Judicial Case Law Citation Timeline (December 2015)

Ken Mansfield
University of British Columbia
Department of Electrical and Computer Engineering

Abstract—Judicial Case Law Citation Timeline is an interactive visualization that looks at the relationships between case law citations (cross references) and case decisions (written judgements) related to the treatment of particular legal concepts over time for legal research and collaboration. The dataset comes from over 20,000 case laws contained in the BC Laws Database and will eventually be expanded to the over 2 million cases contained within the Canadian Legal Information Institute Database. The resulting visualization is presented as a Chord Diagram with each of the arcs representing a case, and each chord representing the citations between cases.

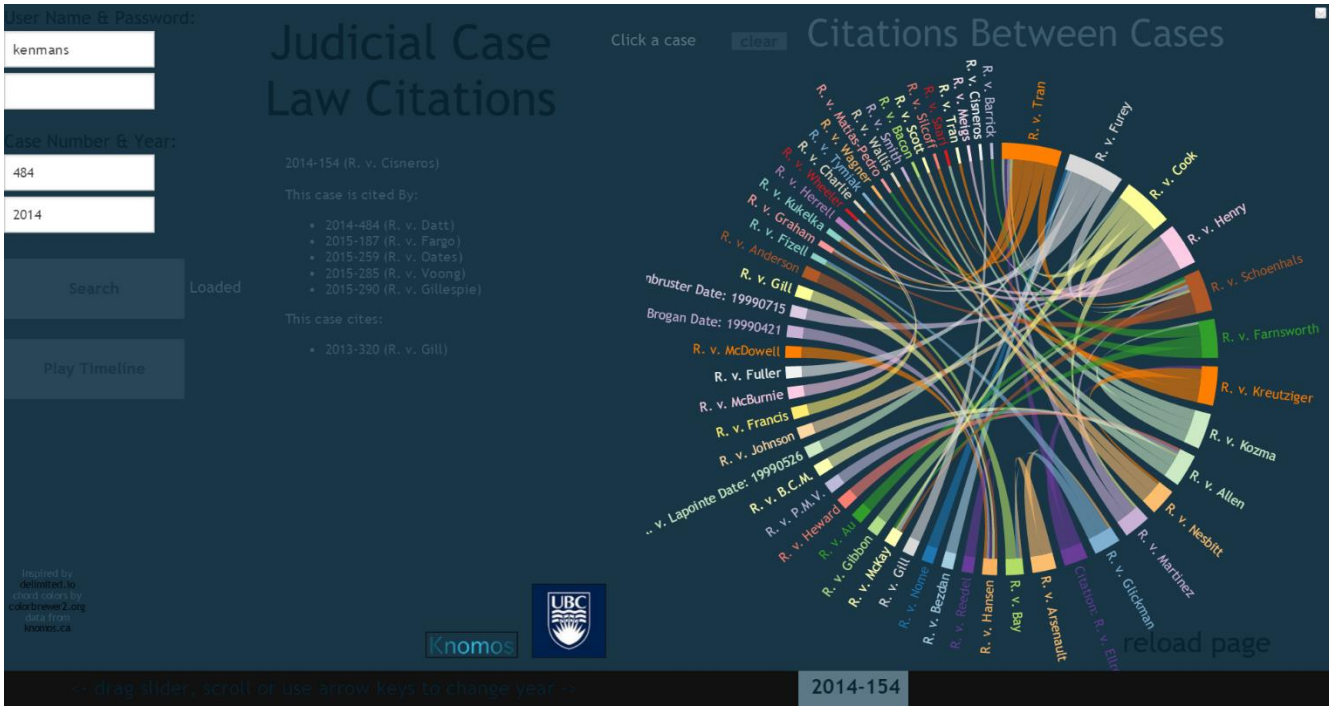

Figure 1. Full screen view of the Judicial Case Law Citation Timeline Visualization.

1 Introduction

CASE decisions are influenced greatly by the other cases that they cite. Consequently if a key case law decision has been made that is considered influential, then that case may become cited by many other cases.

The current process for researching case citations involves the use of legal databases such as CanLII [1] (The Canadian Legal Information Institute). The user enters a known case number and the database retrieves the case document. The user then has to read through the document to extract the other cases that were cited by this case. It can be an arduous process to then recursively search through each of the documents cited. Additionally, the online law databases make no mention of which cases that the case being viewed has been *cited by* because they are not part of the document. To address this I am proposing a visualization that displays the cross references between cases to enhance the research process. Furthermore, the visualization will display the changes in citations over time using animation.

