Bibly – Visualizing Information Trails in the Bible

Gary Lee and Anirudh Venkatesh
Computer Science Department
Stanford University
{garylee | anirudhv}@cs.stanford.edu

ABSTRACT
The Bible is an epic account of humanity with a profound cultural influence and significance. For thousands of years, reading and interpreting the Bible has been a topic of great interest in religious, academic, and general public circles alike. However, the Bible as a physical document is unwieldy: with 40 authors, 66 chapters, and over 700,000 words, it is difficult — even for experts — to traverse methodically. Without a higher-level view of the Bible, trends and themes are hidden behind pages. Organizational meta-data such as books and key events that could serve as strong navigational cues are similarly tethered to pages, and are not leveraged to their fullest potential.

In this paper, we outline the key tasks that navigating the Bible entails. Specifically, we interview seminary students and pastors to see how they search for information in the Bible, and what kinds tasks, tools, and pain points they encounter. We synthesize this with information foraging theory, text visualization techniques, and interaction, all of which lend structure to information traversal. From there, our challenge becomes clear: we need a single interface that supports search, multiple zoom resolutions, granularities, and comparisons across different contexts, all while providing a contextual meta-data information trail for rapid exploration. Finally, we present and analyze Bibly, the tool we built to address this problem, and the new interaction paradigms it introduces.

Author Keywords
Text visualization, information foraging, user interface

ACM Classification Keywords
H5.2 Graphical User Interfaces

INTRODUCTION
The Bible as a text document presents many innate barriers to analysis. Numerous authors contributed to the Bible over a huge span of time (over 1600 years), resulting in extensive stylistic, linguistic, and thematic fragmentation. The Old Testament is largely narrative driven, whereas the New Testament is much more didactic in nature; in addition to different authorship, we encounter artifacts due to transliteration between Hebrew, Greek, Aramaic, and English — interpreting even a single word becomes exceptionally difficult due to the multiplicity of usage contexts throughout the Bible. Furthermore, the sheer volume of words in the Bible makes it very difficult to locate prominent themes, which may be scattered far and wide throughout. With just the physical document, we cannot easily gauge relative text distribution between spatial regions of the Bible and draw comparisons between them. Finally, much of the implicit (historical links) and explicit (books and verses) meta-data is trapped behind physical pages, and cannot readily be utilized as structural and spatial cues for organization and understanding.

We can illustrate the problem with a task: which Biblical passages discuss ‘romantic love’? With just the text and no cognitive aids, this is very difficult to do. First, we need a high-level view to see the hot spots where ‘love’ occurs throughout the Bible. Among these, we need to be able to compare local use cases to see which spots constitute an appropriate treatment of ‘love’ (e.g. love of the law vs. familial love vs. spiritual love, etc.). We have no way of quickly gleaning a single local use context from the raw text itself, aside from reading over the entire segment; comparing usage across many contexts, then, becomes exceedingly more laborious. And how exactly do we define local? How can we empower the user to systematically partition the Bible by specific book or verse, or give them the freedom to define scope? Ultimately, when we have zeroed-in, we want to link each of these contexts to the source text itself.

Dissecting the above task reveals three distinct levels of navigational hierarchy. 1) A bird’s-eye view that draws attention to hot spots, 2) a middle ground that provides contextual overviews and comparisons between regions of interest, and 3) bottoming out with the source text of the Bible. We can draw clear parallels here with information foraging theory as laid out by Pirolli and Card [3], where the user performs an iterative search and at each step, based on cues in the current information environment
(information scent), determines whether to dig deeper, zoom out, or move somewhere else altogether.

The primary goal of Bibly is to allow for a seamless and systematic traversal between these three navigational levels, with the user building his or her traversal path based on a strong information scent at each level. At each level we want to give the user tools to compare and illuminate regions of interest, intuitive handles to travel between different levels or start anew, and spatial awareness of where they are within the information hierarchy of the Bible. We want to supplement this tiered navigational approach with rich Biblical meta-data at each level (books, verses and historical context), which serves the dual purpose of boosting information scent and giving structure to the text visualization.

