Wikitree: Wikipedia Mapping Companion

Alexander G. Burner
Information School
University of Washington
Seattle, WA 98195-2840
aburner@uw.edu

ABSTRACT

Wikitree is a web application built to help undergraduates with their research. Wikitree tracks and organizes the user's browsing history through Wikipedia articles. Wikitree is unique because it emphasizes mental connections the user makes. Wikitree aims to stay adaptable for each individual's research goals. Our research and user testing revealed many struggles undergraduates have. They struggle to dive deep into one topic while staying aware of the big picture. They struggle to mentally categorize connections between different topics during early stages of research. They struggle with feeling overwhelmed by the amount of material encountered, and finding a clear vision within. Wikitree acts as an aid for the user in tracking their explorations. The visualization helps users maintain a high level overview, preventing them from feeling overwhelmed, getting sidetracked, or losing sight of their goals.

Author Keywords

Wikipedia, network, force-directed layout

ACM Classification Keywords

H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION

Research can often result in sprawling journeys with cyclical and branching explorations involving backtracking and falling down rabbit holes. At the end of these journeys a user's browser window may be crammed with open tabs, each with their own long navigation history. Both tabs and history are serial, linear paradigms, and fail to represent the full shape of ground covered during research.

Wikitree aims to more effectively map these journeys.

CHI 2009, April 4–9, 2009, Boston, Massachusetts, USA.

Motivation

Research is rarely limited to a single source. Whether an undergraduate student reading a Wikipedia article or a scholar reading a research paper, people in the process of learning usually need to read related works in order to gain a larger understanding of the concepts they're studying.

We believe research journeys can be accurately mapped with directional networks. We want researchers to be able to stay in their flow state of digging and exploring, while also holding a big-picture overview of their progress. In this way, people can dive deep while simultaneously maintaining perspective.

Why Wikipedia?

Wikipedia is a modern exemplar of our species' increasing democratization of knowledge. Historically, knowledge has been limited to those with the privilege and resources for academic training. Even within the last century, encyclopedia collections have been prohibitively expensive. Public libraries help ease this barrier by offering free access to reference materials. However, only with Wikipedia is the largest, most cutting-edge encyclopedia now available for free to anyone with a basic computer and Internet connection (both increasingly ubiquitous resources).

While navigating Wikipedia, each article has many keywords hyperlinked to other relevant Wikipedia articles. This allows a single article to be supplemented by the content of similar articles, so a user can begin on one topic and move outwards, digesting whatever contextual information is necessary to reach personal comprehension.

However, this branching navigation is difficult to capture with browsers. Modern web browsers have windows, tabs, and navigation history which can all hold separate articles in their own way. However, these tools are relatively flat (in the case of windows and tabs) or linear (in the case of navigation history). A user can save a few key articles, but the full breadth and depth of their exploratory learning may be lost.

This project aims to create a map of the user's navigation through Wikipedia articles, while preserving the native Wikipedia experience. It will create a visualization which records the user's journey as well as providing an overview of the concepts they've explored. The primary goal is to facilitate comprehension and retention. The secondary goal is to increase engagement, helping learning feel like the adventure it truly is.