This project is done in collaboration with a local legal

• *Ken Mansfield, UBC. E-mail: kmansfield@ece.ubc.ca.*

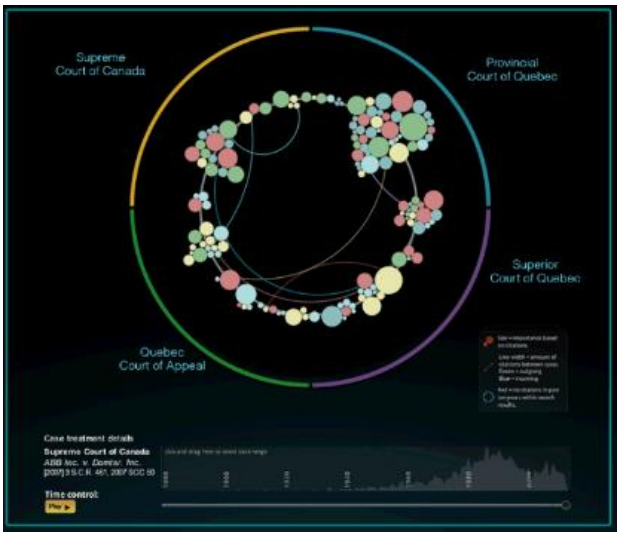

Figure 2: First proposal based on Events in the Game of Thrones Visualization

startup, Knomos [2], whose aim is to develop fully featured tool for visualizing law.

2 Related Work

Cross references between cases can be visualized as a network of citations. There are many popular ways to visualize network data but a unique solution must be found to achieve the results needed for this visualization. Many different visualization approaches were considered and helped to influence the final design.

2.1 Events in the Games of Thrones

The first example suggested by the Knomos team was the "Events in the Game of Thrones" visualization by Jerome Cukier [3]. This visualization uses an adapted form of the node-link idiom. Each link between one bubble node to another indicates a kill. The visualization is a very creative way to depict the interactions between Game of Thrones Characters, but it falls short as being useable as production level software. For example, there is no indication of who killed who (i.e. which direction) between nodes connected by lines. The visualization is also crowded with no labels on the bubbles, the user must hover over a node to determine the character's name and then hover over the node that it links to, to determine who that character killed or was killed by.

2.2 Initial Project Proposal

The initial proposed modification would use a similar layout as above:

- Each node represents a case decision, clustered by jurisdiction (e.g. Supreme Court of Canada).
- The size of the node represents the frequency of the case's citations throughout other cases decisions.
- The color of each node signals the relative treatment of that case decision in other cases (e.g. Green = favourable, yellow = distinguished, red = overturned).
- The lines between nodes represent interactions (the citations) between case decisions.
- The time control scrollbar can be shifted to adjust the visualization, and stopped at a specific point in time.
- Content scope can be filtered by including or restricting the years or jurisdictions selected beneath the scrollbar.

In the original "Events in the Game of Thrones" visualization the nodes were grouped by families and kills mostly happened between one family to another. For judicial cases, citations are most likely to come from the same group (i.e. within BC Laws). It is highly likely that all cases could all pertain within one jurisdiction so grouping the cases this way with lines (citations) looping back to the same group would be difficult to distinguish. Additionally, for this project we are limiting our data to one jurisdiction so the depicted clustering will not work.

A character is usually killed only once, meaning there is only one link coming into it. For judicial citations, there is likely to be a many to many relationship. With the large clusters in the Game of Thrones visualization it is hard to see where the lines are coming in and out.

2.3 Node-Link Layout

The most common idiom for visualizing network data is the Node-Link Diagram. The Judicial case data would be very easy to adapt to this approach as each case could be thought of as a node and the citations can be thought of as links. The d3 Force-Directed Graph visualization [4] is a popular example employed by many. One glaring issue with this approach is that text label occlusion is likely due to the location and density of nodes, as well as link occlusion in large clusters. The force directed graph is already used by Knomos as well so it is useful to study different approaches.