Moving forward, the paper will examine the specific design decisions and implementations that resulted from this task analysis, user studies that we conducted in order to test and iterate on our design, and future directions for our work.

RELATED WORK
As previously mentioned, information foraging theory was first proposed and investigated by Pirolli and Card at Xerox PARC [3]. This work introduces the Scatter/Gather Interface, whereby a user can navigate through the contents of a document at different levels of granularity, by systematically organizing it into a cluster hierarchy and viewing each cluster as a high-level summary of the subset it represents. Pirolli, Card, and Van Der Wege later apply information foraging theory and information scent to the task of searching large tree structures, and introduce the Hyperbolic Tree Browser as a visual encoding and interaction tailored to the task of navigating hierarchical trees. Our goal was to apply the theories put forth by Pirolli and Card to produce an interactive visualization tailored to the task of navigating a text document — specifically, the Bible.

We can look to established text visualization techniques as inspirations for visualizing the Bible. Concordance, which finds all occurrences of single word and shows the words flanking it, is a great technique for highlighting use context. There also exist visualization techniques that give a very zoomed out view of word distribution in a text document and link back to the word in context, as demonstrated in SeeSoft (Figure 3) [2] and The State of the Union in Words [5]. Concordance and the zoomed-out view will certainly come into play in our tool.

This brings us to the last category of related work: visualizations of the Bible. Much of the previous work in this domain revolves around a class of visualizations of Biblical text resulting from running general text analysis techniques (Figure 1) on the raw text of the Bible. In other words, these visualizations use little or no auxiliary information (books, verses, and historical context), relying completely on characteristics of the text itself. One characteristic example involves using general Natural Language Processing techniques to produce a node-linked diagram that visualizes the relation "X begat Y" in the Bible [6]. Similarly, there are visualizations that portray cross-referencing across the Bible [8], and ones that generate a "social network" of the Bible that links two people together if they occur together in a verse [9].

Such visualizations are effective in highlighting already-present relationships between words in the raw text of the Bible, but do not enable the user to make new connections beyond what is immediately depicted by the visualization. Furthermore, we see that many of these visualizations illustrate trends but do not link them back to the source — the spatial geography of where they actually occur in the Bible, down to the verse reference.

Clearly, the aforementioned visualizations of the Bible are static, and do not grant the user the freedom to drive exploration and make custom searches. Even those that are interactive, such as the Bible vs. Qur'an word frequency visualization (Figure 2) [7] and SeeSoft (Figure 3) [2] are not conducive to utilizing the human capacity for information foraging as Pirolli and Card have discussed, since they are missing a middle resolution for quick analysis — the user might be able to pinpoint a high-level trend and view the source, but cannot brush over multiple custom regions of interest and compare their context. This creates a gap in the user’s explorative experience: there is a very high-level information scent given by the arresting visual cues of word distribution, but once we locate regions of interest there are no analytical tools for defining them, summarizing their context, or making connections between them. Thus, these visualizations bottom out at the source too soon, and the user is left to manually sift through each of these results and make sense of them, bringing us back to the difficulties of working with a physical text document.

Figure 1. Phrase Net: ‘X’ begat ‘Y’

Figure 2: Bible vs Qur’an (left), Figure 3: SeeSoft (right)
NEEDFINDING
Through interviews with one seminary student and two pastors, we were able to identify the experiences that lead users encounter in Bible analysis. Specifically, we can break our observations and findings down into four categories: key tasks, example workflow, existing tools, and pain points.

Key Tasks
All three subjects we interviewed expressed the same primary task: interpretation. Commonly raised questions: What does this passage mean? What comparisons can I make to other passages that might illuminate its meaning? What was the authorial intention? Interpretation results in a lucid understanding of what certain phrases or passages mean in context and drawing connections between regions bound by a common theme or which illustrate different facets of a particular theme.

