Vispedia*: Interactive Visual Exploration of Wikipedia Data via Search-Based Integration

Bryan Chan, Leslie Wu, Justin Talbot, Mike Cammarano, Pat Hanrahan

Abstract—
Wikipedia is an example of the collaborative, semi-structured data sets emerging on the Web. These data sets have large, non-uniform schema that require costly data integration, making visualization difficult.
We present Vispedia, a system that reduces the cost of data integration, enabling casual users to build ad hoc visualizations of Wikipedia data. Users can browse Wikipedia, select an interesting data table, then interactively discover, integrate, and visualize additional related data on-demand through a search interface and a query recommendation engine. This is accomplished through a fast path search algorithm over a semantic graph derived from Wikipedia. Vispedia also supports exporting the augmented data tables produced for use in more traditional visualization systems. We believe that these techniques begin to address the "long tail" of visualization by allowing a wider audience to visualize a broader class of data.
We evaluated this system and its interaction techniques in a first-use formative lab study. Study participants were able to quickly create effective visualizations for a diverse set of domains, performing data integration as needed. This suggests that the techniques embodied in Vispedia do reduce the time needed to author ad hoc visualizations of Wikipedia data.

Index Terms—Information visualization, Data integration, Wikipedia, Semantic web, Search interfaces.

1 Introduction

A major hurdle to the broader use of visualization is the cost of finding appropriate data sets. Since visualization methods traditionally require structured input data [4], finding data and integrating it into a single well-defined schema is an expensive and necessary preprocess before visualization can begin. In this paper we consider the challenge of designing a system that can support preliminary or ad hoc visualization in the absence of a well-formed schema.

We use Wikipedia because it contains a wealth of interesting semi-structured data for visualization, in the form of tables, infoboxes, lists, and hierarchies (categories). The data in Wikipedia is hyperlinked, forming a semantic graph. It is a representative example of the heterogeneous data sets being created by the Semantic Web community. Unfortunately, the schema of Wikipedia's semantic graph is large, messy, and unfamiliar, so while the data is richly interlinked, the cost of integration remains high.

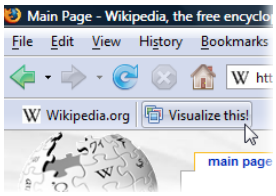
This paper presents *Vispedia*, a web-based system designed to enable exploratory visualization by supporting fast data integration on-demand. In projects such as sense.us [17] and Many Eyes [29], researchers began to consider the implications of bringing visualization to a broader audience. With the design and implementation of Vispedia, we recognize the importance of making data integration more accessible to these populations as well. We aim to allow a wide community of users to integrate and visualize data from Wikipedia, supporting a more inclusive visual exploration loop, as shown in Figure 2.

In Vispedia, users begin building a visualization by browsing Wikipedia and selecting a seed table from a Wikipedia article. After selecting a visualization type, users see a keyword query-based interface for quickly creating a visualization and integrating additional data columns into the initial table. An interactive *query recommendation system* makes use of a novel, fast graph search over Wikipedia's semantic graph, suggesting relevant data integration steps. Finally, Vispedia gathers the additional data requested by the user, augmenting the original Wikipedia table with new columns and then produces a visualization of the result. In a first-use formative evaluation study of the system (n=7), participants successfully created compelling visualizations from Wikipedia data.

In addition to permitting users to rapidly create and refine standalone visualizations, Vispedia also supports new browsing and data integration scenarios. For example, users who create a visualization of data from a Wikipedia article can then follow links in the visualization back to related articles. Vispedia also makes the user-created augmented tables available for re-use, supporting future export to other visualization applications or integration back into Wikipedia.

After presenting a scenario to illustrate the main features of Vispedia, we describe the challenges presented by Wikipedia's complex schema and explain how we designed interaction techniques, algorithms, and the larger Vispedia system to address these challenges.

2 Scenario: NHL Teams



Carol is an avid fan of the National Hockey League (NHL). While browsing the main article on the NHL[1], she notices a table listing the teams playing in the Eastern Conference (follow along in Figure 1). Wanting to explore this data in more detail, she clicks on the *Visualize this!* bookmarklet in her browser toolbar.

Doing so highlights the available data tables on the page and lets Carol pick the Eastern Conference table by clicking directly on it. Within the same Wikipedia page a menu appears allowing Carol to select an initial visualization type. This selection can be changed at any time, but seeing the original Wikipedia table in place may suggest starting points for exploration. For example, Carol notices the "Joined NHL" column, containing the date when teams first joined the league, and decides to create a timeline.

A new browser window opens showing the Vispedia site, with a query interface for specifying the visualization attributes handled by a timeline—*Date*, *Caption*, and *Image*. The data from the Wikipedia table is extracted by Vispedia and the available table columns appear in the box on the left of the screen. Carol can immediately click on a column like "Joined NHL", to map that table column directly to a visual variable, or she can integrate additional data by typing in keywords to perform a search over the rest of Wikipedia. In addition, a query recommender (displayed in a drop-down under the query field) shows a ranked list of potential integration options with example data values. The list is dynamically updated as the user changes the query.