2.4 Other Social Networks

Citation networks can be thought of as a type of social network. Adapting this to judicial cases, we could think of the primary case as the "ego", and its children citations as "alters". A recent paper, egoSlider [5], specifically focuses on viewing the changes of an ego's network over time. The approach that they have taken would not lend itself well to judicial cases, however, because a case does not persist over time the same was a human does. Judicial cases have a single time-stamp so it would not make sense to analyze things like tie-strength.

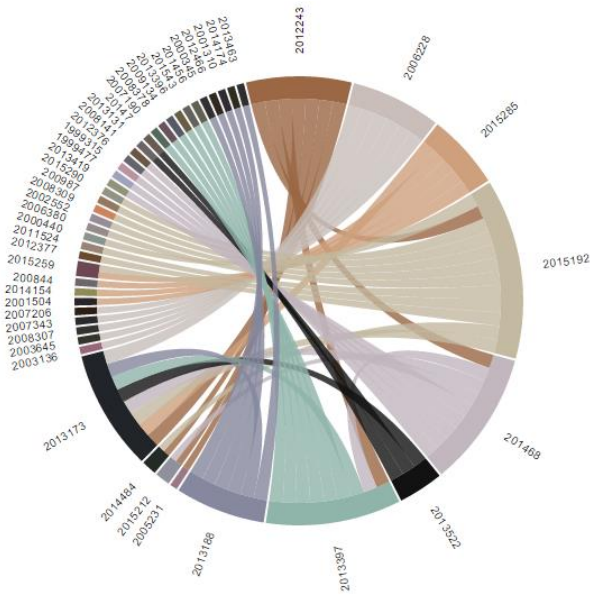

Figure 3: Initial Chord Diagram implementation with width of 1 for the source and target width of chords.

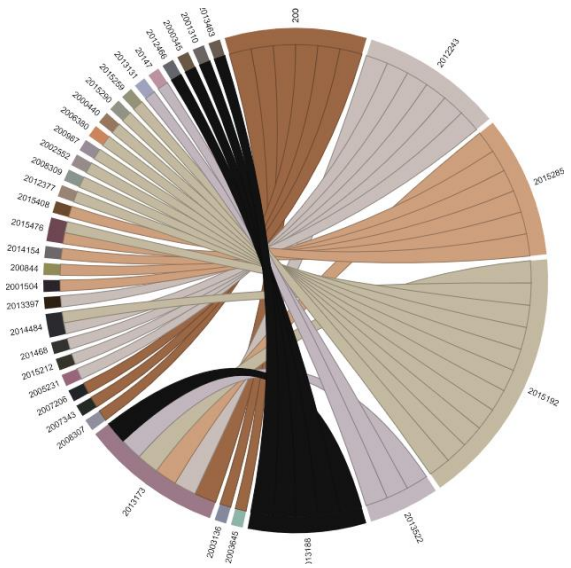

Figure 4: Initial chord diagram implementation with a width of 2 for the case being cited, and 1 for the case that is doing the citing. Note, the difference is very subtle.

2.5 Other Citation Networks

Autodesk has produced a useful visualization specifically for citation networks, Citeology [6]. Their visualization gives the user the ability to follow the genealogy of a different documents over time. There would be some value to applying a design similar to this for visualizing the genealogy of a case over multiple generations, as well as viewing the overall topology of the entire network. However, for our purposes we want to be able to quickly visualize individual cases which have received many citations and do not need an entire genealogy.

2.6 Chord Diagrams

The chord diagram idiom presents a unique way of displaying the relationships between different nodes (or arcs in this case). Most implementations of the chord diagram are used to represent flows, i.e. delimited.io [8] has used the diagram to represent international trade flows. The very first design of this visualization was influenced by the DependencyWheel [9] (figure 3), because the relationship of one case being cited by another case can be thought of as a form of dependency.

2.7 Hierarchical Edge Bundling

Hierarchical edge bundling is another unique design idiom that shares many of the same principles as the chord diagram. The labels are displayed on the arcs on a circle, and links form the connection between different items. The lines have fixed width and instead employ a tension factor to bundle individual links together that are flowing to the same areas within the circle.

The Eigenfactor project at the University of Washington along with Stephen Moritz [10][11] produced an interesting visualization that uses this idiom to form a citation network. The lack of link colouring and link widths makes it hard to decipher any form of directionality in the links, which is a key factor for the Case Law visualization. However, certain design choices such as having the inner arcs specify particular documents, and the outer arcs specifying more generic grouping could be useful for future work on the Case Law visualization.

