first quarter, the projection line will be below the goal line. However, this is far from an urgent concern, so the projection line is close to black. If it is closer to senior year, the line will be bright to indicate the urgency of the pace.

3.1.2 Credit/No Credit Bar

This bar indicates how many CR/NC (credit/no credit) units he or she has completed. A caveat that Stanford has is that certain CR/NC units, called Activity Units, also have a maximum limit, but they add to the overall CR/NC units.

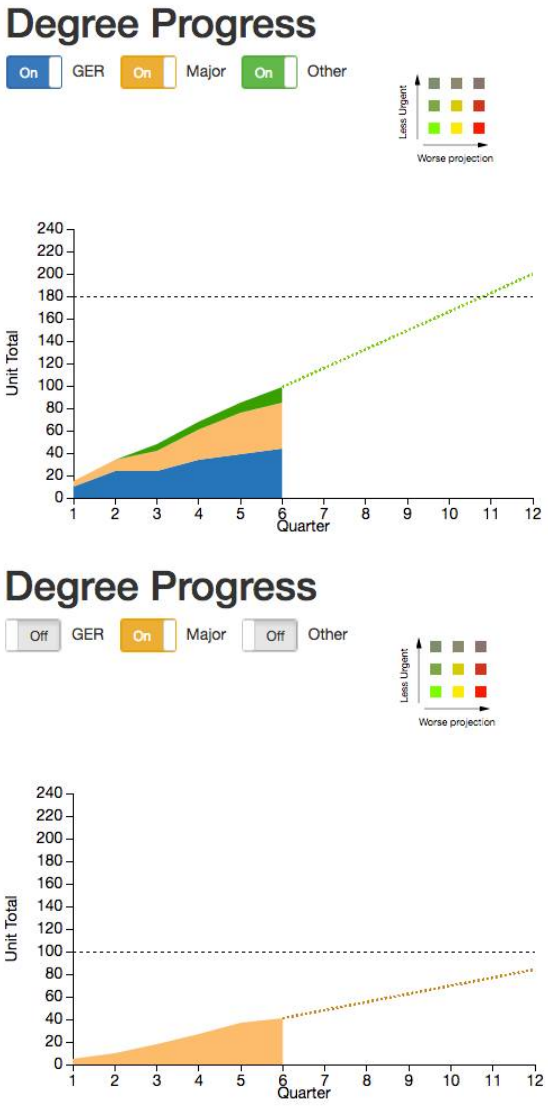

Fig. 4. The two screenshots from our system for the cumulative line graph. The top shot shows a person who is taking classes overall at a good pace, as shown by the fact that the line is reaching the goal line before the end of the twelve quarters at Stanford. The bottom screenshot is when the user filters to only see the major. In this case, this person will not reach the goal line by the end of the twelve quarters, indicating that this person should change the pace at which to take major classes

Fig. 5. The legend for telling pace urgency and projection. Green indicates a great pace with the prospect of finishing before the twelve months, yellow indicates a good pace with graduating right on time, and red indicates a bad pace the indicates a student should start taking more units per quarter. Saturation is used for urgency.

Fig. 6. The credit/no credit bar. It shows the number of activity units in blue and overall credit units in orange, and how much a students has filled up of the 36 allowable credit units and eight allowable activity units

To visualize this, we created a condensed and stacked bar graph. Activity units are added first, then the rest our added. Look at Figure 6 for a screenshot of this. Our main goal was to compact the caveat of Activity units with overall CR/NC units with this visualization.

3.1.3 *In-Major Units*

This shows the percentage of remaining units that need to be taken within the major. A high percentage shows that a lot of the remaining units needed to graduate have to come from within the major. We visualize this by showing a circular progression bar, using color and line thickness to indicate good or bad pace (similar to the first feature we talked about). Red indicates that future quarters will have a lot of major classes, while yellow indicates a lot of out of major classes. Figure 7 shows this.