*http://vispedia.stanford.edu

[1] http://en.wikipedia.org/wiki/NHL

• *Bryan Chan, Leslie Wu, Justin Talbot, Mike Cammarano, and Pat Hanrahan are with Stanford University, E-mail: {bryanc,lwu2,jtalbot,mcammara}@stanford.edu, hanrahan@cs.stanford.edu*

Stanford University Technical Report, April 2008

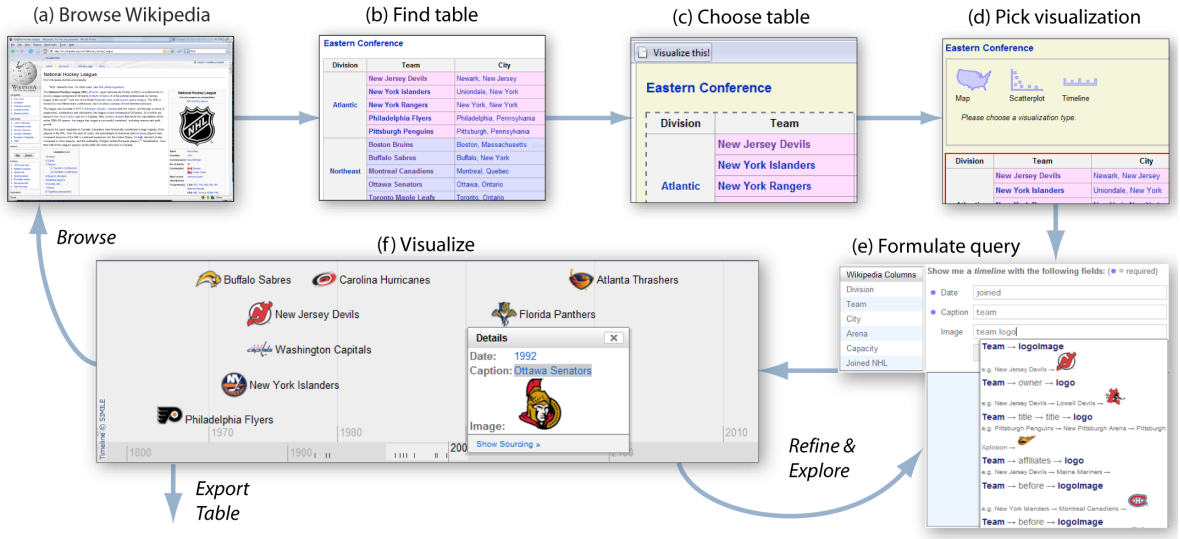


Fig. 1. Vispedia workflow: (a) While browsing Wikipedia, (b,c) a user finds a table and selects it using the Vispedia bookmarklet, (d) then picks a visualization type. On the Vispedia site, a list of table columns and the query recommender (e) help the user formulate an initial search query. Vispedia finds data matching the queries and creates a visualization (f). A user may then choose to browse back into Wikipedia, continue refining the existing query, explore related data using different visualization types, or export the augmented data table.

After creating a visualization, Carol may choose to refine it further. For example, she may modify the query in order to change the images used by the visualization or she might change the visualization type. Alternatively, Carol can click on items in the visualization or fields in the table rows (not shown) to see more details, and browse back into Wikipedia to learn more.

In minutes, Carol produced a useful visualization directly from Wikipedia.

3 Data Set and Challenges

Although in Vispedia the user interacts mainly with data in a familiar table form, the internal knowledge representation format is a semantic graph. The bulk of the semantic graph we use is extracted from Wikipedia by DBpedia [3]. In this section we will describe this data set and show how its large, unfamiliar schema makes data integration difficult for a user and for a system.

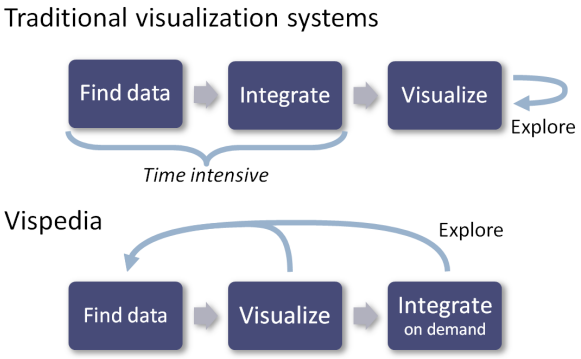


Fig. 2. (top) In existing systems, finding and integrating the data is a slow and necessary preprocess, so interactive exploratory visualization is limited to the data already integrated. (bottom) Instead, Vispedia uses a search interface to make visual exploration over large data sources with unfamiliar schema possible. Casual users can quickly find and visualize data, and then iteractively integrate new data on-demand for a particular task.

Fig. 3. Part of the Wikipedia infobox for the city of Columbus, Ohio. Infoboxes contain some of the structured information available on Wikipedia. However, since they are manually created, infoboxes can be inconsistent and incomplete. Vispedia's search algorithm, described in our previous paper [5], compensates for these problems making data integration easier.

While most of Wikipedia is hyperlinked text, it also contains structured data in the form of tables, lists, category hierarchies, infoboxes, and revision histories. For this paper we use a subset of this information, the infoboxes (e.g. Figure 3), redirections, images, and geocoordinates data in DBpedia 3.0, extracted from Wikipedia in January 2008. The redirections handle alternate names for the same object, and images and geocoordinates hold pictures and locations when they exist for an article. Additionally, Vispedia extracts structured tables directly from Wikipedia when requested by a user. Despite using a relatively well-structured subset of Wikipedia, many challenges still remain.

Vispedia's semantic graph represents semantic objects (e.g. numbers, string literals, images, and Wikipedia articles) as nodes and re-

lationships between objects as edges. For example, in the Columbus, Ohio infobox, one of the extracted edges links the city to its mayor:

$$\text{Columbus, Ohio} \xrightarrow{\text{leader name}} \text{Michael B. Coleman}$$

The edge label "leader name" is extracted from the infobox template[2] which is not visible in Figure 3. In total, this graph has 13 million nodes and 36 million edges taking up 4GB of storage.

3.1 Paths

Paths in the graph with multiple edges correspond to indirect relationships, like Kevin Bacon and Rob Reiner working on the same movie as actor and director.

$$\text{Kevin Bacon} \xleftarrow{\text{starring}} \text{A Few Good Men} \xrightarrow{\text{director}} \text{Rob Reiner}$$

To abstract the relationship a path describes from the semantic objects that are actually involved, we defined two kinds of paths in previous work: schema paths and instance paths [5]. A *schema path* contains only edge labels, and describes the abstract relationship which may occur between pairs of graph nodes. Each portion of the graph matching the relationship is an *instance path*.