3 Data and Task Abstraction

3.1 Domain Background: Law

Case law is the collection of reported cases that form the body of law within a given jurisdiction. Judicial cases can cite many cases and be cited by many more. These citations can be viewed as a network similar to a social network where children nodes can be linked to many other nodes.

3.2 Knomos API

The first step to being able to build the visualization is to acquire the correct data. Legal databases such as CanLII [1] provide access to all the individual cases. The text can be parsed for the all the cited cases and indexed into a new database. Once each case is associated with the cases that it *cites* we then know for each case which cases it has been *cited by* as well. The local startup, Knomos [2] has processed this data and stores it in a Neo4j graph database on a server. For the purposes of the visualization the data can be accessed via a REST API which returns structured JSON data. Each query returns information about an individual case as well as arrays containing the cases it cites and is cited by. For security reasons, browsers restrict cross-

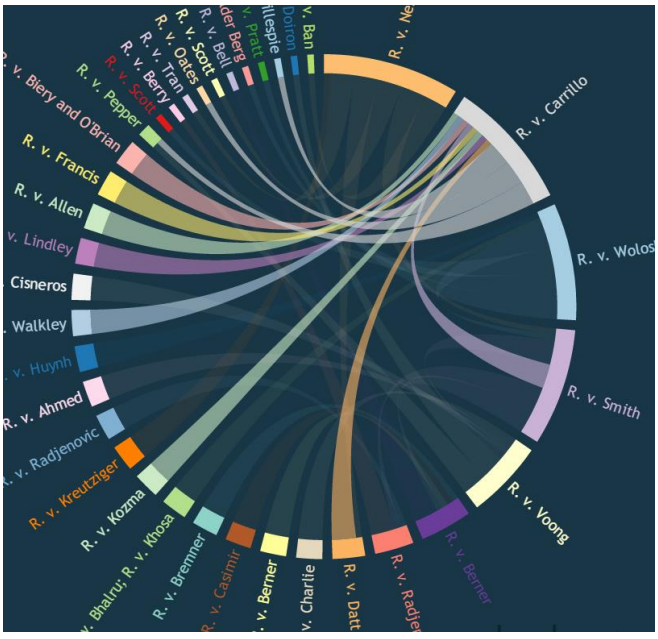

Figure 5: Final design of the chord diagram with the pointer (not pictured) hovering over the arc causing the connected chords to become highlighted.

origin HTTP requests initiated from within JavaScript, so the Knomos server team had to add a CORS header to the database to allow the visualization to access it.

4 Solution

4.1 Chord Diagram

For this visualization the chord diagram idiom has been chosen for its ability to encode a considerable amount of information with limited occlusion of important data.

Data-Driven Documents (d3) [12] has been chosen as the technology of choice by the collaboration partner, Knomos. [2]. The chord diagram example from the d3 website [7] provides example source code that is easily extensible. The first proof of concept iteration (figure 4), was to encode case citations in each of the chords and link cases together. This is achieved by constructing a matrix that defines these connections.

Chord diagrams provide us several ways to encode data by allowing us to change the width of the source chord, the width of the target chord, and the width of both of the arcs. The width of a glyph is useful for attributing importance however it can also be very difficult to visually interpret small differences in width. Colour encoding can also be used for the source chord, the target chord, and the both of the arcs.

Hovering over an arc or a chord can be used to trigger callbacks in which we can decide to dynamically filter or focus the view to highlight individual items or to show information in a separate view.

4.2 Timeline

To show case citations changing over time the data must be reduced for each time step. The visualization needs to indicate the current time step as well has have a way to move between time steps. To allow the user to follow the changes the visualization will animate between time steps.

4.3 Colours

There are many possibilities for use of colours in the chord diagram. For this visualization it was chosen to use colours that look distinct from each other (particularly chords that are next to each other). Having distinct colours allows the user to follow an individual chord from one side to the other. These colours were selected from palettes provided by Color Brewer [13]. The only thing that is encoded by colour is the *cited by* or *cites* relationship. A chord with the same colour as an arc indicates that that is the case that is being cited. A chord with a colour different than the arc indicates it is citing the other case.

