



HiveRel: hexagons visualization for relationship-based knowledge acquisition

Sivan Yogev¹ · Guy Shani² · Noam Tractinsky²

Received: 7 October 2021 / Accepted: 12 February 2022 / Published online: 11 April 2022
© China Computer Federation (CCF) 2022

Abstract

The growing abundance in complex network data models is constantly increasing the challenges for non-expert users who perform an effective exploratory search in large data collections. In such domains, users search for entities related to a topic of interest and acquire knowledge by investigating the relationships between these entities. Designers, in turn, are challenged by the need to provide tools that enable convenient search and exploration to facilitate productive performance on the task. For this purpose, we introduce *HiveRel*, a search system that presents search results as tiled hexagons on a map-like surface with center-out relevance ordering and allows on-demand display of relationships between search results. *HiveRel*'s user interface is based on theoretical principles that reflect how users acquire knowledge through relationships. For the search mechanism, we provide a set of information retrieval definitions leading to the formalization of the Maximal n -Bounded Exploration Subgraph problem and present an implementation of a greedy heuristic algorithm that provides non-optimal solutions to this problem. We develop a proof of concept version of *HiveRel*. We evaluate it in two user studies that compare users' performance using *HiveRel* to standard web search over a range of search knowledge acquisition tasks and two different domains. The results indicate that despite the lack of familiarity with the new system, users were generally more accurate and as fast using *HiveRel*, and provided positive evaluations for the search experience.

Keywords Exploratory search · knowledge acquisition · information retrieval · search interface design · search algorithm · user interface · user study

1 Introduction

Consider the situation where a researcher wishes to explore certain research direction with little a-priori knowledge on related existing work. For either established researchers or students, the ability to map the body of knowledge surrounding a certain concept is an important part of the profession.

The basic requirement towards accomplishing such task is to identify the most important articles published on the subject, as well as the most influential authors. The importance of each entity (either article or author) is related to the entity's place in the network that connects entities through authorship and citation relationships. Therefore, given a textual query, the researcher should be able to view related entities and the relationships between them in a timely manner. As the network could be complex, the initial view of the data should be wide, with an option to focus on subsets of the results.

In recent years, new networks with similar characteristics to the scientific articles network appeared, such as social applications (Facebook, Twitter), biomedical data (Medline), customer relationship management data and others. The data in these networks is composed of entities with textual attributes, which participate in relationships of various types, and users of these systems often experience the need to learn new things about a concept of interest.

Parts of this paper appeared in Yogev et al. (2017).

✉ Guy Shani
shanigu@bgu.ac.il
Sivan Yogev
sivan.yogev@gmail.com
Noam Tractinsky
noamt@bgu.ac.il

¹ Outbrain Inc, Netanya, Israel

² Software and Information Systems Engineering, Ben Gurion University of the Negev, Beersheba, Israel

The aforementioned requirements correspond with Marchionini's definition of *Exploratory Search* made in 2006 Marchionini (2006), where search activities are classified into three partially overlapping kinds of lookup, learning and investigation, with *Exploratory Search* defined as covering all non-lookup activities. Although attempts were made to meet the challenge of providing a good exploratory search experience, no significant break-through has yet been achieved. As a result, the important need for effective exploratory search systems still exists.

In light of all the above, **the goal of this work is to allow non-expert users to perform effective exploratory search** over large data collections in domains containing complex data models. One domain with complex data model that is of interest to the wider public is movie-related information, from which we use the following scenario as a running example throughout this paper: consider a movie fan who, after watching a movie directed by Rob Reiner, wishes to learn which actors played in other films directed by Reiner. Such a task requires obtaining the list of movies directed by Reiner, extracting the cast of each movie, merging the casts and counting the number of total occurrences for each actor. The challenge presented by this information need is that it is not about Rob Reiner or a certain fact related to him. Rather, it aims at investigating an information space with Rob Reiner at its center, gaining knowledge from the relationships of objects in this space. Clearly, this is not a "lookup" search, but rather a "learning" one, and more specifically a "knowledge acquisition" task Marchionini (2006). We therefore propose a system geared towards effective knowledge acquisition through search.

There are currently two popular alternatives for performing such tasks. The straightforward approach is to issue multiple queries to a web search engine while recording some information in another application (such as a spreadsheet). The other method is to use complex data management systems with a query language such as SQL Date and Darwen (1989) or SPARQL Prud et al. (2006). The former usually requires extensive manual labor, while the latter requires uncommon expertise. Perhaps the most important feature that these two alternatives lack is the ability to easily retrieve and present relationships between search results of various types. In the Rob Reiner example, relevant relationships may include those connecting actors, directors, and movies.

Indeed, there are many datasets that contain structured or semi-structured information concerning different types of entities. In the movie domain, for example, the entities are movies, genres, actors, directors, and shooting locations; In the academic publications domain, the entities are articles, authors, venues, research areas, keywords, and so forth. These datasets further contain different types of relationships between entities. For example, in the movies domain a movie is *directed by* a director, and may *belong to* various

genres. In the academic domain a paper is *authored by* a number of researchers, and *published* at some conference or journal, and is *described by* some keywords.