Here is a schema path describing the relationship between Kevin and Rob.

$$* \xleftarrow{\text{starring}} * \xrightarrow{\text{director}} *$$

The schema path can be understood as a path template, where each $*$ can be matched to a node in the graph.

Vispedia finds data values by fixing the first node in a schema path and gathering instance paths that match the remaining path query.

$$\text{Kevin Bacon} \xleftarrow{\text{starring}} * \xrightarrow{\text{director}} *$$

When multiple instance paths match this path query, our system currently presents only the first match. We do not yet handle 1-to-many relationships.

3.2 Complexity of Schema Paths

Since schema paths correspond to types of relationships available in Wikipedia, the data integration problem becomes a problem of finding schema paths that correspond to relationships meaningful to the user. There are two reasons why doing this is difficult:

1. The average number of schema paths starting from a node is large
2. Different nodes have different schema paths available

In other words, the schema is large, non-uniform, and unfamiliar. Hence, as the length of path considered grows, the task of data integration becomes increasingly difficult for a human user to manage.

3.2.1 Large Schema

First, there are a vast number of available schema paths. Most Wikipedia articles are highly interconnected [20]. The number of schema paths starting from an article increases exponentially with path length. Here is a table showing the estimated average number of schema paths starting from an article, using a random sample of 1% of the articles with infoboxes.

Length	Average count
1	19
2	191
3	3,845
4	108,587

While we can reduce the number of paths by only considering those that contain a desired data type (e.g. a geocoordinate), the size of this subset is still substantial. Consider, for example, a few of the locations closely related to a person: birth place, alma mater, company headquarters, residence, or city they govern.

[2] http://en.wikipedia.org/wiki/Template:Infobox_Settlement

3.2.2 Inconsistent Schema

Second, Wikipedia is authored by many contributors who have only loosely standardized on different templates for different domains.

Templates define what relationships an infobox might contain. There are 2139 different templates in our data set using over 8000 different edge labels for relationships. Articles on two different people can use different templates, for instance the article on John McCain uses a senator template while Kevin Bacon has an actor template. Labels used for similar relationships can vary between templates, so while the Senator template uses $\xrightarrow{\text{birthPlace}}$, the actor template uses $\xrightarrow{\text{placeOfBirth}}$. In some articles, template fields are not completely populated, leading to missing data. Also, unlike classes in well-defined ontologies, templates specify relationships, but not the type of the objects that may appear in the range of the relationship.

Once we start following longer paths between infoboxes, these inconsistencies amplify, leading to many path variants. Consider the set of fifteen NHL teams in the Eastern Conference table in the scenario. Compare the average number of schema paths to the total number of distinct schema paths across all teams:

Length	Average count	Total distinct	Teams per path
1	37	84	6.6
2	910	3,907	3.5
3	15,366	111,525	2.1
4	404,497	3,839,530	1.6

From the average counts we see that teams are slightly more well connected than the average article counts from the previous table. Even though the teams are similar kinds of objects, they differ greatly in the kinds of available indirect relationships. On average, for schema paths of length 3 or 4, only two teams out of fifteen will have any particular schema path.

This degree of complexity in the Wikipedia schema makes it difficult to build an index summarizing available relationships in the graph. Without a manageable index, a path search algorithm can only rely on traversing the semantic graph directly. Given the number of possible schema paths, this is impossible to do manually and challenging to do automatically.

4 The Design of Vispedia

In this section we first describe the design goals and criteria that we developed while working on Vispedia. We discuss how these criteria are derived from the principal challenges that arise from using an integration-on-demand approach to visualization. We then describe the design decisions we have made to meet these criteria.

4.1 Users and Tasks

We would like our tool to be used by Wikipedia readers who would benefit from ad hoc visualizations. These users may use visualization as a means to explore Wikipedia as a whole or to understand the context around a specific Wikipedia article. Typically, they may be creating visualizations for domains outside their area of expertise, for data sets and schemas they may not have previously explored. If the cost of data integration and visualization authoring is too high, they may abandon their attempts to create an ad hoc visualization. We would also like our tool to be used by Wikipedia editors, who may want to quickly create visualizations to illustrate Wikipedia articles.

Both classes of users may be fairly nontechnical, so the interface should be straightforward, eschewing complicated data integration tools. We assume no knowledge of database technology or terminology, such as "schema integration," table joins, or SQL. It should be possible to create a visualization in a small number of steps, both as a starting point for iterative refinement, and to support the common case where the first version is good enough. More abstract data integration tools should only appear if user needs to refine the visualization further.

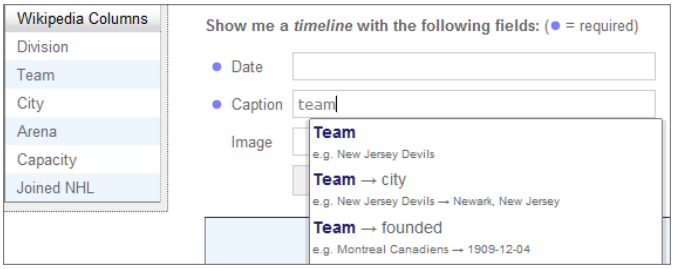


Fig. 4. Vispedia search interface showing a list of columns pulled from the user-selected Wikipedia column and the query recommender showing data items that are available through search.