3.1.4 *Unit Fulfillment and Suggestion System*

The remaining units table which displays the remaining units within each subject of the Computer Science Unspecialized major given a set of taken classes is generated by traversing the majors class requirement tree and matching already taken classes to classes that appear in each requirement. Additionally, when a row of

Fig. 7. Percentage of remaining units that need to be taken within the major. The top screenshot shows a student who has to take a low percentage, and the bottom screenshot shows someone who needs a high percentage

the remaining units table is clicked, the table responds by displaying up to 3 suggested classes that count towards the rows subject, along with both the filled and unfulfilled prerequisites for that class. It is a very simple and clean display of the information that differs from the rest of the product in that it is all text. The majority of the work in building the feature was parsing prerequisite data, and recursively building a prerequisite chain for each suggested class, and finally comparing the numbers of unfulfilled prerequisites for each class before finally making optimal suggestions.

The data for degree requirements is also in a format that can fit the needs of many other universities outside of Stanford. It uses objects that continually build within each other. These objects have properties like ”and” or ”or,” signifying that the set of classes in that object have to all be taken (all) or that one can choose from that the set of classes (or). Cascading these can get many complicated situations into readable data that a system then can sift read through recursively, just like our system does so that we can suggest courses to the user.

3.2 User Test

We will now use our system to user test and get feedback on how our system fares against the current way of accessing graduation data. In this section, we will briefly describe what our user testing entails.

3.2.1 Goals

The goals are to find if our overall system is a step in the right direction in terms of visualizing degree progress. We also want to measure how accurate our hypotheses are with regards to how effective using color, saturation, projection, course suggestion, and specific data we present is to actual Stanford students.

While some of our hypotheses may be correct and some may be wrong, we hope this serves as a foundation for future, more expansive research, specifically for the Carta team (referenced in Section 2) who are eager to see our results.

3.2.2 Procedure

To user test our prototype, we asked users to perform the simple task of understanding where they stood with regards to graduation. Participants were five Stanford students who were either sophomores, juniors, or seniors. Three were men and two were women. Also, two were engineers that were already acquainted with the Engineering Handbook (which is used in the procedure) and three were not engineers.

We showed the participants two systems: the one that students have to use now, and the one we created. We asked them five questions, once they got familiar with the systems, that attempted to test the understanding of a participant’s degree and graduation progression. We used Nielsen’s ten general principles for interactive design to model our questions. We gave them fake degree progression data for a student that finished six quarters at Stanford and was in the Unspecialized track in Computer Science.

First we gave them the fake student’s transcript, told them to imagine that it was them

at the end of sophomore year, and told them to look at that and the Engineering Handbook for 2013 to determine how well they were on track to graduate specifically in the major. This simulated the current system we have now to determine graduation requirements in the major. We also timed the event. After participants said they were comfortable answering questions, we asked them questions about how they were doing with major progression.

Next, we did the same thing, except we told them that the course list they took was different (even though it was the same), but that they were still a student that had finished six quarters and was in the Unspecialized track in Computer Science. Since our system automates these major requirements, the participant would gain no benefit from seeing this first in the Engineering Handbook. The same questions were asked. This was timed.

Finally, we asked what they liked and disliked about each system.

4 Results

The results of our user tests help solidify most of our hypotheses we had when visualizing this data, and a lot of excellent feedback was given on each feature, although some features worked better than others.

Overall, our system did better at helping users understand where they stood at the end of six quarters than the system currently in place, both in time and in understanding the bigger picture of what needs to be done for graduation.

The average time it took for the participants to get comfortable enough to make decisions on how they are doing with regards to graduation progression with the system currently in place was about eight and a half minutes, while the amount of time for our system for this same task was about three and a half minutes. Engineers who did not know of the Unspecialized Computer Science major but were comfortable and familiar with the handbook took about six minutes, while those who were not familiar took just above ten minutes.

Likewise, all of the participants got a more accurate understanding of degree progression with our system. The line graph was effective at helping participants feel more confident that they needed below fifteen units per quarter to graduate but that those quarters would be packed with engineering classes. Also, participants got the same conclusion about the CR/NC situation, except it took approximately one and a half less minutes on average to do so. Furthermore, the percentage of classes needed to be taken within the major was extremely popular and was a stat that every participant cited when discussing how confident they felt when understanding the situation they were in. Finally, the suggestion system raised the participant's confidence in deciding what classes to take. This was especially true for non-engineers, who had never heard of most of these classes and needed reassurance that they were taking classes that needed to be learned early on for other classes in the future.

The other system did not do poorly among engineers, as they all said that they were extremely comfortable with it. It is telling that even after being comfortable with this handbook for years, our system, which nobody had seen before, still produced slightly more accurate and faster results for the same schedule. However, non-engineers got a huge boost in accuracy and time efficiency.