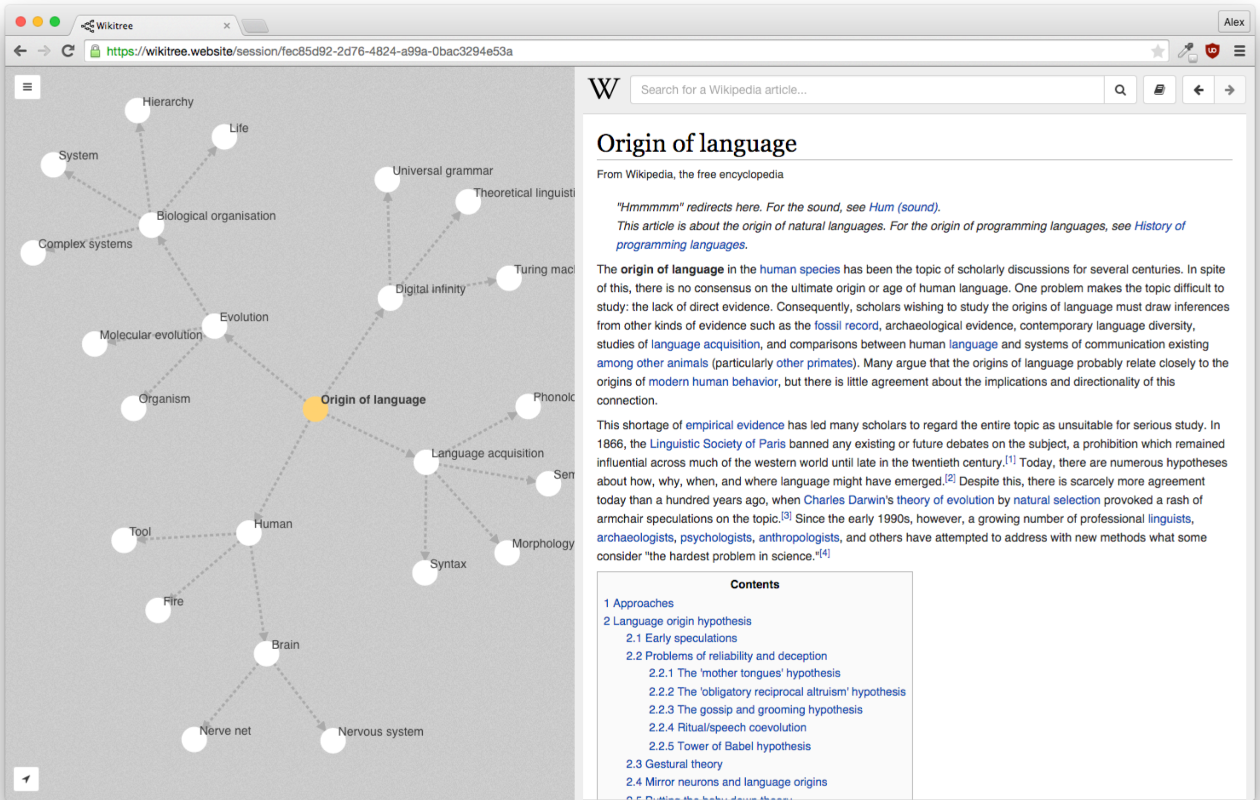

Figure 1. The Wikitree user interface running in a web browser. The network graph article node map takes up the left portion of the interface, while the document article reader takes up the right. Each node represents a Wikipedia article, and the directed edges between them represent inter-article links clicked by the user. Arrows on each edge indicate source and target of link clicks.

Approach

We display the research map and article reader side-by-side, so the user can access both simultaneously. The research map updates as the user clicks through links in the current article. The map serves as navigation: clicking a node loads the represented article into the reader. The map can also be curated, with popover controls for pinning nodes, removing nodes, and breaking links. Sessions are saved and displayed in a pop-out sidebar, so the user can pause and resume multiple projects.

We use the D3.js force-directed graph layout to render the nodes and links. We use Wikipedia's MediaWiki API to fetch title suggestions, article content, category pages, and full search results. We display the Wikipedia article content in an HTML iFrame element and bring in Wikipedia's own stylesheets to give the articles a native look and feel. We use DOM Local Storage to save sessions to the browser. We use AngularJS as our client-side framework and Node.js with the Express framework as our server, hosted on a DigitalOcean droplet.

RELATED WORK

There are many projects with relevance to both assisting research and exploring Wikipedia (although not so much for both simultaneously). We explore a few examples here.