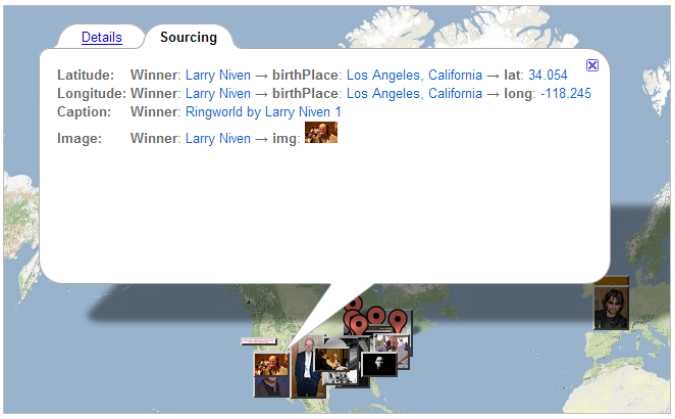
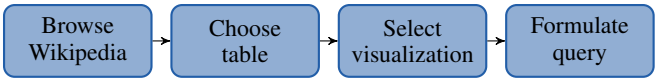


Fig. 5. Users can click items in the visualization to reveal additional sourcing metadata. This permits users to verify and evaluate how Vispedia is locating data in Wikipedia.

4.2 Designing a Search-Based Interface

As discussed previously, the space of relationships within Wikipedia is very large. To permit users to effectively find useful relationships, we propose a design that narrows the search space to be anchored to a specific, user-specified seed table. A simple query interface then hides the complexity of ranking different relationships.



We will consider our design within the user-interface framework proposed by Shneiderman, where search starts by formulating and creating a query, then reviewing the results, and refining the query [24].

4.2.1 Narrowing the Search Space

In Shneiderman's framework, query formulation is the most complex phase, involving the need to decide where to search, what terms to search for, and what variants of those terms to accept. Vispedia's design anchors the user's search to begin at a specific, known table—the one selected by the user through the *Visualize this!* bookmarklet. This design choice greatly reduces the complexity of query formulation, as it gives both the system and the user a starting point for search. The Vispedia backend can limit its exploration of the semantic graph to the neighborhood of the seed table and the user can build an intuitive understanding of the available data by exploring the Wikipedia pages linked from the table. The bookmarklet design builds upon work previously presented in d.mix [16], where a bookmarklet was also used to support *in situ* sampling of semantic data tied to web page elements.

After selecting a table, the user is presented with a choice of a visualization types supported by Vispedia. Picking a visualization type further narrows the search space since Vispedia only has to find data items that match the data types required by that visualization.

4.2.2 Formulating a Search

Even after a user selects a seed table and picks a visualization type, the space of possible relationships is large. To simplify query formulation, the Vispedia search interface helps the user in discovering possible queries.

Since the most common attributes in the schema are likely to come directly from the Wikipedia table columns, the Vispedia system displays these in a list near the query input forms, and makes it possible to directly choose keywords for a query by clicking on a column (see Figure 4). As we will explain later when discussing the search algorithm, using the column name as keywords in the search will always match data directly in the table, when it is available.

If none of the columns in the table meets the users needs, users can type in a query. Vispedia will then try to match the query keywords to relationships in the graph found by following paths.

Since the available relationships are not known to the user beforehand, we provide a query recommendation engine similar in nature to an auto-complete widget (also shown in Figure 4). This widget shows a ranked list of paths in the graph which most closely match the current query string. This interactive feedback makes it possible to quickly see if the search will match useful results and helps suggest additional terms to formulate or refine the query.

As is the case with Web search engines, Vispedia offers small snippets of text with each suggested path. These snippets describe the suggested path through an example data instance that results from following that path. For example, consider the query recommender interface displayed in Figure 4. In this case, the query recommender widget is displaying three possible data integration paths related to the query term "team." These paths are displayed in a sorted order, where the first path is estimated to be the most relevant. The second matched path is shown as "Team → city", and an example data instance retrieved by following this path is also shown, as *"New Jersey Devils → Newark, New Jersey"*. The user can select an item directly from the query recommender. This populates the query field with the terms from the path, guaranteeing that the chosen path will then be used first.

4.2.3 Reviewing Search Results and Refining Query

The next phase in Shneiderman's framework is reviewing the results. After the user fills in desired fields of the visualization template, they can click "Show Visualization" which causes Vispedia to query the Wikipedia semantic graph. Returned data are shown both on the visualization and in tabular form. Showing both representations makes incorrect or missing results easier to see.

The semantic paths used to find the data are available by clicking on columns in the table or by clicking on items in the visualization (see Figure 5). These paths provide links back to Wikipedia which users can use to better understand how Vispedia is finding the results. This allows the user to evaluate the provenance of the data.

Given this information the user can revise their query to improve the results.

5 Implementation

The bookmarklet used for table selection is implemented in Javascript. The web frontend is implemented in PHP, and the visualization interface uses Javascript components from MIT's Simile timeline [22], Google Maps [14], and dōjō charting [7].

Vispedia combines relationships extracted from a Wikipedia table and those extracted by DBpedia, and then the path search algorithm runs over this combined graph. The performance of the current graph search is a significant improvement over the algorithms used in last year's paper.

5.1 Linking Tables and Graphs

At the moment, Vispedia combines the current version of a Wikipedia table with a semantic graph that DBpedia extracted from the January

2008 Wikipedia data dumps.

The system takes the user-selected table, fetches the latest version of the table from Wikipedia, parses the raw wikicode, and converts it to a graph. This graph contains one node for each row in the table, with labeled edges to each field in the row. For example, in the NHL scenario table, Figure 1, the following edge is extracted from the Team column of the first row:

$$\text{row:0} \xrightarrow{\text{Team}} \text{New Jersey Devils}$$

When the DBpedia graph and a table graph reference the same Wikipedia article through a hyperlink, like the New Jersey Devils, or use the same string literal value, then the same node appears in both graphs. Our system is able to follow paths from a table row, through the shared node, and into Wikipedia to find additional data.

Using the latest table version means that a user will have exactly the same data they see on Wikipedia. The other data depends on the frequency of Wikipedia data dumps, which occur once every few months. While tracking all recent changes on Wikipedia is possible, to simplify the implementation, we do not attempt to do this.