A prominent barrier for presenting relationships between search results over such datasets is the currently popular method in which search results are presented in a unidimensional bottom-down ranked list (often called the "ten blue links" Russell-Rose and Tate (2012)). We propose an alternative approach, using a two dimensional layout, where the query is placed at the center surrounded by search results located on the plane in a center-out (or "Radial" Munzner (2014)) relevance order (see Fig. 1). The most relevant results are placed near the center while less relevant results are placed farther, with visual cues implying the relevance level.

In the search task of finding actors related to Rob Reiner, notice that the actors sought are not directly relevant to the query "Rob Reiner", but rather to the query's results. The new visualization approach enables the presentation of relationships as a floating layer above the search results plane. To avoid visual clutter we propose an interaction mechanism which enables on demand dynamic presentation of relationships (see Fig. 2). To answer this class of information needs we define and implement a new type of information retrieval system that, given a textual query, (1) provides the most relevant results, (2) identifies results that are most related to the relevant ones, and (3) captures the relationships between the results.

One of the major differences between the proposed search system and existing solutions is a new approach for search results presentation. In most search systems, the query is presented at the top with a list of ranked result below in decreasing relevance order (less relevant result positioned in lower location). This layout best serves lookup activities Munzner (2014), but is less appropriate for exploratory search. In the proposed new approach, the query is positioned at the center of the screen, with search results surrounding it as tiles. Search results are arranged by relevance order from the center outside, so that the most relevant results are close to the query, receiving initial attention from the user.

The layout design is based on cognitive engineering and visualization principles to facilitate the efficient discovery of information (e.g., Wickens et al. (2021); Ware (2019)). It focuses the user's attention on the center of the display. This layout's structure minimizes eye movements Ware (2010), given that the more relevant search results are concentrated around the center of the display, and movement to the periphery (for lower relevance entities) occurs with lower probability. The layout also supports a continuous overview of the results, retaining focus and context at the same time. This simplifies the exploration of the results for patterns, dependencies, and relations in the data while

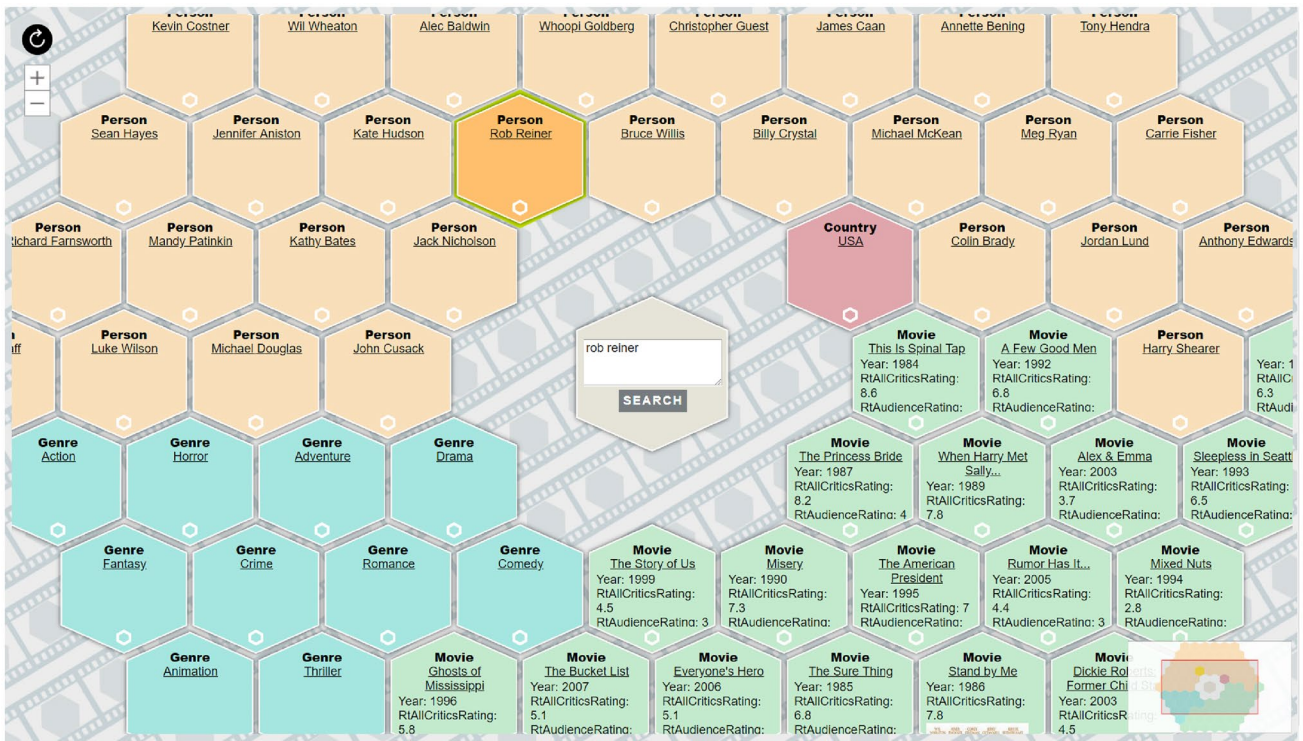


Fig. 1 The initial results view for the query “Rob Reiner” in the *MovieLens* data set