Refinery: Visual Exploration of Large, Heterogeneous Networks through Associative Browsing

Refinery [4] is a tool built to encourage and facilitate "exploratory information-seeking" behavior. It uses "associative browsing" to help users start with familiar information and move into new areas of discovery. Refinery's primary focus is large, heterogeneous networks containing many forms of media (people, papers, presentations, images, videos, etc). Their interface integrates text queries with suggestions, control panels for keyword refinements, and a D3-powered force layout graph for displaying interlinked search result nodes (as well as an alternate "list view" for more fine-grained detail). Refinery's emphasis on suggesting new data and supporting serendipitous discovery is very interesting. We would like to adopt this emphasis on exploration and introduce our own form of article suggestions.

Apolo: Making Sense of Large Network Data by Combining Rich User Interaction and Machine Learning

Apolo [2] is a tool for extracting knowledge from large network datasets. Similar to Refinery, it is focused on discovery of new information. Also similar to Refinery, it uses machine learning to adapt to the user's queries and exploration patterns. However it only leverages homogenous networks (scholarly articles). Apolo allows many user interactions with the graph, such as starring, annotating, pinning, selecting, and hiding nodes, as well as creating node groups. These seem like strong control structures that would give the user a feeling of empowerment over the graph they've generated. We would like to bring similar features into our project.

Infobaleen Wikipedia Map

The Infobaleen [6] Wikipedia Map is a project that attempts to cluster Wikipedia articles by analyzing their network structure using the Infomap network analysis software package [3]. The map allows top-down browsing: starting with large clusters and digging inwards, and also bottom-up browsing: searching for a particular article and diving to its depth, then viewing related articles or "zooming out" to higher levels of clustering. The map is a hybrid of a treemap (nesting clusters) and network graphs (between clusters). It emphasizes the structure of Wikipedia, and doesn't display full article content. Our project will focus more on the user's own generated structure, and allowing perusal of full articles with native Wikipedia markup and styles. However we are interested in the clustering and think this could be a useful feature for helping our users organize their article nodes.

Local Wikipedia Map

Local Wikipedia Map [5] leverages the interconnectedness of Wikipedia articles. It allows the user to choose 2-5 Wikipedia articles and then displays a network graph of shared articles between them. The project uses data from DBPedia and crawls many levels deep to find new articles that are shared by the user's chosen set. The resulting map both reveals the interconnectedness of the chosen articles and offers the user a chance to explore new articles they may find in the generated common pool. We are interested in this tool's ability to traverse multiple degrees away from current/selected articles and form connections using relatively distant articles and link paths. This could be a useful feature for our project's suggestions: finding how a current article is indirectly linked to other articles in a user's current graph.

METHODS

Here we will discuss the methods that went into the creation of Wikitree. We will talk about the goals we held in the design of the project, the technologies we used to create the project, and the features we implemented in the project's construction.

Design Goals

The main goal for Wikitree's design was to keep it simple and intuitive. With most projects, the challenge is not thinking of new features, but selecting from the flood of potential features spilling in. There were many different things we could have done with Wikitree, and so we made a concentrated effort to only add the most essential ideas. We wanted to keep the interface clean, so that a new user was not overwhelmed. We wanted our controls to be intuitive, so a new user could quickly pick up the application and get to work, without a significant learning curve. We were forced to balance our desire for simplicity with the need to give users full control, so a more advanced user would not feel limited or hampered by a lack of interaction.

But, the goal of "simple and intuitive" is true for any piece of software with a user interface. Our core design vision for Wikitree specifically was to make it a research assistant for the user. A successful research assistant suggests new material and organizes existing content. Due to time constraints, we did not push deeply into the area of suggesting new content (see Future Work for more). However, we did make sure the user was unimpeded in fetching new content for themselves. We provide easy access to Wikipedia's title suggestion and full search capabilities (see Technologies and Implementation for more) for bringing in any articles from Wikipedia, and the purpose of Wikitree itself is to encourage inter-article link clicking within articles, and bring in this new content while maintaining easy access to the your starting points. For organization, we rely on a two-dimensional network to display the travels of the user (see Technologies) and give them the ability to rearrange and curate this to their whim (see Implementation).

