Visualizing Online Learning to Demonstrate Course Structure and Provide Intelligent Feedback

Joshua Powers
College of Computing
Georgia Institute of Technology
powersj@gatech.edu

ABSTRACT

I propose a visual representation for online course structure in order to drive intelligent feedback. The project considers the need to utilize advanced technical features over traditional non-digital educational methods and the advantages of a visualization in the development of online learning. The effectiveness of this approach is demonstrated by applying the visualizations to real world, live courses offered by edX.

Author Keywords
Education; intelligent feedback; massive open online course; visualization.

ACM Classification Keywords
Human-centered computing → Dendrograms.

INTRODUCTION
The continual blending of education together with technology has led to a trend of taking traditional educational experiences and placing courses online with no changes [1]. Prior to the massive open online course (MOOC) revolution, the general online course experience involved students watching lectures, normally recorded in-classroom lectures, and then taking a multiple choice test. If the student passes he moves on to the next set of lectures, but if student does not pass, he is forced to go back and review lectures with little to no guidance or feedback.

The type of online education described above does not take full advantage of new technologies that may exist or that have been enabled by online platforms like MOOCs [2]. As an example, consider methods for providing intelligent feedback to students during assessments. Taking a traditional multiple choice quiz and only making it accessible online ignores the fact that educators could instead be providing real time feedback by mixing and matching additional assessment methods, or analyzing student inputs to determine where there are places in lectures that may need to be refined. Only recently have advances in online education begun to utilize the technologies available to enrich the online experience.

In addition, writing well thought out assessments is incredibly difficult. A teacher needs to spend a considerable time refining and crafting an assessment to help guide a student's learning. Therefore, enabling teachers to take advantage of the tools and processes available to them in online learning will enable them to be more effective.

Here I present a visualization to enable both teachers and students to view course structure and features. The goal of this is to assist educators in understanding the methods they are using, or not using, to teach and assess students and to demonstrate to students the course layout and set expectations on how the course is structured.

VISUALIZTION APPROACH

This section describes the approach in terms of four areas: completeness of a visualization, automation of building the visualizations, interaction with the visualizations, and finally the direct feedback to students and instructors.

Completeness
The intent behind the visualizations is to be able to highlight a course's structure in one view in an effort to showcase the entire course content and provide direct details and feedback about particular parts of the course. Many online MOOCs already provide a course outline and even traditional education will provide some outline showing what a course will involve, however these views are only of the high-level parts and particular sections.

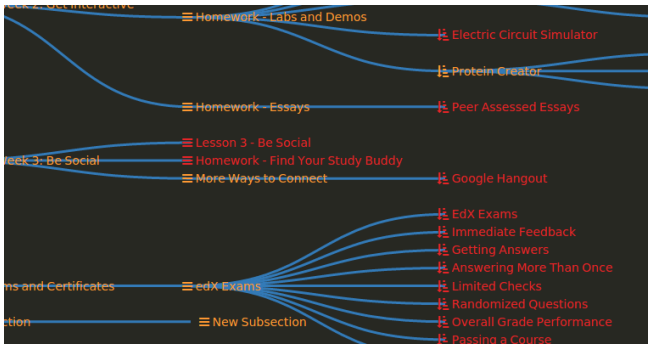

Figure 1. Shows a snapshot of the generated visualization from an example edX course.