An alternative approach might have used colours to indicate some sort of quantity, such as the number of times a case has been cited. This could be achieved by using a single hue sequential colour scheme with darker colours indicating more connections. This would make it easy for the user to find an important case, but it might make it harder to trace the chord between two arcs if all the chords have similar hues. It would be worthwhile to do more user studies, or perhaps allow the user to switch between options.

5 Implementation

5.1 Querying the DB

Adding a text field and a submit button enables the user to select the primary case they wish to build the chord-diagram for and initiates the load. Since several API requests are needed to build the matrix, the visualization takes approximately 20 seconds to load. A simple loading counter is provided to list how many requests have been made.

The user will be presented with an input box for year and case number. The two numbers together represent a unique identifier for a case, i.e. 2013-137 and will be used to make the initial query that starts the visualization. Each query to the database returns the data from a single case. This data would be enough to create a small network that contains the primary case and each of the cases that it cites, and is cited by. To expand the network, each of the primary cases citations must be queried as well. For the example of 2013-137 as our primary case, it contains 4 cases it *cites*, and 2 that it is *cited by*. The API is queried consecutively for each of the 6 case citations. These cases can be considered our 1st level cases. These first level cases may *cite* or be *cited by* other first level cases or contain citations that are not part of the primary cases. These cases that are cited or are cited by the first level cases, and not the primary case, can

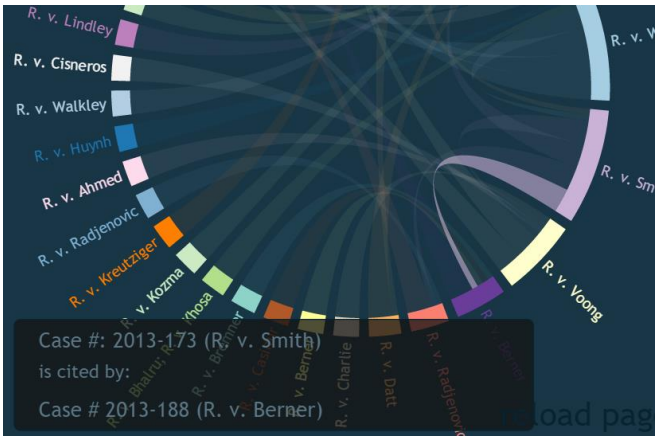

Figure 6: Hovering over an individual chord highlights that chord as well as brings up a window that displays more information.

be considered 2nd level cases. For the purpose of this visualization we are only recursively searching the primary case to a depth of 2 levels. More levels of recursion would extend the amount of time required to query the database and process the data. Additionally, the chord diagram has a limited number of chords that it can support before both the cognitive load becomes too high, and pixel resolution becomes an issue on standard monitors.

5.2 Building the Matrix

The input of the d3 Chord Diagram is a matrix. The value located at *Xi, Yi* represents the flow between case *i* and itself. The value at position *Xi, Yj* represents the flow from case *i* and case *j*, and the value at position at value represents the flow from case *j* to case *i*. These flows are represented as chord widths. For the purpose of this visualization the widths are being used to identify directionality in the form of a *cites* vs *cited by* relationship. A wider chord coming out of an arc indicates that the case is being *cited by* the case on the other side. So for a *cited by* position on our matrix (ie. *Xi, Yj*) we use the value 4, and the arc on the other side of the chord is the *cites* position (i.e. *Xj, Yi*) a value of 1 is chosen.

5.3 Slider

The package Dragdealer [14] (dragdealer.js) is used to implement the slider that determines the position on the timeline. Before we can display the slider we need to determine our timescale from the data. The nature of the data requires a non-linear timescale. It is possible that there could be one case in 1999 and 30 in 2012, for example.