As part of our vision for Wikitree, we wanted it to be an unhampered portal into the actual research content (in this case, Wikipedia). To those ends, we split the screen, giving ⅗ to the article reader and the remaining ⅖ to the map (giving the content some priority over the graph). We also strived to make the article reader match Wikipedia's native content. (More on this in Technologies and Implementation). We wanted the application to stay out of the user's way, letting them find their research flow state.

We wanted Wikitree to allow the user a big-picture overview of the content they've been exploring. Unlike a simple list, we didn't want to just give them access to the locations they've visited. We wanted to provide structure in the record of their journey. We wanted it to be a useful map that both showed them where they had been and also gave them a high-level perspective of the areas they've been exploring. Research itself is done on the ground, walking through pages. We wanted the user to also have a bird's-eye view of their trails.

Another important design aspect was curation and cleanup. We wanted the user to feel free to explore, without fear of making wrong turns. To this end, we put in place controls to allow users to arrange nodes for themselves, as well as remove any unwanted nodes or links from the graph (see Implementation for more). These controls encourage the user to take an active role in the creation of their map. It is not just a passive record of their journey, but a living document they can engage with and alter as they see fit. That being said, the user cannot add links that don't exist (you must click an actual article link, ensuring that the map only shows true paths from Wikipedia's larger network of interconnections). We wanted to make accidents cheap, so the user could explore fringe articles without them permanently adding to the clutter of the map. We wanted to encourage the user to keep their map clean and useful, to craft it into something they felt was their own.

Technologies

We built Wikitree as a web app. This was an obvious move for our team, both in terms of the tool's context and our own resources. In terms of context, the Wikitree tool is a visualization of Wikipedia, itself a web site. Wikipedia shares their content via the MediaWiki API [7], easily accessible through JavaScript's AJAX techniques. In terms of our team, both developers (James and I) have the bulk of our experience building web applications. The web itself has many advantages (and drawbacks) as an app platform, too numerous to dive into here. As a web app, Wikitree is built on HTML, CSS, JS, and SVG. Our tool essentially exists to glue together the D3 Force-Layout graph [7] and the MediaWiki API. Its server-side is built on Node.js and Express, and its client-side is built on AngularJS and jQuery. The JS DOM localStorage interface is used to save and restore serialized versions of the user's graph "sessions". The Bootstrap library was used for basic buttons and other HTML+CSS components, along with the FontAwesome library for icons.

We use the D3 Force-Layout graph to display a user's journey through Wikipedia articles. Each article visited appears as a node in the graph, and each link followed is a directed edge between two nodes. The Force-Layout allows a relatively even distribution of the nodes. The layout uses an approximation of four physical laws: each node has an electric charge (so they repel), they all share a common center of gravity (so they stay collected around a center point), each link is a spring (to allow for more flexible arrangements when things get crowded) and the overall simulation has air resistance (so the nodes don't drift forever, allowing the graph to come to a graceful halt and save on CPU processing).

The D3 Force-Layout graph uses SVG elements to form the nodes and links. This allowed us to leverage native browser DOM events to track user actions such as mouse hovers and clicks, per each element in the visualization. Using a technology such as canvas would have meant we could only listen for these events on the entire canvas, and then would have needed a scene graph to derive which element was the user's target. D3 trusts the SVG DOM to be its own scene graph. We also added an HTML layer on top of the SVG nodes and links for enhanced user controls (see more about the popovers in Implementation).

D3 also offers a number of helpful utility functions, such as cross-browser event listeners that can work for both mouse and touch events. We also utilized their zoom behavior, so the user can scale and pan the graph. This allows them to zoom in and get details on small areas, or zoom out and get a large overview of the general structure they've put together.