5.2 Matching Keywords to Paths using Graph Search

The system answers a user's search queries by finding a ranked list of schema paths that best match the user's query keywords.

While there are database solutions for storing and querying semantic graphs, they are not designed for the sort of path search operations we depend on. Instead, we keep all the graph edges in a memory mapped adjacency list, with a Berkeley DB for storing strings from node and edge labels.

In our previous paper, we used a naïve, exhaustive graph search which took several minutes to search for paths to a depth of 4. In Vispedia, the graph search is designed instead to support the response times needed for interactively populating the query recommender and finding data for the visualization. We use an A* search that returns the top schema paths found within a given time limit.

5.2.1 A* search

The intuition for using an A* search is that keywords in a path that do not appear in the keyword query give a poorer match. This makes it possible to direct the graph search towards more promising regions and to prune many irrelevant paths early.

While in the previous paper, we depended on document retrieval metrics designed to find documents that contain keywords, these metrics do not account for extra words in the path "document" that are not in the keywords. Because of this, we used extra heuristics that accounted for the path length.

Instead, in Vispedia, we use a simple metric for measuring document similarity: a distance between the term frequency vectors of the keyword query and the path [23]. This accounts for unmatched words in the path. More complex similarity measure are available, but we opted for a simple to implement metric that can be incrementally updated as a search proceeds along a path.

5.2.2 Distance Function

The raw term frequency vectors count how often words appear in the path and query "documents," where words are extracted from strings by considering non-alphanumeric separators and camelCasing. For example, the term frequency vector for the keyword query "team" is the following:

$$\text{tfv}(\text{"}team\text{"}) = \{t_1, t_2, ...t_n\}$$

where each t_i represents the count of a different word appearing in some edge label in the entire semantic graph. Each $t_i = 0$ in this case, except for the one that represents the word "team."

The weighted Manhattan distance between the query, with vector $\text{tfv}_{\text{query}} = \{q_1, q_2, ...q_n\}$ and the path $\text{tfv}_{\text{path}} = \{p_1, p_2, ..., p_n\}$ is

$$\text{dist}(\text{tfv}_{\text{query}}, \text{tfv}_{\text{path}}) = \sum_{i=1}^{n} \text{tfidf}(i)\text{w}(p_i, q_i)|p_i - q_i|$$

$$\text{where w}(p_i, q_i) = \begin{cases} \alpha & \text{if } p_i > q_i \\ 1 & \text{if } q_i \geq p_i \end{cases}$$

Terms in the distance function are weighted by TF-IDF, to give common terms like "of" and "the" less impact on the distance. There is also a parameter α, which makes unmatched terms in the path less important than unmatched terms in the query.

Using this distance metric, we can find a lower bound on the distance possible by extending the path. This gives us a way to prune paths early during the A* search. This lower bound is

$$\text{mindist}(\text{tfv}_{\text{query}}, \text{tfv}_{\text{path}}) = \sum_{i=1}^{n} \begin{cases} \text{tfidf}(i)\alpha(p_i - q_i) & \text{if } p_i > q_i \\ 0 & \text{otherwise} \end{cases}$$

For example, if the query is "team" and the path is "name of team" then the dist and mindist are both 2, since the word "team" matches but there are two unmatched words in the path. At this point, adding more terms to the path can only increase the distance.

Currently α is set to 0.5, but we have not tested this extensively.

5.2.3 Time Limits and Heuristics

We search for possible schema paths using the A* search, one row at a time, until all rows finish or the search takes 1 second. Any rows not explored by the time limit will attempt to use schema paths found previously when searching for instance paths. The rows are considered in random order, to prevent bias in the search.

One drawback of searching one row at a time is that if a row has no matches to the keywords, it can monopolize the available time. To avoid this limitation in our system, we also restrict the search to 0.1 seconds per row

To improve recall, we also search for the top 5 paths for each row that is fully explored. This gives the user a wider range of choices to consider in the path recommender.

Vispedia also keeps several useful heuristics from our previous system. It ignores paths with high branching factor and only follows paths through string literals, not numbers. As discussed in our previous paper, paths failing these criteria are less desirable.

5.2.4 Performance

We will evaluate the algorithm based on the number of schema paths considered by the search and the execution time. Evaluation of the result quality was done as part of the user study.

Using queries from the NHL timeline scenario (Section 2), the search prunes most of the longer schema paths with too many irrelevant words, so it avoids the exponential growth in the number of schema paths. The table below shows the average number of schema paths explored at each length over all rows of the NHL table, with each column showing a different query. Note that the first step of any schema path always passes through one of the 7 columns in the Wikipedia table.

Length	"Joined NHL" (date)	"Team" (string)	"Team logo" (image)
1	7	7	7
2	200	186	204
3	2314	268	1407
4	3274	255	1499
5	136	2	390

The search for "Joined NHL" matches the table column, but then finds few other paths matching the rare keywords. As a result, the search to find the top 5 paths continues with few opportunities to prune early.

Since the NHL table is relatively small, all the items are fully explored well under the time limit of one second. The next table shows the total execution time of the A* search over all rows of the table. For comparison, we also show the times of a depth first search of the entire space of schema paths up to length 4 and length 5.

	Time(s)
"Joined NHL" (date)	0.34
"Team" (string)	0.026
"Team logo" (image)	0.13
DFS (to length 4)	3.5
DFS (to length 5)	96

Over two weeks of usage that included the user study, we logged a larger sample of 526 unique queries on 37 different tables from our website. We instrumented our system after the study and re-executed the logged queries.

Most of the queries finished within the time limit (457 queries). The remaining 157 queries that reached the 1 second barrier tended to be from larger tables (more than 120 rows on average), and they still managed to fully explore 38% of their rows on average. Partial exploration primarily affects recall on the least frequent schema paths.