Figure 8 shows the results of asking what people liked and disliked about each system. Note that we also asked which system they would prefer, and they all responded that, if they had to choose one to look at on a regular basis and one to depend on to guide them through Stanford, it would be our system. However, many noted the value in having a resource like the Engineering handbook available anyway.

5 Discussion

The results we got from the user testing were very helpful in understanding the strengths and flaws of our new system. First we will discuss the strengths, and then we will discuss

	What Was Liked About It	What Was Disliked About It
Handbook and Transcript	- Had a lot of detail - Once it's learned well, it's easier to use - Easily see all requirements that needed to be done	- Took more time than desired to read it - Information is cluttered - Intimidating for novices - Not personalized - Difficult to navigate - Lots of skipping around between different sources and pages -"Annoying" was a term used by four of the five participants
Our System	- Units are very easy to track - Projection line was helpful and intuitive - Suggestion system, even without visualization, was helpful - Easy visualization for various components of graduation (filtering system) - Colors were not only helpful, but also made it cheerful and fun to explore - Gamification makes it more desirable to come back and check often - Individual GPAs for each region helpful for in-major GPA (for resume) - Animations when loaded add to progression and make it more fun than a hassle - Requirements for major and units left immediately and easily visible	- Doesn't display courses taken already - Not yet fully polished - A little guidance needed to fully appreciate - Intercept point of pace and goal line should be more visible

Fig. 8. What people liked and disliked about each system after taking this test. Systems are in the first column.

what can be improved for future iterations of a similar project.

Our project was well received among all the participants, and a lot had to do with the information that we presented. The three most popular aspects of our implementation that all the participants commented on were the projection line, the percent of remaining units needed to be taken within the major, and the requirement table with course suggestion. First off, while the line graph was generally well-received overall, the projection line is what many people gravitated towards in helping them understand their current situation. Having a projection line for not only overall units but also just major units (which we took the expected value of since their is small variability in this) was helpful in analyzing the whole picture. In this case, it was understanding that the user could take fewer units but have more engineering-dense quarters, a conclusion that was not reached by three of the five participants with the old system.

The second popular aspect was the percent of remaining units needed to be taken within the major. Presenting this piece of data helped greatly boost the confidence in degree progression, and is a feature that we believe, based on these results, should be presented in future degree progression systems.

The third popular aspect was the course suggestion system. Even after the participants had competed the tasks we had given them, they still played around with this feature simply out of curiosity of what classes they could take given the foundation they had set themselves up with. This feature was played around with the most without us prompting them to do so.

If anything is to be taken away from this study, it is that projection lines, percent of remaining units that are in-major, and course suggestion based on given classes are all three aspects that gave people the tools to accurately and efficiently understand their degree progress situation much better than the old system.

Gamification of progression data was popular as well. Many said that they found it fun to see all the progress they made, even if the progress was fake data we concocted for this user test. The participants also explored the user interface more intensely and more cheerfully, instead of looking for something specific in the other system and immediately quitting. We hope that this would pushes students to more regularly check graduation progression so that they can avoid any potential barriers or obstacles they might run into if they did not have this tool.

As for the other aspects, they worked well but ultimately had their flaws. The CR/NC bar

was useful but we did not see a significant accuracy or efficiency boost, at least not enough to justify just displaying the numbers like what is done on Axess and on one's transcript.

Color was helpful for parsing nominal data, but participants got slightly overwhelmed by the amount of colors that represented different things on the same page. Likewise, colors used for not nominal data (in our system specifically) ended up being superfluous information. For example, while participants understood the saturation and color legend for the pace line, they never needed to reference it since all of them understood that they should not be anxious if they are under the goal line when they are, say, freshmen. The projection line with just red, yellow, or green with no saturation change, or a projection line with no color at all, would have sufficed. Another solution might be to play with line thickness and opacity as it gets closer to graduation.

An interesting note to finish off this section: the people who benefited most from our system were people that were not familiar with the Engineering Handbook. This means that this system benefits people who do not understand the degree requirement process the most, and can help them catch up so that their understanding can be the same as others who may be fortunate enough to already, say, have mentors, parents, or advisers that can personally lead them on. While mentors and advisers are irreplaceable, we hope this system can help those (like first generation students as discussed in Section 1) not be intimidated by Stanford and graduation requirements, and instead feel invited, encouraged, and eager to learn from Stanford.