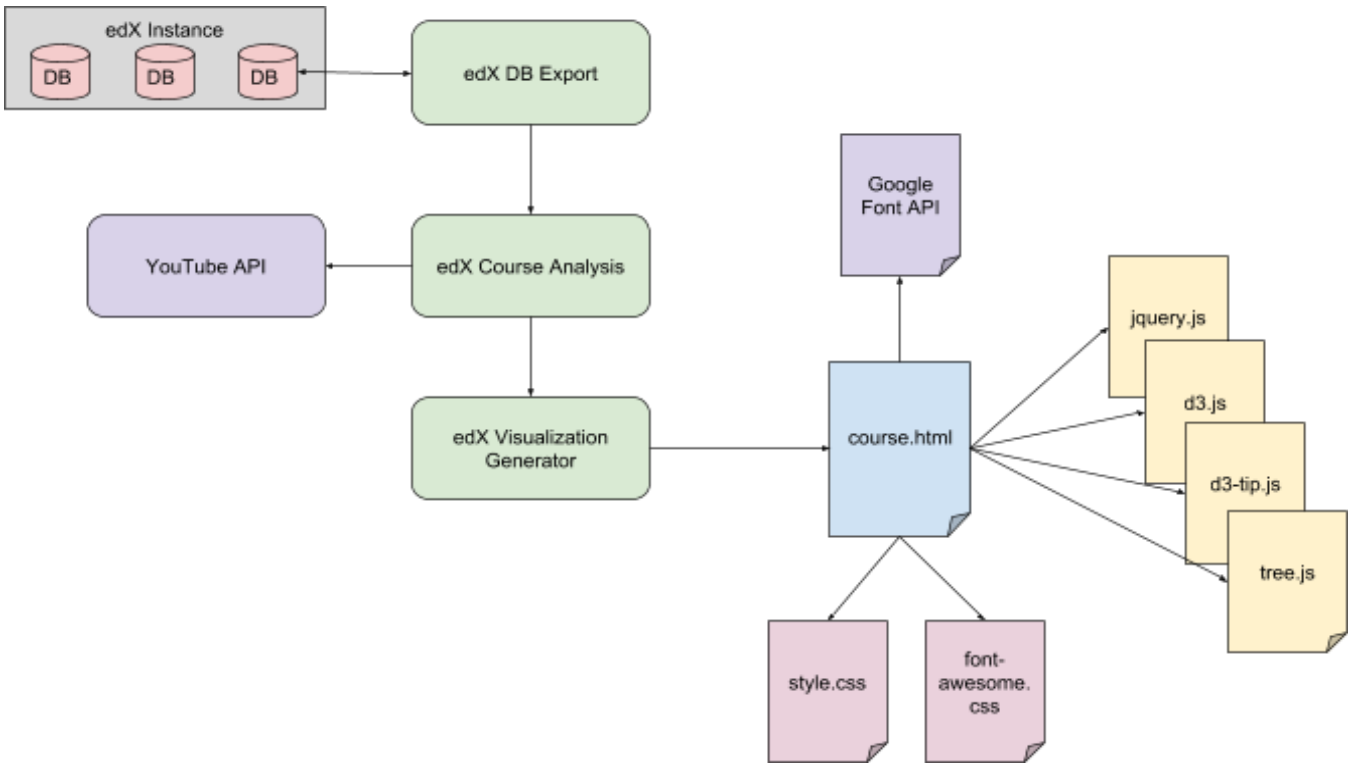

Figure 2. Shows an overview of the complete, end-to-end system used to generate the visualizations directly from an edX instance.

Automation

Automation was a key requirement for the visualizations in order to make them easily generated, reused, and leveraged across more than a single course. I had originally planned a complete end-to-end solution that would have integrated the visualizations directly into the edX platform. This allowed a course designer to set up a course and then view the visualization after making changes to see how the course was laid out. This final integration work however did not get completed and the visualizations are left as a standalone product.

What did get completed was the following: extraction from the edX database, parsing of the data, analyzing of the course data, and the generation of the visualization into a standalone HTML page.

First, the tool connects to the edX MongoDB instance where all unstructured course data is stored and the data is exported from the database. Once the raw data is in hand each course content is read and specific data is kept. Certain data that was not immediately needed for the visualizations was thrown out in an effort to keep the analysis working quickly so as to limit the scope of the visualizations. The final data is then produced into JSON format for easy reading at the next step.

Next, the course is then analyzed for specific features. This involves building the child-parent relationship for all the course components and to eventually build the visualization tree. This step includes the generation of intelligent feedback. As an example, I analyzed the problem set text to see what intelligent feedback features were considered and determined the length of lecture videos.

Finally, the data is then put into a JavaScript data structure and copied into a uniquely generated HTML page. The page has already been formatted with the necessary JavaScript libraries and cascading style sheets that import the necessary styles that produce the look and feel of the visualizations.

Interaction

Interaction with the visualizations was essential in giving users a way to manipulate and see the course content. The visualizations are able to be zoomed in and out on as well as moved around the screen. This was tested on both a desktop system as well as a mobile device. The experience here was fairly intuitive and fluid allowing individuals using the visualizations the ability to drill down into the sections that they were particularly interested in.

Feedback

While considerable time was spent on the visualization design and implementation one of the primary goals was to provide feedback to the teachers. In order to restrict the scope the focus for feedback was placed in two main areas: problems sets and videos.