Example Workflow
One of our subjects, a pastor, gave us the following workflow, which he had encountered in preparing for a sermon:

In Revelation, we see the term “overcomer” or “he who overcomes” (depending on the translation). It is a difficult term to understand the meaning of, and I could do a direct search of the whole Bible and look at every instance of the word “overcome.” But the trail should start at the author of Revelation — we know that the author of Revelation is the Apostle John, so it makes sense to see if the word turns up in his other writings: the Gospel of John, 1 John, 2 John, and 3 John. It narrows down the field much more quickly to browse by books to which you can relate shared context. In fact, you can see that 1 John 5:5 defines “he who overcomes” as “he who believes that Jesus is the Son of God,” and you can then interpret the usage in Revelation that much more clearly.

This investigative procedure corresponds precisely to the information trail that Pirolli and Card posit — in order to locate other usages of “overcomer” the pastor begins at the highest level of search in determining where the word even appears in the Bible. He then proceeds to narrowing down the search field by books of interest, in this case corresponding to his knowledge of common authorship, and checks to see if the word turns up there. When the word “overcome” does turn up in this narrowed scope, he analyzes trends in its local usage to elucidate what it means elsewhere.

Existing Tools
Our subjects’ tools were primarily book-based. They all referred to hard copies of commentaries or Topic Bibles, where the text of the Bible itself is annotated throughout with contextual overviews and meanings of each passage. As far as software, all three subjects used BibleWorks (Figure 4), primarily for its word and phrase search that returns every single verse where the query is found. [1]

![Figure 4: BibleWorks search](image)

Pain Points
Both the interviews and the example workflow reveal the difficulty in manually interpreting and comparing local use cases, and not being able to define multiple custom ranges for examination and summary. Additionally, our subjects emphasized that the different tools they used were very disconnected — that is, there is a strong mental foundation involved (reading the Bible over and over again to have a good sense for spatiality and themes by book or chapter), but this often must be augmented by search software, and finally traced back to a Topic Bible or the source itself. Our subjects frequently found themselves jumping between these three different media, and experiencing workflow discontinuities each time they would have to map results between different tools in order to proceed.

METHOD AND DESIGN
Synthesizing information foraging theory, previous text visualization work, and user-specific needfinding, we arrived at a list of design goals that are critical to a fluid, interactive interface for navigating the Bible. Our guiding principles are information scent and hierarchical search, and these themes set the stage for the features and interactions we provide at each step. To that point, we organized our design goals by the level of search they correspond to — bird’s eye, middle ground, and bottoming out — and describe the design process for each one.

Figure 5: Bird’s eye view of query

Bird’s Eye
This is the top level of navigation and where the user begins. From this vantage point, we want to provide a

![Figure 5: Bird’s eye view of query](image)
simple, engaging search mechanism to give a clear picture of word distribution and regions of interest. This addresses the task of finding where a word appears in the Bible.

**Bar Graph in Bible Order**

We segmented the Bible into groups of words in Bible order and arrange them (left to right) along the $x$-axis. Each bar’s height represents the searched word’s frequency in that $x$-group (Figure 5). This gives the user immediate visceral feedback about that word’s density throughout the Bible. This is a vast improvement over a textual result list, since the words appear true to their spatial location throughout the Bible and we can very quickly get a sense of distribution patterns in space (between books, in relation to historical events, over elapsed time, etc.).

**Verse Annotation and Book Segmentation**

Since we use $x$-space to encode the progression of words in the Bible, it is critical we also let the user know specifically where they are at any value $x$. The user can hover over any bar to determine the book, chapter, and verse that bar represents. To provide additional context, we segmented the $x$-axis with alternating colors (similar to genome visualizations) that delineate precise book boundaries down to the pixel. Finally, we give historical annotations below the $x$-axis that correspond to the precise group of words where that event is described.