For content, we used Wikimedia's MediaWiki API to access Wikipedia's materials. Four of their endpoints were of particular interest. Central was their Parse endpoint, which accepts an article title as a parameter and spits out the formatted HTML if a match is found. This endpoint only works with exact title matches, so we paired it with the Opensearch endpoint, which returns article title suggestions based on string fragments (so sending it "sugges" would return an array of strings such as "suggestion", each the proper title of a Wikipedia article). This helped users find exact title matches for the words they were seeking. However, users occasionally reach beyond even Wikipedia's massive collection of article titles, and so we also integrated their Search API, which accepts a query and returns results of a full-text search of articles. Fourth and finally we hooked up to their Query API which allowed us to get subcategories and member pages for a given category (as the Parse API didn't return full category pages, only the descriptions).

We displayed Wikipedia content in an iFrame, allowing us to use Wikipedia's full stylesheets without their CSS interfering with our own app's styles. The iFrame contained Lodash templates for mimicking the Wikipedia markup of articles, category pages, and search results. We wanted users to feel they had a complete and unhampered Wikipedia experience, so we worked hard to ensure that Wikipedia's native styles were preserved. The iFrame captures user click events and evaluates their target, deciding whether to extract an article title and return it to the main Wikitree application, or pop open a new tab for unhandled links (such as File pages or external sites).

We used AngularJS as our client-side framework. In Wikitree's current version, the entirety of the application logic occurs client-side, so we needed a robust framework that could organize our code effectively. We found Angular to be helpful in many ways. However, it was also difficult, especially when it came to syncing events and data across otherwise distant application modules. We will be considering other libraries for future versions of Wikitree.

We used Node.js and Express for Wikitree's backend. Currently, Wikitree is entirely a client-side app, so any web file server would have been fine. However, we plan to add support for server-side user accounts and session saving (see Future Work for more). When that time comes, the Node app will be given a larger responsibility.

Implementation

Wikitree has 4 major interface components: the welcome page, the map, the reader, and the saved sessions sidebar. These are divided into two pages, the welcome page on its own and the other three existing as three columns in the main page. The sessions sidebar is designed to mostly exist out of site, and only slides in when called by the user. The map and the reader are both always visible, and start with a 2:3 split of the screen. The user can resize this divide, giving more space to the map or reader as desired.

The welcome page is designed in the form of the Google homepage: simple and to the point. The user is presented with the Wikitree name and a large search bar, inviting them to find a Wikipedia article. In the background, an automated D3 force-layout graph unfolds. It is randomly generated, 200 nodes are slowly added, each popping out linked from a randomly chosen node that's already been inserted (except, of course, for the first node). This animation serves to both entertain new users and demonstrate the capabilities of Wikitree. During user testing (see Results for more) many users only followed the article links in a straight line. The hope of the animation is that it will help to illustrate the branching nature of the tool.

The map is the core of the information visualization. It is the extra functionality we add on top of the normal Wikipedia experience. As explained in Technology, we use the D3 force-layout graph to represent each article as a node and the user's inter-article navigation as directed edges between them. The nodes themselves are very simple, just a white disc with the article title. Initially, we tried represent each article with small tiles, containing title, an image, and the introductory paragraph. This seemed reasonable on paper: we would give the user enough information to simply read the map and learn about its content. However in practice the tiles provided too much clutter, and the overall effect was overwhelming. The simple disc nodes, originally intended as temporary placeholders, are now permanent. The links between the nodes are represented with dashed lines. This was another feature that started as a temporary experiment, but was appreciated by users during testing (see Results for more). The dashed lines gave increased feedback to the "bounciness" of the graph structure, which many users enjoyed. The bounciness itself had to be carefully balanced using D3's force-layout parameters. Too much bounciness was distracting, but too little and the graph felt sluggish.

We added an additional layer of user controls over the force layout graph. The nodes themselves were clickable, serving as links to the articles they represented (opening them in the adjacent reader). They could also be dragged into new positions, which would "pin" them in place (using D3's "fixed" attribute). This allowed users to arrange maps to their own liking. The new layer we added was a set of HTML "popovers" containing buttons allowing for additional node controls. Standard node popovers had buttons for toggling their pin state and removing them from the graph. Link popovers had a button for breaking the link between two articles. Note node popovers had a third button for entering "linking state" (discussed further in the next paragraph). This additional layer of popover controls gave users an increased degree of control over the graph, while keeping controls unobtrusive until the moment they were needed (the popovers being hidden until the relevant node or link element is hovered over).