6 Evaluation

We conducted a first-use evaluation study with 7 total participants: 2 were female, 5 were male. Their ages ranged from 18-24 (five participants) to 25-34 (two). Most participants (over two-thirds) used Wikipedia on a daily basis, and had no experience with database technologies such as SQL or Microsoft Access.

The first two participants took part in a pilot study, designed to uncover major usability flaws and provide incremental feedback on Vispedia's design.

6.1 Study Protocol

Each study session took approximately 60 minutes. Participants were seated at a workstation in our research lab, and used a standard web browser that had the "Visualize this" bookmarklet pre-installed. We demonstrated Vispedia's interface and briefly walked through the process of specifying and refining search terms.

We asked participants to complete three design tasks of increasing complexity. The first task tested the basic usability of our system, and instructed participants to create a timeline of NHL hockey teams and their logos, according to when teams joined the NHL. All participants successfully completed this basic data integration and visualization task, although completion times varied, from approximately five to twenty minutes.

According to Wikipedia, "a fourteener is a mountain that exceeds 14,000 feet (4,267.2 m) above mean sea level." Participants were asked to plot all fourteener mountain peaks in California on a map, and use this depiction to plan a mountain climbing trip, beginning with the southern-most mountain.

As with task one, completion times for task two varied, but not all participants successfully found the southern-most fourteener. Besides the difficulty involved in finding the appropriate data—participants as a group attempted to use three or four distinct tables for this task, some of which were not sufficient to complete the task given—at least three participants confused latitudes with longitudes.

Finally, the third design task was open-ended, asking users to pick a topic of personal interest, and to create a compelling visualization from a table found on Wikipedia. We provided several suggested topics, but participants generally diverged from this list, preferring to explore, for example, topics ranging from the Roman Empire to aeronautic disasters of the twentieth century. Participants employed various strategies for finding data, but all found suitable tables to send to Vispedia in three to ten minutes. The table sizes varied from five rows (Roman Emperors from the Julio-Claudian dynasty) to 577 rows (United States Micropolitan Statistical Areas).

6.2 Successes

Overall, results from evaluation study support our hypothesis that the Vispedia system and associated interaction techniques were easily learnable (mean=4.1, median=4 on a 5-point Likert scale, σ=0.7). In combination with a novel, fast graph-structured search algorithm, the resulting system was found to be usable by first-time users (mean=4.3, σ=0.8), who were able to browse or search for tabular data sets and quickly prototype visualizations.

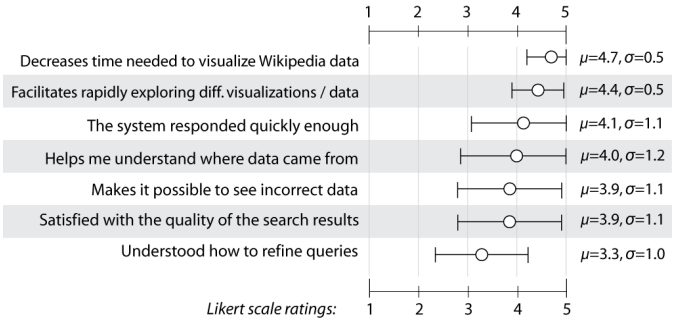


Fig. 6. Post-experiment questionnaire results. Error bars indicate one standard deviation in each direction.

6.2.1 The cost structure of visual sensemaking

We hypothesized that the Vispedia system would "decrease time needed to visualize Wikipedia data" and the post-test questionnaire results strongly supported this claim (mean=4.7, σ=0.5). One participant wrote that Vispedia could help in "sparing the user from the tedium of manual spidering" when one is interested in information not presented in the original Wikipedia table.

Participants generally agreed that the Vispedia system responded quickly enough (mean=4.1, σ=1.1). Furthermore, they strongly agreed that Vispedia "facilitates rapidly exploring different visualizations and data" (mean=4.4, σ=0.5).

6.2.2 Data integration on demand

Before running the lab study, we hypothesized that Vispedia would support data integration and visualization on demand, by supporting a fast, iterative, and visual sensemaking loop. Observing our participants as they completed the provided design tasks, we noted that participants would enter a few query terms for a subset of fields, display the corresponding visualization, and then return to refine their query terms, much as one would refine search terms for a keyword search on the Web.

Participants generally agreed that they were satisfied with the quality of the search results returned (mean=3.9, σ=1.1), and were able to perform data integration and visualization for tables the authors had not previously encountered. One participant said they thought it was "impressive that the system did the smart thing in most cases."

6.2.3 Data quality and provenance

While integrating data from multiple sources, participants often used the visual representations of data to heuristically check data quality. For example, when plotting data specific to one country or state, participants easily noticed visual outliers. In contrast, participants performing the California Fourteeners task often failed to recognize numeric outliers if they inspected data mainly in the table form.

Participants mostly agreed that the Vispedia "makes it possible to see incorrect data" (mean=3.9, σ=1.1) and also "helps [them] understand where data comes from" (mean=4, σ=1.2). We observed that participants made frequent use of the ability to quickly toggle between "data sourcing" view and data view in the table provided, as a way to quickly get a sense of where and how different items are being sourced.

6.2.4 Compelling visualizations

We believe that participants were able to create compelling visualizations of their own choosing, and several participants wanted to e-mail the visualizations they created to friends. One participant thought Vispedia was "an easy way to share graphs... can argue points in blogs visually." Examples of user-generated visualizations are included after the references.

6.3 Shortcomings

6.3.1 Learning the query model

While the majority of participants reported that they "understood how to refine queries," the other participants either disagreed or were neutral about this claim. All of the participants were able to refine queries and author visualizations, so we infer that participants may have struggled to align their mental model of Wikipedia's semantic graph (infoboxes, tables, and entries) with system's understanding of Wikipedia. For example, a NHL logo image may be labeled "logo_image" in the infobox template, but neither "logo" nor "image" may appear in the rendered Wikipedia article itself.