![Figure 6: Small multiples (left) and Figure 7: Multi-brush with annotations (right)](image)

**Small Multiples and Search History**

We also want the user to be able to compare results across queries — with a small multiples panel of recent searches, the user can compare high-level trends, correlations, and discrepancies between different queries, and use this feedback drive new directions for search. (Figure 6)

**Middle Ground**

At this level of navigation, the user has already located regions of interest, made clear by the bird’s eye view. From here, the primary task is to define regions and make comparisons of local use case, which corresponds to the key task we identified in user testing of comparing usage context in discrete regions of interest. The user can carve out any number of regions for further inspection, and we should provide a contextual information scent trail such that they can compare information scent between next steps.

**Advanced Brushing**

How can we reimagine the brush such that it can be multiplexed to enable contextual comparison? We want to allow for any number of user-defined brushed regions, such that we can provide summaries for any range parameters the user drags out, whether it’s over a part of a book, a particular book, or any arbitrary range within the Bible. A user can locate any region of interest and drag out a brush for it, either by book or by dragging along the graph to define a custom region. When a region is brushed out, the user gets feedback as to what verses the region is bounded by, as well as aggregate query word count within that region. Brushes persist through searches, to facilitate the task of comparing queries for different words within a single focused region (for example, does “faith” or “love” appear more often in Revelation?). Finally, we support multi-brushing, so that a user can launch an indefinite number of independent brushes, and manipulate them at will (resize, move, delete) — this allows for contextual comparison of a query across multiple books or ranges, which addresses a primary user task. (Figure 7)

![Figure 8: Concordance and verse linking](image)

**Concordances**

Once we have our brushed regions, we want to be able to glean our query word’s local use case within each region and compare them. To that end, whenever a user focuses on a brushed region, they can view the query word’s concordances within that region, and launch a new, independent concordance panel for each of their multiple brushes. (Figure 8)

**Bottoming Out**

After a user has tracked a word all the way from the high-level bar distribution, through comparisons of specific regions of interest and concordance lists, we want to end by anchoring our findings to the source text of the Bible itself.

**Verse Linking**

Once a user has followed an information path all the way to the concordance list, for each entry in the list, we provide a link back to the verse. A user can also link to the source from the verse annotations that appear over the bars.
IMPLEMENTATION

In this section, we will discuss the specific implementation details that we took to realize our design principles and translate them into a fully functional web application, Bibly. We processed the text using Python, and the end product is built entirely in the browser using the D3 framework, jQuery, HTML, and CSS. Accordingly, our implementation details are divided into back-end text processing and front-end interactivity.

Text Processing

In order for the user to be able to navigate the Bible rapidly and iteratively, we realized we must ensure that the results for any possible query must be very responsive and displayed without any delay. This posed the interesting implementation problem of how to manage the entire contents of the Bible within our visualization so as to efficiently compute word frequencies over an arbitrary range of verses or books. Ultimately, we decided that the best approach would be to perform a series of text processing steps to slice the text into appropriately-sized chunks, pre-compute aggregate and by-verse frequencies for each word in the chunk, write the data to a file in a browser language (JSON), and feed the resultant data structure directly into the browser application for efficient in-place querying without a server. We wrote all of our text processing scripts in the Python programming language, and, built two JSON data structures: grouped maps of words to frequency and verse, and a verse-to-source lookup map. The procedure is described below:

Filtering Text, Tagging Words

In the first pass of text processing, we took in the raw text file of the King James Bible, iterated through each word in order, and output it along with the book, chapter and verse it occurs in within the text. We filtered out stop words at this stage by ignoring any word specified as a stop word by the Natural Language Toolkit (NLTK) Python library. Our rationale behind doing this was that stop words, by virtue of their sheer frequency, would overshadow other words that would likely be more interesting to analyze in our application. In addition, we filtered out punctuation since our application only looks at the occurrences of a word, and hence, is agnostic to the punctuation used along with it.