Each case represents a single, unique point in time, ie. 2013-173 represents case #173 in 2013, so using individual cases is a logical choice for time steps. However, the problem with this is that we are visualizing pairs of cases rather than individual cases so the earliest case in our dataset will be *cited by* a case later in time. As we progress along the timeline the diagram will be filled with more cases with future dates. This is non-intuitive and also results in many

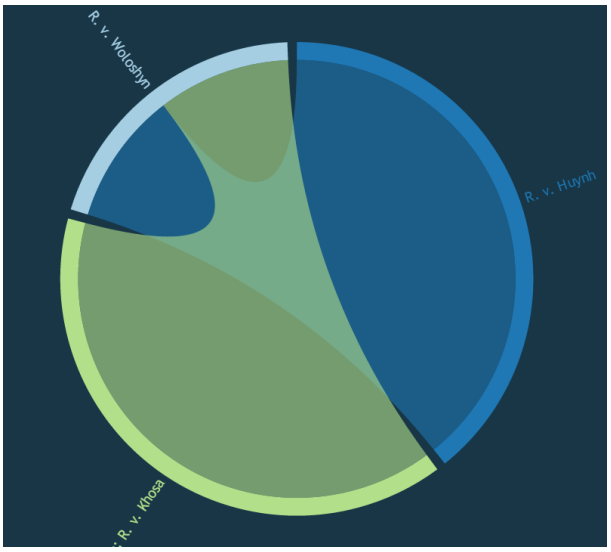

Figure 7: Chord diagram with only 3 cases on it.

slider movements where no new cases pop up on the diagram because they are already referenced by previous cases. To deter future cases showing up on the diagram we filter out cases that occur at a later date. The time steps are also restricted to cases which cite at least one other case to ensure that there is something to actually draw when the slider position is moved to that case. By the time the slider reaches the end of the timeline, all cases are on display on the diagram.

5.4 Animation

To allow the user to visually track the change in time the chords and arcs will animate between time steps. The implementation used is inspired by delimited.io [8] (interactive chord diagrams in d3). The scripts chordDirective.js and matrixFactory.js were borrowed from [8] with some modifications. Their code leverages the Angular javascript package (angular.js), however it is not absolutely necessary. Essentially, for each time step on the slider, a new matrix is being built from our timeline reduced dataset. When the slider is changed, new items are added (or removed) from the diagram. These items will be initialized to the default position and then their positions are interpolated to their final positions.

A Play/Stop Timeline button is included to allow the diagram to automatically animate from start to finish.

5.5 Other Views

Highlighting over a chord displays a tool tip with more information about the relationship between chords. For this relationship formatted text is used to display: *Case # (Case Title): is cited by: Case # (Case Title)* in a superimposed window (figure 6).

Another view in the center of the window displays the relationships between the case specified on the timeline slider and each of its citations. This is displayed as (figure

Figure 8: Centre window displaying detailed information about the current case on the timeline.

8):

Case # (Case Title)

This case cites:

- Case # (Case Title) for each case cited

This case is cited by:

- Case # (Case Title) for each case this case is cited by

This centre view is displayed by dynamically creating html objects in Javascript each time the slider moves and adding them to an html section.

5.6 Other Software

Foundation [15] was used for some of the CSS styling and paletton [16] was used to help come up with a website colour scheme. Github is being used to host the website.

6 Solution

6.1 Initial Attempts

A first attempt set the width of the chord to 1 on the *cited by* side and 0 on the *cites* side. Most of the 2nd degree cases are not cited by any other cases so they end up with an arc of 0 width causing their labels to be bunched together and occluded. A subsequent attempt (figure 3) set the width to 1 on each end. This reduced any text occlusion, however, it made it unclear what the relationship was between the two cases. Next a width of 2 on one end and a width of 1 on the other end was used to differentiate the relationship, but the difference was not easy enough to decipher with ease. A width of 4 for the case that is being cited vs a width of 1 for cases that the case cites made it much easier to differentiate between the two. This also has the effect of making the arc width wider for cases that are cited by many other cases increasing its importance, which is specifically useful for finding key cases.

6.2 User Opinions

Figure 9: View of a case that is cited by 3 cases (wide chords) and cites (thin chord with a different colour) only 1 other case.

From a development standpoint it is most useful to reference data by its unique identifier, in this case the combination of year and case # ie. 2013-173. These were initially used as the labels for each case on the chord diagram wheel. After speaking with 3 non-domain expert test subjects, it became clear that the visualization was not entirely intuitive. On consultation with a legal expert, the decision was made to display the case title (also known as a SOC) instead of the case number. For the tool-tip view (figure 6) and the detailed view in the center (figure 8) both the Case Year – Number and Case title are displayed.