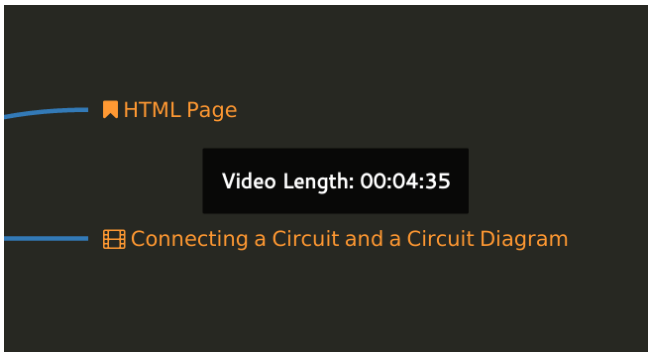

Figure 3. Example of video length tooltip.

For videos the overall video length is directly placed on the page as a tooltip for each video node. With many MOOCs video time can vary radically and video duration could mean a 30 second video or a much lengthier 10 minute video. Having the video length easily viewable can help teachers understand total length of a section and possibly encourage students to plan for the amount of time to devote to a particular section.

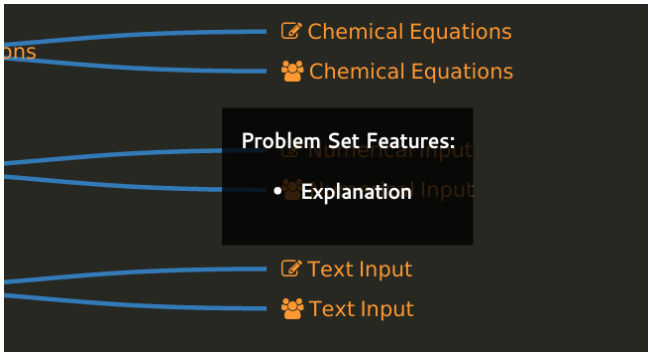

Figure 4. Example of problem set tooltip.

With respect to problem sets, the edX platform has a few methods of intelligent feedback: *feedback*, *explanation*, *incorrect answer hint*, *general hint*. The tool analyzes the problem set markdown and determines which, if any, intelligent feedback was used. This data is then used to produce a tooltip indicating the usage.

DESIGN CONSIDERATIONS

In this next section I describe various design considerations involved with the production of the visualizations: the choice of edX, the overall shape and layout of the visualization, the theme, the various icons used throughout the visualization, and finally YouTube integration.

edX

I choose to utilize the edX platform [3] as the MOOC platform to pull data from and generate visualizations based on that data. This choice was made due to the fact that it is completely open source and I was able to setup my own instance so as to learn how to extract data directly from the platform. Setting up a personal instance of the edX platform was made incredibly easy by readily available images and developer guides that could be deployed and updated in a matter of minutes. I was able to setup the edX platform on my laptop in less than hour.

There were numerous edX example courses available for download on GitHub that were utilized for the purpose of testing and generating these visualizations. The three primary courses I used for development test cases are: edX Demonstration Course [4], edX Test Course [5], and the edX Multivariable Calculus [6]. The last of these is an actual course offered on the edX platform. The first two courses are offered up as development platform test cases that make use of large portions of the edX platform. These courses made for great use cases during development.

Data-Driven Documents

Because the end goal was to produce a web-based visualization, numerous JavaScript libraries were considered for the production of the visualizations. The eventual choice however was the use of the Data-Driven Documents, or d3, JavaScript library [7]. d3 is a heavily maintained and utilized JavaScript library with hundreds of examples [8]. Many other libraries that were considered were more focused on producing plots, graphs, or charts and only very few had specific abilities to produce maps or trees to highlight course hierarchy.

Due to the fact that a course has specific parent-child relationships a tree diagram made the most sense in representing the course structure. The d3 library has the ability to produce various types of trees, but the ultimate selection was on a dendrogram [9], which comes from the Greek “tree” and “drawing”. This type of diagram shows hierarchical clustering, like that of a course structure very well [10].

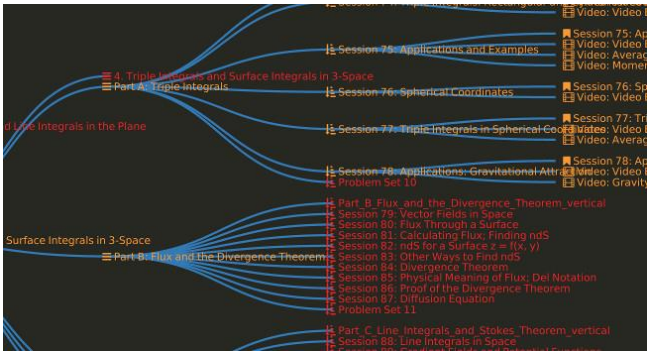

Figure 5. Snapshot of a horizontal tree layout. This layout was chosen due to its easier readability and more natural reading progression of left to right.