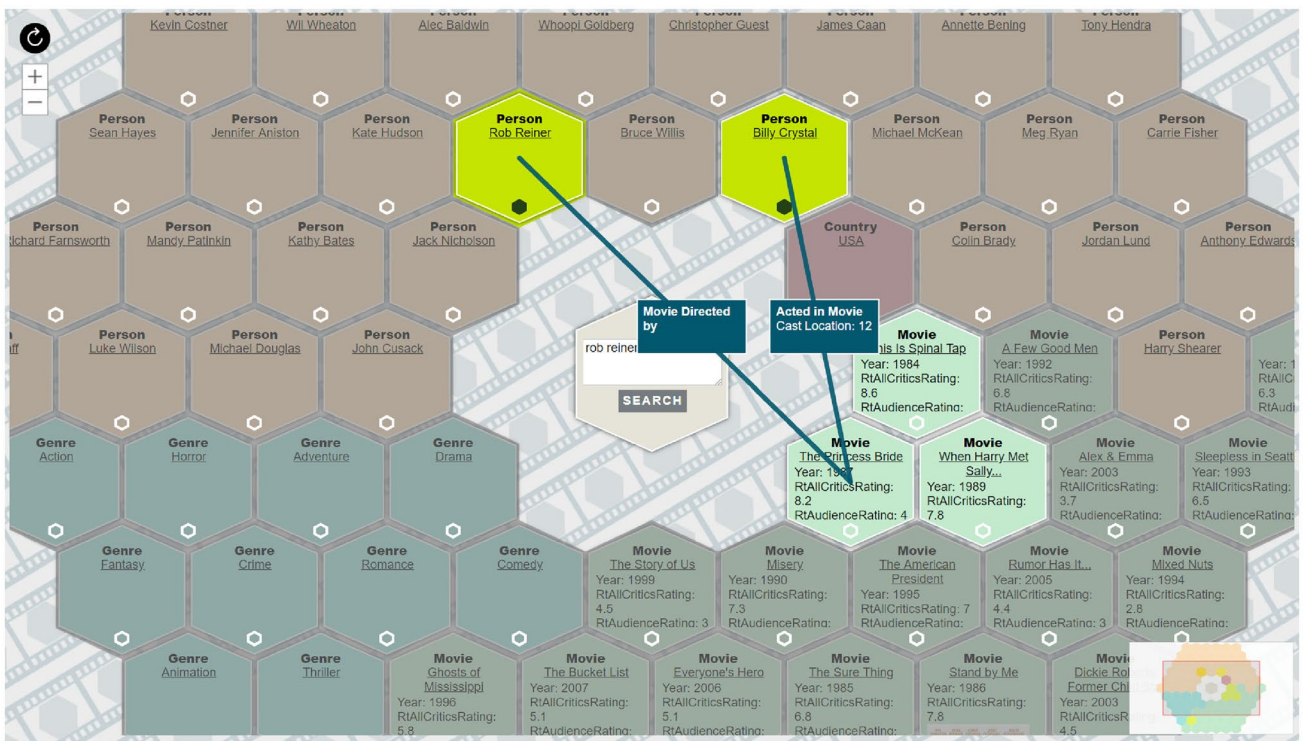
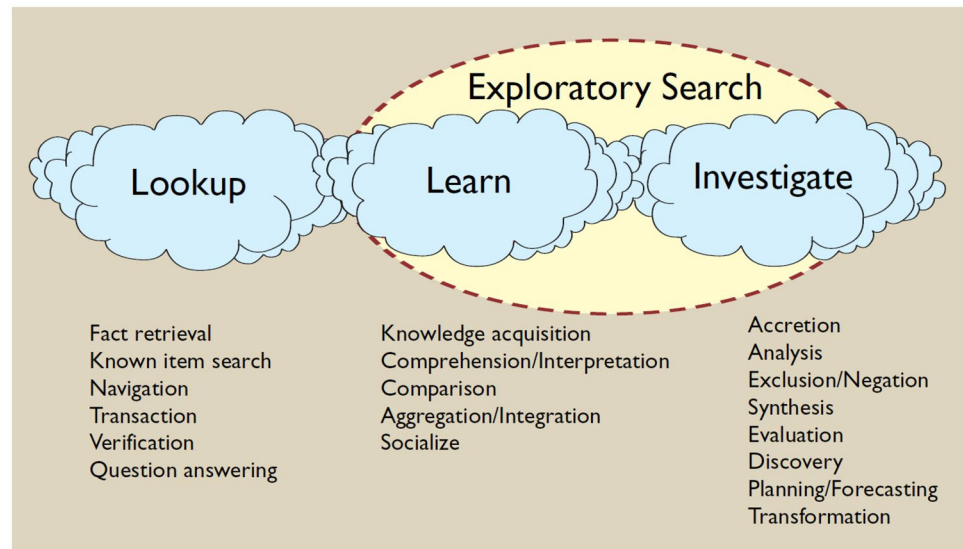