6.3 Final Result

Approaching the experts and non-experts alike the consensus is that the final result is much more intuitive. *R. v. Mansfield* resembles a case decision much more than 2013-173. Additionally having text in the tooltip view as well as the centre view that explicitly states *cites* and *cited by* relationships helps train the user if they are not already aware of the relationship from the Chord Diagram.

6.4 Use Case

As mentioned earlier, the aim of the visualization is to aid the user in finding key cases which are cited by many other cases. The first step would require them to enter a case of interest in the text field and initiate the search. The visualization starts by playing the timeline from the beginning. Oftentimes a key case can be one that happened many years ago and has been cited by many cases since then. It that situation the case will show up earlier on in the timeline, and *cited by* links will be added to it as the slider progresses. It will be clear to the user that a case who has many links added to it is important.

The user can stop the animation at any time, or wait until it finishes. For the most part it is easy to follow the links between two cases visually. There exists some occlusion of chords in the centre of the diagram due to overlapping, however, this is alleviated somewhat by having non fully opaque chords so that you can see which chords are passing through which. It is also generally easy to follow a chord based on its arc and trajectory, as well as its unique colour.

In the case that it is not easy enough to follow the links between cases, the user may simply highlight an arc to see

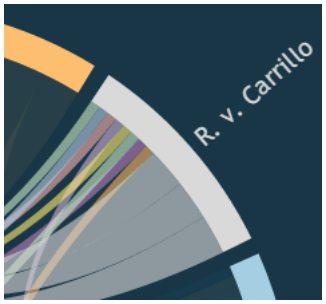

Figure 10: A case which is cited by only 3 cases however cites 8 cases.

which of the neighboring cases it links to. If there is a specific link that the user is interested in, they can hover over that chord, and more detailed information will pop up in an information box (figure 6).

The main way to differentiate a case that is cited by many cases between all the rest is by observing the width and colour of the chords, as well as the overall width of the arc. A key case will have many wide chords coming out of it. Those chords will also have the same colour as the arc of the key case (figure 9). Alternatively, a case that is not as important may choose to cite many other cases, but may not be cited by any other cases at all (figure 6). This will result in many thin chords of different colours coming out of its arc.

7 Discussion

7.1 Issues

The current database API only returns the data for a single case. It currently takes about 20 seconds to make 7 consecutive DB queries and then build the matrix and draw the visualization. The greatest amount of time is spent making the queries. There exists cases that make 20 or more queries and take over 45 seconds to load. This time could be reduced if the database were to return all the information we need in a single query. If a database change is not possible, another option would be to query the primary case to receive the list of other cases we need to query, and then send out all the subsequent queries asynchronously all at once rather than consecutively.

7.2 Scale

Conceivably there may be cases with hundreds of citations. This could cause problems with the current implementation. The data would take a considerable time to load as well as become very slow. Building the required matrix between each time step is a CPU intensive process, as is interpolating between positions during animation. A greater problem may lie in the cognitive load of trying to visually process a diagram with too much information. Additionally, an overly crowded diagram will most definitely result in significant occlusion of chords and labels.

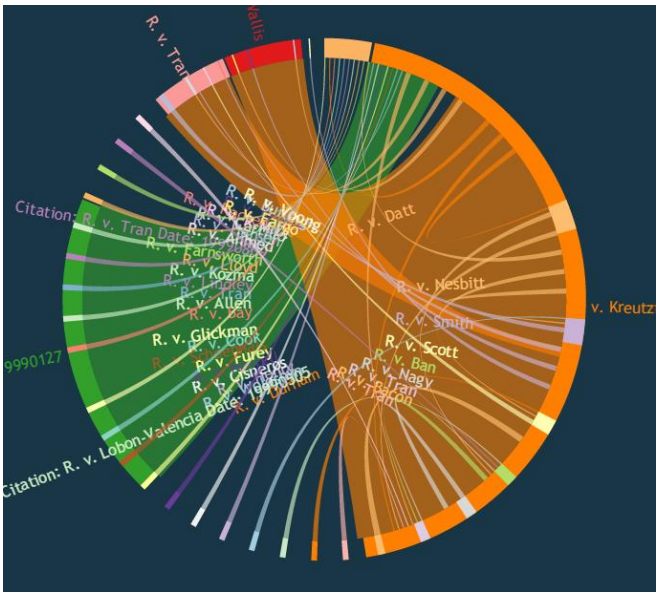

Figure 11: View of the chord diagram in the middle of animating.