The d3 library itself makes use of the Reingold-Tilford algorithm [11] for the generation of the trees. The original paper for this NP-complete algorithm sets out numerous aesthetics that produce “tidy” drawings of trees. The result is a tree that looks very symmetrical with constantly centered parents over their children. The downside is that some of the results can become quite wide. To remedy this the visualizations are flipped and presented in a horizontal

fashion. As a positive side effect the text flows and the visualization reads more easily left to right.

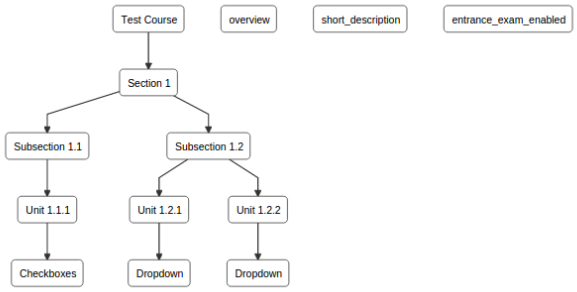

Figure 6. Early prototype using top to bottom layout. This layout grew too tall far too quickly and was rejected in favor of the more readable horizontal layout.

A plug-in for the d3 library called d3-tip [12] was also used to produce the tooltips seen by the feedback elements. A slight modification to the library had to be made due to the assumption the library makes that all nodes will have tooltips. Given that feedback is not produced or provided to all nodes the modification will have the library return no tooltip when nothing is passed in to be shown in the first place.

Theme

With anything visual the appealing nature that allows the viewer to discern the elements and comprehend the layout is essential. I preferred a darker theme over a more traditional white background due to the fact that it is much easier on the viewer's eyes. The colors chosen were meant to be in contrast with the near black background and work together with one another.

Figure 7. Color theme used throughout the visualizations.

The red is the primary color for nodes that have children that are not expanded out. Once the user clicks on a node it will expand and the parent will turn orange thus indicating that the node is fully expanded. All nodes in the tree are connected by edges in blue.

Color blindness of the viewer and general accessibility in design has become an important consideration as online content becomes more broadly relied on [13]. The edX developer's guide [14] even has a large section on accessibility in an effort to reach and educate as many people as possible. I did put the color scheme through a color blindness generator [15] to simulate the effects of dichromacy.

Figure 8. The color theme as seen by someone with deuteranope (top), protanope (middle), and tritanopia (bottom) color blindness.

The figure above simulated the three main types of dichromacy. While the red and orange colors can appear similar they are still seen as two distinct colors. I would have also liked to see how grayscale looked.

Icons

Rather than having a generic bullet point for each node by default I choose to have a unique icon for each edX node type. This was done in order to show the differences in nodes and make them easily identifiable. I utilized the AwesomeFont [16] scalable vector icon library to assign icons to each of the nodes.

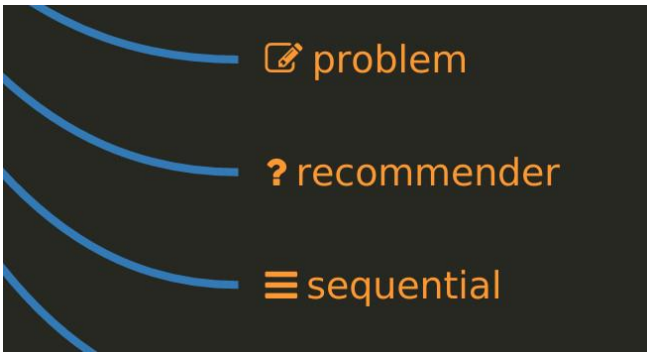

Figure 9. This figure was generated to showcase a subset of the various icons and types available, specifically with an edX course.

This allows for a more easily scanning by the user, student or teacher, to see what content is where. For example, this can highlight when a section or grouping of videos may occur or where lots of problem sets will be coming up together.

YouTube Integration

Finally, to return the YouTube video lengths I leveraged code to query the YouTube API [17] and query for the video lengths. I set up my own API key and would take the given video id from the edX course data and query YouTube for the length. The lengths were returned in an ISO 8601 string and converted to a more traditional standard time format of 'HH:MM:SS'. This data was then returned as the tooltip data.