Fig. 2 Screenshot of results for the query “rob reiner” in the *MovieLens* data set, with results representing Rob Reiner and Billy Crystal selected as indicated by their yellow background color. There are four movies related to both people, and only these movies are unshaded

(one movie visible only in the overview panel at the bottom right of the screen). The cursor hovers over the movie “The Princess Bride”, revealing the relationships between the movie and the two selected persons

Fig. 3 Figure taken from Marchionini Marchionini (2006), defining three kinds of search activities with sub-activities detailed. Exploratory Search covers “Learn” and “investigate” activities



maintaining contextual orientation Shneiderman (2003); Munzner (2014). Finally, HiveRel’s interface provides consistent context, entity type, and relevance cues by using Gestalt principles such as proximity, similarity, and common region, which are operationalized in terms of distance from the center, coloring, and grouping of entities Ware (2019).

Based on this new information retrieval system, we implement a proof-of-concept live system which we call *HiveRel*, where in response to a user query, search results are presented as hexagons tiled over the plane, with relationships presented on-demand. Our system is currently applicable only to structured databases that explicitly maintain entities, types and relationships.

To test the effectiveness of the new system we conduct two user studies, comparing the execution of knowledge acquisition tasks in *HiveRel* with standard web search using the most popular search engine — Google. The user studies demonstrate that even with a very short learning period, *HiveRel* is a viable alternative for performing complex knowledge acquisition tasks.

The rest of this paper is organized as follows: in Sect. 2 we survey the context of this work in relation to previous research, and propose a new classification space for assessing the level of support search methods provide for knowledge acquisition through search. In Sect. 3 we provide description of the proof-of-concept *HiveRel* search system, starting with the proposed graphical user interface and the reasoning behind its design decisions in Sect. 3.1, followed by selected implementation details given in Sect. 3.2, information retrieval definitions in Section 3.3, and finally the novel search algorithm in Sect. 3.4. System evaluation through two user studies is described in Sect. 4. Section 5 contains discussion of the research results, and portrayal of the major future challenges that stem from this work, and

Sect. 6 concludes the contributions of this work to the relevant research fields.

2 Background

In this section we discuss previous research in five different relevant aspects: exploratory search, exploratory search interfaces, knowledge acquisition and information visualization and knowledge acquisition through search. We propose a classification space for approaches aiming to support the latter aspect, placing current approaches as well as our proposal in this space.

2.1 Exploratory search

In 2006, Marchionini Marchionini (2006) classified search activities into the three partially overlapping kinds of lookup, learning and investigation, and termed non-lookup activities as “Exploratory Search”. Figure 3 presents more details on these observations.

In recent years, attempts were made to meet the challenge of providing a good exploratory search experience, with no significant break-through achieved. As a result, the need for effective exploratory search systems still exists.

Two developments make this need increasingly more acute. The first is the steady growth in the amount and complexity of data generated in various domains (termed “Big Data”), which requires new ways for organizing and querying structured, semi-structured and unstructured data. In addition, there is a shift in the characteristics of people exploring complex data. While in the past it was sufficient that only domain experts access complex domain-specific

data stores, there is a growing demand for non-experts to be able to perform complex tasks.

These two trends are well exemplified in the context of scientific publications, whose number increase steadily, along with the complexity of the network formed by articles through citations, authorship and common venues Gazni et al. (2012). On the other hand, there is growing demand for ways to explore this data from researchers, students and in some cases the public (for example patients suffering from life-changing illnesses Wicks et al. (2010)).

2.2 Exploratory search interfaces

It was shown (e.g., (Perkhofer et al. 2020; Liu et al. 2018)), that when searching over multi-dimensional data, choosing the appropriate visualization for the task can significantly reduce the required effort for exploring the data.

While examining the layout of search results in existing exploratory search interfaces, we identified two predominant types of interfaces. The first and most prominent layout is a vertical list, where the location in the list is determined by relevance to the query, with the most relevant result at the top of the list. The second widespread layout is a graph layout, where each result is a point in a two-dimensional space. The location of points is usually decided according to the level of distance or similarity between points. In many cases the graph contains edges which represent relationships between results. There is a small group of systems using layouts which are neither vertical list nor graph.

The rest of this section contains further discussion of each layout type through exemplary systems. A comparative assessment of the layout types is given in appendix A, based on a set of eight features proposed by (White and Roth 2009) as features that must be present in systems that support exploratory search activities. One generic observation is that while modern search engines provide query suggestions (i.e. Google Suggest), this capability was not found in any of the exploratory search systems surveyed.