Our most recent addition to the graph is the "note nodes" which allow annotations of the graph. These are nodes that behave similarly to the standard nodes. However, they are user-created, and can have both titles and body text, allowing for more information to be displayed in the graph. Instead of a circle, they are represented by an icon resembling a piece of paper. In addition to adding text, the note nodes can be linked at will to other nodes (both standard and note). The links are colored blue instead of grey, for differentiation. The note nodes are linked via their third popover button, which toggles a "linking" state in which a link attaches to the user's mouse cursor (actually an invisible node following the mouse). During this state, any node the user clicks will be linked to the source note node. The user can click any number of nodes, and ends the linking state by clicking anything that is not a node (or by pressing the escape key on their keyboard). This linking allows the user to "wrangle" together nodes, forming new shapes in the graph layout. For example, they could pull together all nodes that represent capitol cities, or geographical features, or musical techniques. In this way, the notes can behave as a form of local categorization. The note nodes become both textual annotations, as well as spatial.

The next interface component is the reader. As discussed in Technology, we took care to make the reader accurately simulate the native Wikipedia experience. The reader has an iFrame that displays the MediaWiki API results, as well as search bar for "teleporting" in new Wikipedia content. Anything from the reader search bar is brought in disconnected from other nodes (though the user can connect them all together if they're able to follow corresponding links). If the user enters a direct article title match, that article is displayed. Otherwise, they're given the Wikipedia full-text search results for that term, as its own node. In this way they can investigate each search result and return to the full list at any time, as they're all available in the map (or, if they find one correct result, they can easily delete the search results node). Currently the editor for note node content is placed in the article reader (sharing a horizontal split with the article content itself). However, users found this very confusing (see Results) and note editing functionality will likely be moved to a larger popover available for each note node.

The final interface component is the sessions list. Each search initiated from the welcome page begins a new session, and all inter-article link navigation and searches stay within that session. The user can pop open the sidebar to see a list of all their sessions. These can be renamed, deleted, and reordered (by dragging). The sessions allow users to have multiple simultaneous research projects.

RESULTS

We performed user studies during multiple stages of Wikitree's development. These included preliminary research interviews, paper prototype testing, and user testing of Wikitree in various stages of completeness. Our user testing centered around "think aloud" studies where we sat a user down with the Wikitree tool and had them accomplish simple tasks (and freely explore the system) while vocalizing their cognitive process. Through these testing sessions we found both good and bad qualities.

There were many attributes of Wikitree the users enjoyed. Users enjoyed the clean, simple aesthetic. Many liked the bounciness of the force graph, perceiving it as lively and engaging. User enjoyed the ease of entry: the overall concept of the tool was intuitive to most, and to some it seemed so obvious they were surprised it didn't already exist. Users enjoyed the branching aspect, as many started with simple trails and were delighted to see they could navigate back and start new branches off of older nodes. Users had fun trying to loop back on themselves. Users enjoyed the native Wikipedia look and feel. Users appreciated that their sessions were saved and awaiting their return. Almost all users we spoke with sympathized about the Wikipedia "rabbit hole" (getting distracted during research and pursuing long link trails) and the issue of having an unmanageable number of tabs open (one user at the 512 presentation night declared "I literally have 12 Wikipedia tabs open on my computer right now").

There were also many shortcomings discovered. Some shortcomings were system bugs and interface tweaks. For example, many users tried to use the built-in browser back button to travel to the previously visited article, which we had not hooked up (and, it turns out, a recent routing upgrade caused our system to handle it poorly, generating new sessions each time). No users intuitively understood the meaning of the dot when an article is "pinned" and many struggled with how and when pinning occurs (nodes are pinned after they're dragged). The node popovers were finicky and some users struggled to reach their buttons. The location of the note node editor was confusing, and overlooked by many (in the future, it will be added as an additional popover).