CONCLUSION

In this paper I presented a novel approach to visualization for online course content. The approach consists of four main features: *completeness*, *interaction*, *feedback*, and *automation*. Each of these items alone are existing techniques, but together have not been widely applied to online course content for the purpose of visualizations. This approach marks one method for technology to be applied to course structure and content in order to better online learning and education. The crucial point in this approach is to provide intelligent feedback to students and teachers.

Future Work

There are several directions for future work:

First, the most immediate and apparent issue is the overlapping of very long node names. This was very apparent with the edX Multivariable Calculus example course, which contained very lengthy names. This causes the node text to run over into the next section of nodes creating a visually unappealing overlap. I propose two possible solutions: First, the text could be cut so it only contains a specific number of characters, anything after that is left off, and replaced with '...' indicating additional text. This approach unfortunately may make some node text confusing when the important text is towards the end of a long string. It is however, the easier of the two options to implement. The second option would have the text add a new line after a certain number of characters. This would split the text across more than one line in order to prevent the overlap. I was unable however to determine if the tree node would be able to handle this additional line of a node text. After looking through many D3 examples I was unable to find a scenario where this was the case.

Next, the current approach only applies directly to the edX platform. An interesting extension would involve allowing the inclusion of data from other online platforms (e.g. Udacity, Coursera, etc.) and produce visualizations that may be tailored for them.

Next, on a similar note the visualization generation could be fully integrated directly into an existing MOOC. I had desired to get this working entirely with the edX platform, but due to time and lack of experience, failed at completing that in a timely manner. This work could still be done with edX or some other platform.

Finally, the solution to the question what would provide the most value to the tool is the extending of the tooltips to additional edX types. I was only able to complete the analysis of the edX problem set and video types. This same concept could be extended to many other types giving students and teachers more feedback and information. For example, section information could be included in the high-level types or links to the actual page content for the course.

ACKNOWLEDGMENTS

I wish to thank Dr. David A. Joyner for his mentoring and guidance throughout this project, Dr. Walter and Marjie Powers, Olga Martyusheva, Alex Balderson, and Aaron Anderson for their feedback, evaluation, and support, and finally his classmates of CS 6460 Educational Technology for their peer feedback and insightful conversations throughout the semester.

REFERENCES

1. Office of Educational Technology. Ed Tech Developer's Guide. 2015. Retrieved from http://tech.ed.gov/developers-guide/
2. Robles, M., & Braathen, S. Online Assessment Techniques. *The Delta Pi Epsilon Journal 44*, 1 (2002), 39-49.
3. Open.edx.org. About Open edX | Open edX | Open Courseware Development Platform. 2015. https://open.edx.org/about-open-edx.
4. GitHub. edx/edx-demo-course. 2015. https://github.com/edx/edx-demo-course.
5. GitHub. edx/demo-test-course. 2015. https://github.com/edx/demo-test-course.
6. GitHub. mitocw/18-02sc-fall-2010.edx. 2013. https://github.com/mitocw/18-02sc-fall-2010.edx.
7. Bostock, M. D3.js - Data-Driven Documents. *D3js.org*, 2015. https://d3js.org/.
8. GitHub. mbostock/d3. 2015. https://github.com/mbostock/d3/wiki/Gallery.
9. Strongriley.github.io. d3.js ~ Dendrogram. 2015. https://strongriley.github.io/d3/ex/cluster.html.
10. Wikipedia. Dendrogram. 2015. https://en.wikipedia.org/wiki/Dendrogram.
11. Reingold, E. and Tilford, J. Tidier Drawings of Trees. *IEEE Transactions on Software Engineering SE-7*, 2 (1981), 223-228.
12. GitHub. Caged/d3-tip. 2015. https://github.com/Caged/d3-tip.
13. Mkweb.bcgsc.ca. Data Visualization, Design and Information Munging // Martin Krzywinski/ Genome Sciences Center. 2015. http://mkweb.bcgsc.ca/colorblind/.
14. Edx.readthedocs.org. edX Developer's Guide" edX Developers Guide documentation. 2015. http://edx.readthedocs.org/projects/edx-developer-guide/en/latest/.
15. Color-blindness.com. Coblis" Color Blindness Simulator | Colblindor. 2015. http://www.color-blindness.com/coblis-color-blindness-simulator/.

16. Gandy, D. Font Awesome, the iconic font and CSS toolkit. *Fortawesome.github.io*, 2015. https://fortawesome.github.io/Font-Awesome/.
17. Google Developers. YouTube | Google Developers. 2015. https://developers.google.com/youtube/.