2.2.1 Vertical list layout

The dominant layout of search results in search engines is the vertical list layout, where the most relevant results appear at the top of the list Hearst (2009). This layout is very common also in exploratory search systems, probably as many of these systems are based on search engines, in particular text-based systems. It should be noted that search engines are designed to support lookup searches.

The basic information element in modern search engines is a document containing a collection of fields, where a field may contain text, numeric value, date or data of other types Zobel and Moffat (2006). In order to support efficient search, documents are pre-processed and saved in an inverted index.

However, an inverted index is not designed to efficiently capture relationships between documents, and as a result in most search engines there is no option to see relationships between provided search results for a certain query.

The main examples of a vertical list layout are the widely used *Google Scholar* (<http://scholar.google.com>) and *Microsoft Academic Search* (<http://academic.research.microsoft.com/>), two free access search engines based on indexes containing full texts of scientific literature from various disciplines and formats. Both services cover most online peer reviewed journals of the important publishers in North America and Europe, major scientific conferences, science books and selected journals with no peer review process. Examples of search results in both Google Scholar and Microsoft Academic Search are given in Figs. 4 and 5 respectively.

Perhaps the main disadvantage of 1D vertical lists for exploratory search over entities with relationships between them, is the difficulty in showing the relationships in a clean manner. In a list, drawing lines between entities either requires that the line is drawn over almost all entities, or that the lines are drawn outside the list, thus requiring additional space, forcing the list width to be reduced. Moving from a 1D to a 2D representation reduces these concerns significantly.

The systems interfaces support keyword queries, and return the top ten results with optional paging to view more results. For each result the following details are provided: the article title, article authors (some of them with a link to a webpage summarizing the author's articles), publication venue, publication year, link to the website from which the article was obtained or another page with summary of the article, a snippet containing parts of the article matching the query, and when available, link to a PDF version of the article.

Vertical lists are also suggested in current research on exploratory search. For example, (di Sciascio et al. 2018) presents an example of a non-commercial vertical list representation for exploratory search. They augment the vertical list with a number of additional visual information, such as keywords, a textual summary, color codes for relevance, and much more. For many tasks, especially for non-experts, such a user interface may become over crowded, flooding the user with information that is hard to process. (Shukla and Hoerber 2021) use a cleaner interface, presenting keyword information, and allowing the user to augment the search by selecting papers and keywords to focus the exploration.

Faceted search options are given on the left end of the screen. In Google Scholar there are three facets, where the first allows narrowing the results by result type, the second allows restricting the results by date, and the third provides the option to order results by relevance (the default) or by date. In Microsoft Academic Search, there are seven

The screenshot shows the Google Scholar interface with the search query "exploratory search". The search results are sorted by relevance. The first result is "Exploratory search: from finding to understanding" by Gary Marchionini, published in Communications of the ACM in 2006. The second result is "Supporting exploratory search, introduction, special issue, communications of the ACM" by RW White, B Kules, and SM Drucker, published in e-prints.soton.ac.uk in 2006. The third result is "The influence of display characteristics on visual exploratory search behavior" by C Janiszewski, published in Journal of Consumer Research in 1998. The fourth result is "TweetMotif: Exploratory Search and Topic Summarization for Twitter" by B O'Connor, M Krieger, and D Ahn, published in ICWSM in 2010. The fifth result is "Exploratory search: Beyond the query-response paradigm" by RW White, RA Roth, published in Synthesis lectures on information concepts, retrieval, and services in 2009. The sixth result is "Algorithmic mediation for collaborative exploratory search" by J Pickens, G Golovchinsky, and C Shah, published in Proceedings of the 31st ACM SIGCHI Conference on Human Factors in Computing Systems in 2008.

Fig. 4 Search results for the query “Exploratory Search” in Google Scholar

The screenshot shows the Microsoft Academic Search interface with the search query "exploratory search". The search results are sorted by relevance. The first result is "Exploratory search: from finding to understanding" by Gary Marchionini, published in Communications of the ACM in 2006. The second result is "TweetMotif: Exploratory Search and Topic Summarization for Twitter" by Brendan O'Connor, Michel Krieger, and David Ahn, published in the International Conference on Weblogs and Social Media in 2010. The third result is "Exploratory Search: Beyond the Query-Response Paradigm" by Ryan W. White, Resa A. Roth, published in Synthesis Lectures on Information Concepts, Retrieval, and Services in 2009. The fourth result is "Storylines of research in diffusion of innovation: a meta-narrative approach to systematic review" by Trisha Greenhalgh, Glenn Robert, and others, published in Social Science & Medicine in 2005.