Other shortcomings were larger, with users wishing for things that would require significant development. Users wished for suggestion nodes that indicated where to travel next (and raised interesting questions about whether the suggestions should lead them deeper into similar articles or purposefully bring them into new but related areas of study). Users wished there was some indication of inter-node relevancy in the map, other than their own links (such as being able to see all links between nodes present in the Wikipedia network, or some sort of clustering based on relevancy). Users wanted a way to share the maps they created with others. Users wanted an indicator of which node they originated from. Users wanted indications of time: when was a node added? When was it last accessed? Users wanted to break out of the Wikipedia ecosystem and use it for general web browsing, or a larger scale of research. Users wanted more solidity and structure for the map, fearing that the network was too fluid and would result in their mental model being continuously deprecated as node positions shifted. These are discussed further in section Future Work.

General shortcomings in the system became apparent both through user testing and our own use. The network is prone to over-linking, and can easily become a tangled hairball if not pruned. The same is true of the note nodes, which can be useful if linked to nodes that are already nearby, but can bring clutter if linking nodes that are at distant areas in the graph (as the springiness of the links will pull the graph together). The entire system becomes sluggish and choppy with 20-50 nodes added, depending on what computer we're running it on.

Overall, we determined Wikitree as effective through the engagement users showed. As mentioned, many users sympathized with wandering through Wikipedia, and they were excited to have a map that could chart their journey. Without much prodding, users were happy to sit down and click around and build maps for themselves (they especially enjoyed the challenge of finding a way to loop back on themselves). A significant majority of users expressed interest in accessing the tool on their own time and were excited to learn that a public alpha is available online. There is much work to be done, but we're on the right path.

DISCUSSION

Wikitree shows the user more about what they already know. As one user said, "you can see how things connect together in a more concrete way." Wikitree seems to successfully reveal patterns in interlinked data as a user's journey unfolds. During testing, users uncovered neighborhoods of related articles (such as the cloud around Systems Theory and Emergence). It seemed that users were successfully encouraged to dig deeper, to fill out their map without fear of getting lost.

However, there were also many shortcomings. As mentioned in Results, the network diagram turned into an unmanageable hairball with too much interlinking. The diagrams were only interesting (and more importantly, readable) to a certain level of complexity. Also, users wished for more encodings about the content of the map. They wanted richer details about what they had already visited, and more information about the larger context of the areas they were moving through (more in Future Work).

Another revelation, somewhere between shortcoming and insight, was the structure of Wikipedia itself. Many articles have an overabundance of links, and as such there were surprising proximities. Users may move quickly through radically different topics, but by the few short links between them on the graph they would appear more related than they actually were. Here again, more analysis of Wikipedia's content and some sort of structuring to reflect this could help the user gain more accurate insights into the knowledge structures they're moving through.

FUTURE WORK

There is much work to be done with Wikitree, both in terms of its current implementation and in new directions the platform can be taken. The current implementation has room for both iterative improvements in its current feature set, as well as expansion with interesting additional features. New horizons also exist for the platform, taking the interface itself beyond the bounds of Wikipedia.

In terms of iterative improvements, there are many upgrades that could be made to the current implementation. Server-side user accounts and sessions storage would allow users to access their sessions from multiple devices. It would also give us a corpus for examining user patterns and deriving new node suggestions based on human behaviors. Server-side data storage would also allow us to implement sharing sessions, either through passing a copy or opening common access. Passing a copy would allow receiving users to make their own changes, without affecting the original creator's session. We could also track the spread of shares and manipulations, which could allow us to measure the "clout" of a given session (more shares more clout, and perhaps with more complex measures for how many derivative works it inspires?). Common access sharing could allow live collaboration, similar to a project group sharing a Google doc to collect their findings.