To avoid overloading the diagram there are multiple options. The first option would be to not load any 2nd level cases, i.e. only show the cases that are referenced by the primary case. Another option would be to only show cases that have a certain number of *cited by* links.

There also exists the possibility that the user loads a case which has very few citations, resulting in a diagram with very few cases on it. This is less of a problem than loading too many cases, but the diagram could be augmented by searching 3rd level/depth cases as well.

7.3 Different Devices

The visualization was made with scaling in mind so that it can function on various devices. The result is that it works on mobile devices in either portrait or landscape, or in browser windows of different sizes/resolutions. However, there are additional things that could be considered to try to make the visualization work better on smaller less powerful devices. The slider is very easy to use with a mouse or a keyboard, but it is harder to use with a small touch screen. Making it much larger for a mobile device would be helpful.

The visualization was tested on a quad core i7-4770k, a Core M laptop, and an Android phone. Diagrams with up-to 120 cases worked flawlessly on the desktop computer. But the laptop and Android device slowed down when the number of cases surpassed around 60. To account for devices of different processing power, having a checkbox to limit the visualization to 60 cases might be useful.

7.4 Other Future Work

An obvious choice for future iterations would be to allow the user to click on a case to quickly load a new diagram

centric to that case. Given that it can take 20 seconds or longer for the diagram to load it did not seem as though it would be a great time saver over having to enter the case into the form and hitting search again.

7.5 Critique

After the first iteration with a static diagram with uniform width and poor colour choices (figure 3), it was not clear whether using a chord diagram was a good choice. It was hard to determine the relationship between cases, and the diagram was non-intuitive. However, after resolving many issues it has become clear that the chord diagram is a powerful tool for displaying information. Adding many new features to the visualization has also helped to make the tool more intuitive.

8 Conclusion

Legal research can be done more efficiently with a well-developed visual tool. The Judicial Case Law Citation Timeline visualization has shown that it can be used to effectively visualize a case law citation network, and help identify key cases. The slow process of reading through many documents is no longer necessary and can be replaced with tools that display all the information more effectively.

References

[1] The Canadian Legal Information Institute, "CanLII," https://www.canlii.org. 2015

[2] Adam La France, Jesse Abney, "Knomos," www.knomos.ca. 2015.

[3] Jerome Cukier, "Events in the Game of Thrones," http://www.jeromecukier.net/projects/agot/events.html. 2015.

[4] Mike Bostock, "Force-Directed Graph," http://bl.ocks.org/mbostock/4062045. 2012.

[5] Yanhong Wu, Naveen Pitipornvivat, Jian Zhao, Sixiao Yang, Guowei Huang, Huamin Qu, " egoSlider: Visual Analysis of Egocentric Network Evolution," *IEEE Trans. Visualization and Computer Graphics*, vol. 22, no. 1, Jan 2016, (IEEE Transactions)

[6] Justin Matejka, Tovi Grossman, George Fitzmaurice, "Citeology: Visualizing Paper Genealogy," *Autodesk Research*, Alt.chi, 2012

[7] Mike Bostock, "Data-Driven Documents," d3js.org. 2015.

[8] Stephen Hall (delimited.io), "Interactive Chord Diagrams in D3," http://www.delimited.io/blog/2014/11/18/interactive-chord-diagrams-in-d3. 2014

[9] François Zaninotto, "Dependency Wheel," http://www.redotheweb.com/DependencyWheel/. 2013.

[10] J.D. West, "Eigenfactor: Ranking and Mapping Scientific Knowledge," Pg. 137, PhD Dissertation. University of Washington. 2010

[11] Stephen Moritz, "Well-Formed Eigenfactor: Visualizing information flow in science," http://truth-and-beauty.net/projects/well-formed-eigenfactor. 2010

[12] Mike Bostock, "Chord Diagram," http://bl.ocks.org/mbostock/4062006. 2012.

[13] Cynthia Brewer, Mark Harrower, "Color Brewer 2.0: Color Advice for Cartography," http://colorbrewer2.org. 2015

[14] Ovidiu Chereches, "Dragdealer.js," http://skidding.github.io/dragdealer. 2015

[15] Zurb, "Foundation," http://foundation.zurb.com. 2015

[16] Paletton, "Paletton: In Love with Colors," http://paletton.com. 2015