Fig. 5 Search results for the query “Exploratory Search” in Microsoft Academic Search

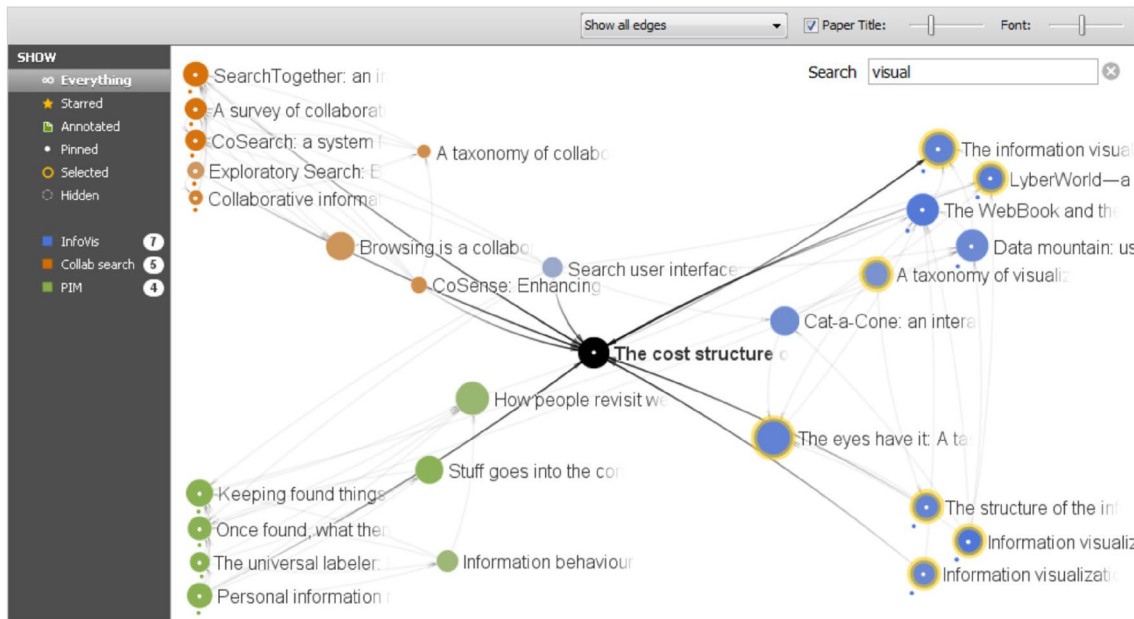


Fig. 6 Search results in the *Apollo* search system, taken from Chau et al. (2011)

different facets allowing various narrowing options. There are also three new queries stemming from each search result based on relations between articles:

- A query for retrieving all articles citing the result. The total number of citing articles is given with the result.
- A query for retrieving all articles cited by the result.
- A query for retrieving all versions of the result. The total number of versions is given with the result.

2.2.2 Graph layout

Currently, the most common alternative to the vertical list layout is the graph layout, where entities are nodes in a graph, and the relationships between entities are edges connecting the graph nodes. Search results presentation contains points scattered in space representing the entities which correspond to the query, and relationships between them are shown as lines or arrows connecting the points. In many cases the point size and line width reflect the importance of the entity or relationship they represent. While this approach allows presenting many results and supporting various visualizations, a new problem arises—how these points should be spread in space.

The *Apollo* system presents a different approach for exploring the network of scientific articles Chau et al. (2011). Unlike the conventional search process, where a large number of results are obtained and then filtered to find the ones with relevant information, the *Apollo* search process

begins with a single article known to the user. A screenshot from the *Apollo* system is given in Fig. 6.

Once the initial article is selected, the system applies an algorithm for identifying the articles most similar to it. The initial article is presented as a black circle, and the proposed articles in grey. Details regarding every article can be seen by hovering with the mouse over a certain circle. From here on the user performs an iterative process where she can select be categorized into groups, with each group represented by a different color.

New articles are added to the graph by selecting an article and obtaining new articles similar to it. The user can move any circle which represents an article to improve the visualization of data. The result of this process is a visual map which reflects the user's cognition of the articles and related fields of work. New insights could stem from identifying overlap between research fields—if a number of articles from one field cite (or are cited by) a number of articles from another field, the map will contain two clusters of articles with linking edges.

There are many additional examples of exploratory search interfaces with a graphical design. For example, (Amal et al. 2020) use a 2D graph of scholars to allow users to browse the relationships between scholars. They also add a word cloud over the edges between scholars, displaying information about the type of relationship. This type of visualization can help users in an exploratory browsing of social networks.

(Hope et al. 2020) also suggest a graphical visualization for a particular task—identifying relationships in a biomedical domain (COVID-19) between various terms and

between researchers. They suggest a number of separate visualizations, including a clique of terms, and a graph or researchers.

The Panacea system, Yokoyama et al. (2021), is also a task specific exploratory visualization, designed to help HR specialists identify appropriate candidates. The system uses a graphical layout, with larger circles representing more important entities. It is difficult to see how this method can be applied to domains with many types of entities, with varying types of relationships between them.

(Jacksi et al. 2020) propose a general use exploratory search interface, where entities are presented as circles, with edges between them marking relationships. They augment the representation with additional information on top, and the sides of the screen. We believe that an hexagon layout provides a better exploitation of the available space than using circles.