6 Future Work

Ideas for extensions and future work build on the feedback and suggestions from our peers and also from related work that we found while researching. One of the most impactful additions to our project would be extending our projections and suggestion system to account for students who want to co-term, pursue dual degrees and/or minors, or students completing their undergraduate degree in more than twelve quarters. Such an extension would involve creating multiple milestones in our graph, and possibly expanding the list of subjects on the remaining units table to include classes that count towards the masters degree.

Additionally, our graph does not account for students taking summer school classes. A future version of this project would allow for four quarter cycles instead of three. A very powerful extension to this tool would be allowing students to upload schedule plans, or just a set of classes that they want to take, and build a real schedule based on when courses are offered that accommodates these plans. This would let the see how their workload would distribute over twelve quarters and let them optimize for things like studying abroad or making extracurricular plans.

Building upon this idea of scheduling help is the ability to view the schedules and enrollment patterns of other students. If this data could be collected, it could be used to help freshman familiarize themselves with course sequences within a major they are considering. Projecting GPA given a students academic history could be another interesting extension (although it should be done with extreme care or be given a strong disclaimer). Also, while we used a heuristic for class difficulty (percentage of in-major courses), using data to predict course difficulty and make more accurate nonlinear progress projections would be a useful extension that would make our projection line more meaningful.

Finally, for students who have not declared their major, providing simple visual comparisons for course loads and scheduling differences between majors may help those students make more informed decisions when they do decide to declare.

7 Conclusion

In this paper, we have discussed a system that we hope helps Stanford students with degree progression. We listed four specific features: cumulative units counts and projection lines, percent of remaining units that need to be taken within the major, a CR/NC bar, and a

course suggestion and requirement fulfillment system. We hoped that our hypotheses about using color and saturation for progression, a line graph to help students visualize pace, and using a graph structure in code for prerequisites and suggested courses would help students effectively and efficiently understand the bigger picture of their current situation with degree progression.

After user testing, we concluded that three aspects of our visualization did a particularly excellent job of helping students understand degree progression: the projection line, the percent of remaining units to be taken in the major, and the course suggestion system. Color is good to use as a nominal value, but mixing it with ordinal data can be confusing. Using saturation was intuitive, but in our case, unnecessary. Gamification of a system like this was well-received, as it made people feel less intimidated and encouraged users to explore more than just the task that was given.

We hope that our findings and our system can be a foundation or an aid for systems trying to solve this same problem. Solving this problem can have a direct and positive impact on students, leaving less time to stress and more time to learn, play, relax, and enjoy the college experience.

REFERENCES

Aldrich, P. R. (2014). The curriculum prerequisite network: a tool for visualizing and analyzing academic curricula. *arXiv preprint arXiv:1408.5340.*

Gestwicki, P. (2008, October). Work in progress-curriculum visualization. In *Frontiers in Education Conference, 2008. FIE 2008. 38th Annual* (pp. T3E-13). IEEE.

Harand, Michael (2014, June 11). *How to Visualize Progress?*. http://www.agiledss.com/en/blog/how-to-visualize-progress

Kahlenberg, Richard. D. (2016, February 24). *How Low-Income Students Are Fitting In at Elite Colleges*. http://www.theatlantic.com/education/archive/2016/02/the-rise-of-first-generation-college-students/470664/

Kriglstein, S. (2008). *Analysis of ontology visualization techniques for modular curricula* (pp. 299-312). Springer Berlin Heidelberg.

Lee, S., & Pea-Mora, F. (2006). Visualization of construction progress monitoring. In *Proc., Joint Int. Conf. on Computing and Decision Making in Civil and Building Engineering* (pp. 2527-2533). Reston, Va.: ASCE. Chicago

Opidee, Ioanna (2015, March). *Supporting first-gen college students*. http://www.universitybusiness.com/article/supporting-first-gen-college-students

Siirtola, H., Raiha, K. J., & Surakka, V. (2013, July). Interactive curriculum visualization. In *Information Visualisation (IV), 2013 17th International Conference* (pp. 108-117). IEEE.

Sommaruga, L., & Catenazzi, N. (2007, April). Curriculum visualization in 3D. In *Proceedings of the twelfth international conference on 3D web technology* (pp. 177-180). ACM.