To address this mismatch, a future system may provide a way to colocate semantic metadata with the original web page. In particular, the "Visualize" bookmarklet could also allow users to mouse over wiki elements which have invisible semantic representations, surfacing them visibly.

6.3.2 Transforming and Tailoring

Similarly, participants were split as to whether Vispedia allowed them to "specify data [they] want to show" on the visualization (mean=3.3, σ=0.8). At times, participants wanted to plot "unrelated" images or iconic representations such as triangles or arbitrary pictures from Wikipedia. One participant asked for the ability to edit the table sourced from Wikipedia directly. These suggestions would improve users' ability to transform and re-represent sourced data. They also suggest the tailoring and editing of existing visualizations as one avenue for future work.

6.3.3 Change blindness

Because data integration happened on demand, participants would occasionally exhibit change blindness when refining queries. That is, they were unable to tell whether or not changing query terms or paths actually resulted in a different integrated data set. Participants also failed to notice missing data when only one or two cells were incomplete. More work needs to be done in understanding how visualization systems can help to highlight missing data and combat change blindness.

6.3.4 Sharing and Collaboration

Multiple participants requested the ability to share and annotate their visualizations. Techniques described in Heer's sense.us [17] and ManyEyes could be extended to a system that closes the loop by allowing users to embed created visualizations or tables back into the original data source (in this case Wikipedia).

7 Related Work

Vispedia builds on the research in two areas: web-scale collaborative data exploration, and semantic graphs.

7.1 Collaborative Data Exploration

Collaborative visualization projects such as sense.us [17] and Many Eyes [29], as well as commercial endeavors like Swivel [27] and Tableau Server [28] have made it easy to explore and share well structured data with other users. GraphWise [15] is another example that combines user-created content with visualizations automatically generated from Wikipedia tables. These systems depend on structured data with a known schema, so Vispedia complements these advances in social visualization by addressing the case where the schema is large and unknown.

Visualizations that make transparent the social dynamics of Wikipedia authors, like WikiDashboard and the revert graph [1, 26] help users better understand the source of the data itself.

The challenge of performing data integration at web-scale has been explored by others. The Cimple [6] project created a web platform for supporting the social aspects of community-driven data integration. Madhavan et al. promote a formal "Pay as You Go" [21] model where web-scale data integration is done on demand and implicit or explicit user feedback is incorporated to improve the results. Like our project, they use queries and ranking to enable integration. Google Base [13] and FreeBase [10] are two commercial ventures whose goal is to create large semantic databases through collaborative contributions from users. In contrast to these systems, Vispedia emphasizes visualization as the major focus.

7.2 Semantic Graphs

A second rich research area is the work on how to extract, search, and visualize semantic graph data.

There has been growing interest in extracting semantic relationships from Wikipedia. DBpedia provides an extractor for the infoboxes [3] and links the graph they extract with a wider Linking Open Data community project [2]. The KYLIN system uses the relationships in infoboxes to seed a relationship extractor on the free text of Wikipedia [33]. The YAGO system seeks to build a better category hierarchy, a first step to more meaningful classes of objects on Wikipedia, by combining the Wikipedia category hierarchy with the Wordnet hierarchy [25]. Dontcheva et al. [9] have suggested a personal data extraction and integration system based on templatized screen scraping. Making more organized, interlinked semantic data available increases the benefits of building visualizations from such data.

Many query interfaces and search algorithms exist for semantic graphs. There are graphical query interfaces for RDF such as the W3C's Tabulator [31], and the basic DBPedia search interface [3], use a link following model similar to web browsing. The W3C SPARQL query language [30] provides a way to specify graph patterns against a semantic graph.

Compared to the stepwise query interfaces found in all of these systems, we support a more flexible keyword query for discovering and matching paths, more akin to the proximity search system in Lore [11], which ranks objects based on their distance in the graph to focus objects. The DBpedia Relationship Finder [20] allows users to browse paths connecting user-specified pairs of nodes.

The graph search algorithm we present is similar in spirit to the Bidirectional keyword search by Kacholia et. al. [19]. They developed a method for matching keywords to subtrees that searches the graph as directed by a scoring function. Indexing schemes for keyword search on heterogeneous graphs [8] and for summarizing relationships in DataGuides [12] also exist; however, it is unproven whether these approaches can handle indirect path searches on a schema as complex as Wikipedia's.

Exhibit is a tag based method of specifying a visualization from an underlying RDF graph [18]. Its associated MediaWiki extension, Wibbit, allows specifying visualizations directly using Wikipedia's wikicode [32]. Vispedia aims to support a wider community of users by addressing the issue of data integration.

Our query recommendation system and our visualizations that link back into Wikipedia provide an alternative way to understand and navigate semantic graphs.

8 Conclusions and Future Work

We introduced a novel system, Vispedia, for building visualizations sourced from Wikipedia, and described its algorithmic and interaction design. Vispedia addresses the long tail of visualization, by reducing the work required to integrate and visualize data.

In a first-use lab study with 7 participants, we found that participants did indeed perform data integration and visualization on demand, switching back and forth between visual and tabular representations. The study also demonstrated the usability of Vispedia and the interaction techniques it embodies, as well as the feasibility of an interactive visualization authoring tool, enabled by a fast, novel graph-structured search algorithm. We hope that future visualization researchers consider the challenges involved with data sets such as Wikipedia, and continue to work toward a tighter sensemaking loop, which seamlessly blends foraging for data, data integration, visual and non-visual sensemaking, re-representation, and storytelling in a compelling fashion.

Acknowledgements

This work was supported in part by an RVAC grant from the Department of Energy and the Max Planck Institute. Thanks to Björn Hartmann for figure assistance and inspiration, and to Mark James for the use of his icons.

References

[1] Lifting the veil: Improving accountability and social transparency in wikipedia with wikidashboard.