Categorization of documents plays an important role in other exploratory search systems, where it is usually done automatically. In *TopicNets* Gretarsson et al. (2012), a topic-modeling algorithm assigns articles to topics, and the topics are added to the graph as nodes, where each topic is related to the articles assigned it is assigned to. Once the user submits a query, the results layout is optimized to minimize the length of edges, thus clustering articles by topic. This allows identifying the topics within the results set, and the user can then focus on the results of interest based on a certain topic. Another outcome of this layout is a visualization of topics relatedness—two topics sharing common articles are presented close to each other, thus portraying a landscape of the topics.

The HiveRel hexagon-based visualization that we suggest is kin to the graphical visualization above. It differs from them in the exploitation of the available space, avoiding empty unused space. We also leverage a color code, allowing us to represent multiple entity types, while most systems above are restricted only to a handful of entity types (e.g., researcher, keyword, venue). Also, our layout model, where hexagons closer to the center represent entities that are more relevant to the search query, is not often used in similar exploratory visualizations.

2.2.3 Other layouts

Layouts that deviates from the conventional vertical list or graph layouts were proposed in several systems, most notably for searching over the UNESCO multilingual digital library Ruecker et al. (2012), and in *ImpactVis* which aims at providing a tool for gaining deep understanding of research impact embedded in citations network of academic research papers Wang et al. (2018). However, the system which we find currently most efficient in conducting exploratory search is *PivotPaths* (<http://mariandoerk.de/pivotpaths/>),

which provides a novel relationship-based presentation of search results.

PivotPaths presents a visual interface for exploring faceted and interconnected resources Dork et al. (2012), based on a data model containing 3 entity classes—people, resources, and concepts. People and concepts are associated to resources as facets, forming relationships. For each query the results from different classes are layered, with people at the top, resources in the middle and concepts at the bottom. Resources are located on the horizontal axis based on user selected ordering, defaulting to relevance. People and concepts are positioned according to their associated resources in a manner which puts highly connected objects far from the middle layer, inferring importance of these objects. The relationships between resources and other objects are visible, and upon hovering over certain object all objects related to it are highlighted including the connecting relationships. An example search with for the concept “Exploratory Search” in the *PivotPaths* search system is given in Fig. 7.

A query in this system is always an entity. A search session starts with a keyword query, with an auto-completion mechanism suggesting entities that can serve as queries. Once an entity is selected as query it is shown as part of the result with different coloring. Each entity on the screen can be either used as a new query or added to the existing query. If the query contains two entities, a special layout is used. Thus, while *PivotPaths* presents a clean and attractive visualization, it is difficult to see it applied in other domains, where multiple entity and relationship types apply.

Several newer approaches for information visualization were suggested, leveraging other metaphors. For example, the *Metrosets* method Jacobsen et al. (2020), presents entities as stations on a metro map, with tracks between them representing relationships. Such a representation is suitable for domains where only 1 or 2 relationships connect the entities.

(Wang et al. 2021) suggest the F2-Bubble interface, an interesting layout for representing sets of items in a 2D layout. They surround entities with a colored area, called a bubble. These bubbles can connect multiple entities that share a relationship. Although not originally designed for different types of entities, their method supports different types of relationships using color codes.

2.3 Knowledge acquisition and information visualization

Visualization as a means for knowledge acquisition has been studied in two different contexts. One area of research investigates *knowledge visualization*, where knowledge is visualized through *concept maps*. These browsable maps can improve understanding of knowledge spaces, and can be useful in various knowledge related tasks Novak (2010).