There are also many iterative encodings we could work into the current graph. As one user mentioned, it would be useful to know how long it was since he last visited a given node. To support this, node opacity could be adjusted with last time visited, so old nodes "faded" while more actively visited nodes remained clear. Article length could be encoded in the radius of the node. The time a user spent on a given article could be encoded on the node as well, perhaps in the radius of a ring or halo around a node (to allow for both size and time visited radius encodings). The distance of each node from the origin node could be encoded using a simple color ramp, creating a heatmap.

In addition to these more simple improvements, there are complex new features that could be added to Wikitree. One is relevancy analysis. Currently, each article is added naively, and the only indication of connection is the links added by the user's own interactions. Relevancy analysis could be conducted in many ways, such as the network structure of the local area around the current articles, the text content of the articles as measured against the Wikipedia corpus, or by using patterns derived from the prior interactions of other Wikitree users. These relevancy metrics could be used to add features such as node suggestions or clustering in the map. Nodes could be suggested to encourage discovery and serendipity, enabling the user to discover new material (the "suggest" of the research assistant). Nodes could be recommended to either help the user tighten their coverage within the local neighborhood they're exploring, or nodes could be suggested to break the user out into new, relevant areas. Relevancy analysis could also allow for clustering or coloring of nodes by shared categories or keywords. This would give the spatial encodings of the graph more meaning than simple physical distribution.

Another complex new feature could be article text highlighting and annotation. This would undoubtedly be a useful tool for researchers (a sentiment echoed by many users during testing). However, there would be nontrivial difficulty in creating and maintaining text selections for a living document such as Wikipedia. Since Wikipedia articles could be edited at any time, it may be difficult to track a highlighted portion of text, as that portion undergoes changes (or even complete deletion). Currently, Wikitree only caches Wikipedia articles in memory, so refreshing a page fetches the articles fresh from their API. This means that new edits can be loaded instantly, but it also means that text must be considered fluid and volatile. This would be an interesting problem to consider, reaching into the world of git and version control.

The Wikitree interface could also be applied to new frontiers. At its core, Wikitree is a map and an article reader. This concept could be applied to datasets other than Wikipedia. For example, many users expressed a desire to use a Wikitree-ish interface as a replacement for tabs. If the interface could be installed as a browser plugin, it could track a user's entire navigation history and allow them to store visited websites in a 2D network instead of the two sets of 1D tabs and history we use currently. It has also been suggested that Wikitree could be applied to academic papers, which have their own rich network of interconnections via their references. In a similar vein, it has been suggested lawyers working case law could use a Wikitree-like interface to track through precedence networks.

However grand or modest the scope becomes, we will continue to iterate on Wikitree. We would like it to be stable and useful enough to serve as an educational tool that could find its way into classrooms and homes (and maybe even onto Wikipedia itself).

ACKNOWLEDGMENTS

Wikitree was built with help from UW Informatics Capstone teammates James K Prow and Christina K Xiao. We thank the many users who have lent us their time for this project. We thank the numerous advisors at UW who have contributed their insights to this project, including Michael Freeman, Jeffrey Heer, Jessica Hullman, David McDonald, David Stearns, and Jevin West. We thank the Wikimedia foundation for their friendly API and commitment to free and open knowledge for the world.

REFERENCES

1. Bostock, M. D3 Force Layout. https://github.com/mbostock/d3/wiki/Force-Layout
2. Chau, D. H., Kittur, A., Hong, J. I., Faloutsos, C. Apolo: Making Sense of Large Network Data by Combining Rich User Interaction and Machine Learning. ACM CHI Conference on Human Factors in Computing Systems. (2011)
3. Infomap http://www.mapequation.org/index.html
4. Kairam, S., Riche, N. H., Drucker S., Fernandez R., and Heer J. Refinery: Visual Exploration of Large, Heterogeneous Networks through Associative Browsing. Eurographics Conference on Visualization. (2015)
5. Krempel, R. Local Wikipedia Map. http://lwmap.uni-koeln.de/
6. Lancichinetti, A., Rosvall, M., Infobaleen Wikipedia Map. http://www.infobaleen.com/wiki
7. MediaWiki API http://www.mediawiki.org/wiki/API:Main_page