[2] http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData, 2008.

[3] S. Auer, C. Bizer, J. Lehmann, G. Kobilarov, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *Proceedings of ISWC 2007 (To Appear)*, 2007.

[4] J. Bertin. *The Semiology of Graphics*. Univ. of Wisconsin Press, 1984.

[5] M. Cammarano, X. L. Dong, B. Chan, J. Klingner, J. Talbot, A. Halevey, and P. Hanrahan. Visualization of heterogeneous data. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1200–1207, 2007.

[6] A. Doan, R. Ramakrishnan, F. C. 0002, P. DeRose, Y. Lee, R. McCann, M. Sayyadian, and W. Shen. Community information management. *IEEE Data Eng. Bull.*, 29(1):64–72, 2006.

[7] dōjō javascript toolkit. http://dojotoolkit.org/.

[8] X. Dong and A. Y. Halevy. Indexing dataspaces. In *SIGMOD Conference*, pages 43–54, 2007.

[9] M. Dontcheva, S. M. Drucker, D. Salesin, and M. F. Cohen. Relations, cards, and search templates: user-guided web data integration and layout. In *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 61–70, New York, NY, USA, 2007. ACM.

[10] freebase. http://www.freebase.com.

[11] R. Goldman, N. Shivakumar, S. Venkatasubramanian, and H. Garcia-Molina. Proximity search in databases. In *Proc. of VLDB*, 1998.

[12] R. Goldman and J. Widom. Dataguides: Enabling query formulation and optimization in semistructured databases. In *Proc. of VLDB*, Athens, Greece, 1997.

[13] Google Base. http://base.google.com/, 2005.

[14] Google Maps. http://maps.google.com.

[15] Graphwise. http://www.graphwise.com.

[16] B. Hartmann, L. Wu, K. Collins, and S. R. Klemmer. Programming by a sample: rapidly creating web applications with d.mix. In *UIST*, pages 241–250, 2007.

[17] J. Heer, F. B. Viégas, and M. Wattenberg. Voyagers and voyeurs: supporting asynchronous collaborative information visualization. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1029–1038, New York, NY, USA, 2007. ACM.

[18] D. F. Huynh, D. R. Karger, and R. C. Miller. Exhibit: lightweight structured data publishing. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 737–746, New York, NY, USA, 2007. ACM.

[19] V. Kacholia, S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, and H. Karambelkar. Bidirectional expansion for keyword search on graph databases. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 505–516. VLDB Endowment, 2005.

[20] J. Lehmann, J. Schppel, and S. Auer. Discovering unknown connections - the dbpedia relationship finder. In S. Auer, C. Bizer, C. Mller, and A. V. Zhdanova, editors, *CSSW*, volume 113 of *LNI*, pages 99–110. GI, 2007.

[21] J. Madhavan, S. Cohen, X. L. Dong, A. Y. Halevy, S. R. Jeffery, D. Ko, and C. Yu. Web-scale data integration: You can afford to pay as you go. In *CIDR*, pages 342–350, 2007.

[22] MIT. Simile timeline. http://simile.mit.edu/timeline/.

[23] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.

[24] B. Shneiderman, D. Byrd, and W. B. Croft. Clarifying search: A user-interface framework for text searches. *D-Lib Magazine*, 1997.

[25] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 697–706, New York, NY, USA, 2007. ACM.

[26] B. Suh, E. H. Chi, B. A. Pendleton, and A. Kittur. Us vs. them: Understanding social dynamics in wikipedia with revert graph visualizations. *Visual Analytics Science and Technology, 2007. VAST 2007. IEEE Symposium on*, pages 163–170, Oct. 30 2007-Nov. 1 2007.

[27] Swivel. http://www.swivel.com.

[28] Tableau. http://tableausoftware.com.

[29] F. Viegas, M. Wattenberg, F. van Ham, J. Kriss, and M. McKeon. Manyeyes: a site for visualization at internet scale. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1121–1128, Nov.-Dec. 2007.

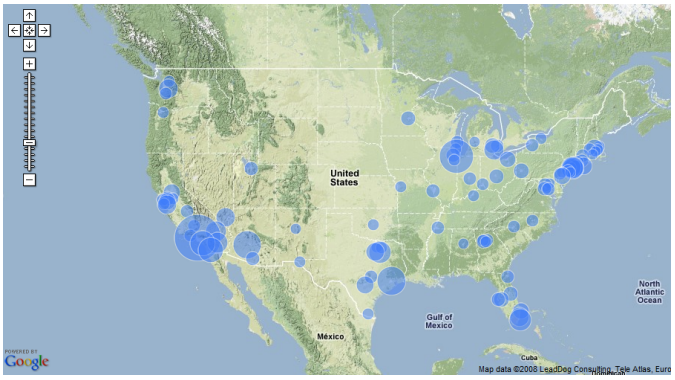
[30] W3C. SPARQL Query Language for RDF. http://www.w3.org/TR/rdf-sparql-query/.

[31] W3C. The Tabulator. http://www.w3.org/2005/ajar/tab.

[32] Wibbit. http://simile.mit.edu/wiki/Wibbit.

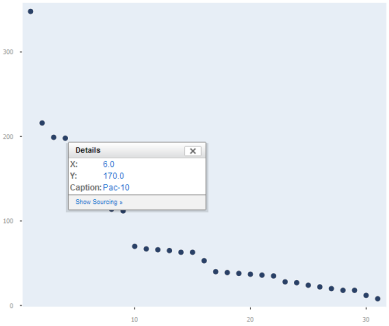
[33] F. Wu and D. S. Weld. Autonomously semantifying wikipedia. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 41–50, New York, NY, USA, 2007. ACM.

A Select User-generated Visualizations



Largest counties in the United States.

Birth place and image of Hugo Award winners (red marker indicates missing image).



NCAA Men's Basketball Tournament bids by conference.