Chunking into Groups

The next step involved taking the result of the first pass, the in-order list of roughly 360,000 words and their associated Book, Chapter, and Verse information, to cluster words into groupings of 300 words. We came up with this number for the size of our word groupings by observing that we have roughly 1200 pixels in the x-dimension of our display. Since we wanted to lay out the words of the Bible contiguously from left to right on the x-axis and each grouping to correspond to 1 pixel on our display, the division came up with a grouping size of 300 words.

Pre-computing Meta-data

Once we had clustered the words into groupings, we then pre-computed pieces various pieces of meta-data for each grouping. Specifically, we pre-computed the start and end verse range that encompasses all words in that grouping. For each word within the grouping, we computed the aggregate frequency of that word over the entire grouping, and for each verse that contains that word, a mapping of the verse name to the number of times the word appears in that particular verse. We then packaged this information into a JSON object and wrote it to a file so that it could be read directly by the application. This completes the first JSON data structure, the grouped maps of word to frequency and verse.

Building the Verse-to-Source Map

In addition to this pre-computed breakdown of frequencies by word grouping, our application also needs to support efficient lookup of the text of a particular verse, as described by the bottoming out level of zoom, where we link back to the source text. For this purpose, we also the generated a file similar to the one resulting from Filtering and Tagging stage described above, in which we output the text of each verse along with its verse number, and the book, chapter, and verse it appears in. Using this, we built a single JSON object mapping each verse name to its text and read it into our browser application. This completes the second JSON data structure, the verse-to-source lookup map.

Front-end Interactivity

The primary charting, animation, and book segmentation is implemented in D3. We used jQuery to add lots of additional interactivity such as tooltips, concordance lists, and verse linkage. We detail new interactions and technically notable features below:

Advanced Brushing

The most challenging feature from a technical standpoint is the multi-brush, which allows for a user to conjure an indefinite number of independently functioning brushed regions along a single axis. To our knowledge, this is a novel interaction technique, and the D3 framework does not support this. To accomplish this effect, we had to deftly manipulate JavaScript data structures, D3, and SVG canvas elements. Essentially what happens when a brush is multiplexed is that there is only one live D3 brush at any given time (the focused brush), whereas the rest are simply SVG rectangles that are stand-ins for brushed regions until they come into focus on mouse hover. At that point, we swap the attributes of the focused stand-in with the current live brush, so that the live brush always remains as the topmost element in the SVG canvas, which allows it to be resized and dragged. Likewise, whenever we add a brush, we always add it behind the live brush, to respect the ordering of elements. To keep track of the multiple brushed regions for the correct bar coloring, we use an array of 2-tuples, each of which represents the extent of a single brush.
Book and Verse Annotations, Concordance

Our choice of one pixel per bar coupled with our pre-processed data made it particularly convenient to map metadata to exact pixel locations. We read in our JSON arrays of start and end verses for each grouping, and for the query word its verse appearance across each group; the index within the array is precisely the pixel offset in both cases (to produce the verse reference brushing and verse appearances lists, respectively).

RESULTS

Reviewing our list of design goals and bringing them to life, we produced a final product: Bibly. The application fully embodies all of our stated design goals and more importantly, it facilitates rapid and systematic navigation of the Bible based on a hierarchical information trail. We can see, in Figure 9, all of our proposed visualization techniques together in a seamless interactive application. In this section, we will evaluate our approach with performance metrics, evaluation based on a sample use scenario, and user feedback.

Figure 9: Bibly, final user interface

Performance and Efficacy

As a result of our choices in data structures, architecture and traversal algorithms, all query results, verse reference meta-data, and concordance panels within Bibly come back instantaneously — this is exactly what we hoped to achieve to give the user the most responsive, immersive interface necessary for rapid exploration and iteration.

Evaluation: ‘Romantic Love’ Revisited

To illustrate the efficacy of our application in solving the task of locating a theme in the Bible, let’s revisit the question we asked earlier: Which Biblical passages discuss ‘romantic love’? How might we use Bibly to answer this question?

1) First, we can search for the word ‘love’. This produces immediate clues as to areas where the word is prominent.

2) Brushing by book, we can see that a huge peak on the right corresponds to 1 John, a peak in the middle corresponds to Song of Solomon, and maybe we’re curious about how the book of Psalm treats love.

3) Showing concordances for each selected region, we can make a clear assessment of context:

   In the book of Psalm, we see “Oh how I love thy law!”, “…love thy testimonies”, and “…love the LORD” repeated many times.

   In the book of 1 John, we see “love one another” is very prominent.

   In Song of Solomon we see “love; behold, thou art fair”, “thy love is better than wine”

4) We can conclude that Song of Solomon features love prominently, and indeed discusses it in the romantic sense. From here, we can zoom in to the text itself for a closer look.

DISCUSSION

User Follow-up

We conducted another series of interviews with the same subjects after the final product was finished, to see whether this could be a real asset to our intended audience and whether their mental model and interaction flow matched that which we had imagined for our users.

All of our subjects found the tool to be a substantial time improvement over their scattered-media approach, and found it to be very useful for quick drill-downs into thematic comparison. One of the subjects that used BibleWorks much more extensively than the other two really appreciated that our interface wasn’t purely text spreadsheets. He noted that the graphical visualization and interaction techniques complemented the text anchoring very well, with the density map and brushing capabilities, and concordances allowing him to rapidly structure outlines for small group Bible study, where the objective of a session is often to exegete a particular region of the Bible, explore thematic elements, and make connections. Overall, he felt that it was a much more practical solution to creating lesson plans around the Bible, compared to a lot of existing heavyweight tools which might be more appropriate for linguistic analysis, as one might do in seminary.

New Interaction Paradigms

New interaction paradigms that came out of our project include brushing with meta-data and multi-brushing along a single axis. These techniques are particularly well suited to information foraging within the Bible, which is rife with navigational meta-data, and where allowing the user to
track and compare results across multiple regions of interest is a very useful application.

**FUTURE WORK**

**N-gram and Regular Expression Inputs**
One limitation of our visualization is the inability to compare co-occurrence of multiple words or n-grams and wildcard regular expressions such as those inputted into Phrase Nets [6].

**Multiple Translation Support**
Furthermore, our subjects suggested edition and language support as a way to bridge artifacts of translation and account — for example, a search for ‘faith’ reveals that it only appears in the New Testament, when in fact there are many nuanced Hebrew expressions for faith in the Old Testament which simply aren’t captured by the English language.

**Natural Language Processing**
We would also like to move beyond concordances in our comparison of local usage — perhaps leveraging NLP techniques such as sentiment analysis and text visualizations such as WordTree.

**CONCLUSION**
We introduce the problem of text visualization as it applies to the Bible, a very lengthy corpus loaded with linguistic and thematic fragmentation. Our primary task was to stitch together the content of the Bible, its immense contextual metadata, and interactive visualization techniques to illuminate understanding. To that end, we dove into an exploration of user needs and key tasks that seminary students and pastors encounter in analyzing the Bible. We also looked to previous literature on information foraging as a manual on how humans systematically follow information trails. We synthesized task-specific user observations and information foraging theory, and combine them in an interactive graphical interface, Bibly. Bibly is anchored strongly to the idea that the Bible can be traversed systematically, and that fundamentally, we must provide access to multiple zoom levels and information scent at each level to guide iterative exploration. To our navigational hierarchy, we introduce the visualization techniques of word distribution bars, context-aware multi-brushing, book and verse annotation, and concordances to make it a fully featured and immersive experience.

Overall, we see our work as being applicable to navigating any well-formed text document — particularly large texts, which are difficult to view holistically and where it might be very illuminating to slice the text into organized substructures and make contextual comparisons between them.

**REFERENCES**
5. The State of the Union in Words [online].
7. Word Frequency Comparison Between the Bible and the Quran [online].
8. Bible Cross-References [online].