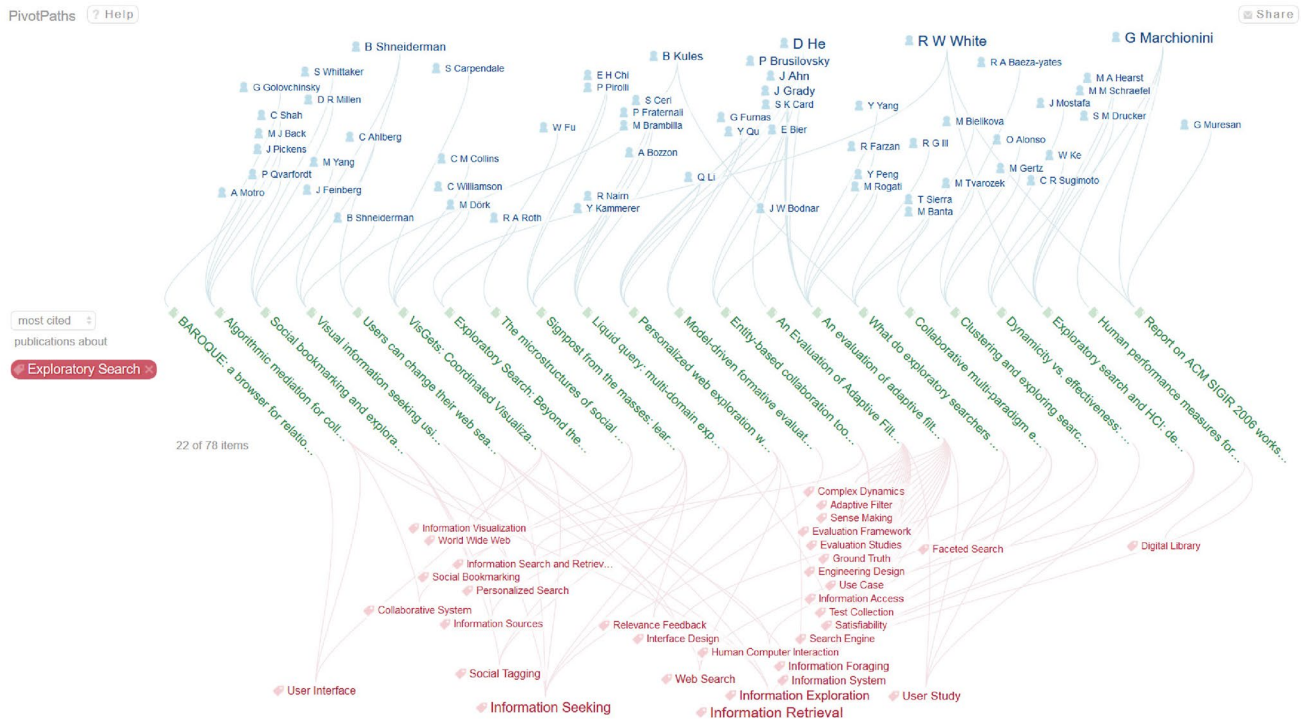


Fig. 7 Search results for the concept “Exploratory Search” in the *PivotPaths* search system

(Sarrafzadeh et al. 2014; Sarrafzadeh and Lank 2017) combined search results with concept maps using a side-by-side presentation of retrieved documents and a graph of concepts extracted from them. The positive takeaway from user studies conducted with the system was that it can be effective when the user’s information need is well defined. One conclusion from this work is that an important challenge for future research in this area is finding methods for easy, structured navigation while discovering hidden connections. This challenge stands at the core of our *HiveRel* system.

Another research area relevant to knowledge acquisition is *information visualization* Ware (2019). This field investigates ways to spatially arrange data and present it to a human viewer, in many cases aiming at gaining insights of underlying structures that translate to knowledge. (Keller et al. 2006) consider information visualization as interactive graph representations over abstract data, and discuss the knowledge acquisition aspects of such systems. Special emphasis is given to understanding the process that users must undergo when using the visualization, and how they are expected to trigger perception in different cognitive levels. They provide clear evidence that two-dimensional visualizations outperform three-dimensional ones, and that color-coded views increase performance compared to monochromatic ones.

Indeed, many of the systems above use color codes, and may hence be less useful for people which are not sensitive

to colors. Other graph based approaches also use different node sizes, which are not appropriate for our hexagon based method. This is a limitation of our system that is left for future research.

2.4 Knowledge acquisition through search

When turning from information visualization in general to systems in which a user query defines the context, relevant research falls within the intersection of human computer interaction and information retrieval. A survey Wilson et al. (2010) of the ways in which search results are organized proposed categorization into three types: result lists, two-dimensional representations and three-dimensional representations. While in result lists the focus is on fulfilling a narrow information need and three-dimensional representations have not reached the maturity level to be useful, some two-dimensional representations support knowledge acquisition. These two-dimensional visualizations arrange search results in a space based on similarity or distance derived from a single attribute type, and include most noticeably self-organizing maps (SOMs) Kohonen and Somervuo (1998); Kohonen (2013); Ahmed et al. (2020), treemap views Shneiderman and Wattenberg (2001); Sathiyarayanan and Burlutskiy (2015); Harb et al. (2021) and cluster map visualizations Fluit et al. (2006); Verbert et al. (2013). The knowledge

gained by the visual cluster structure is related to the specific attribute used to construct them.

In addition to search results layout, knowledge acquisition can be supported by providing additional actionable information of certain aspects of the result set. Thus, faceted search Mahdi et al. (2020) allows narrowing results by certain aspects, by providing the distribution of values of certain attributes across the entire result set. In addition to presenting these distributions, the user can limit attributes to specific values, thus reducing the result set to only results that match all selected values Ben-Yitzhak et al. (2008b). One example of this approach is Shneiderman's dynamic queries Shneiderman (1994), where data is provided through graphical visualization, and different controls allow the user to easily highlight specific areas in the information space. This combination provides a way to visually identify phenomena that are otherwise hidden. A good state of the art example of this approach is presented in the *TextTile* system Felix et al. (2017), which combines in a more generic way structured data with unstructured text. In this work we focus on the basic presentation of the search results and relationships between them, leaving incorporation of additional information to future research.

When placing search results on any plane, relationships and relevance should be taken into consideration, as the relevance of a result indicates its importance while relationship presentation enables knowledge acquisition. While balancing the goals of relevance and relationship presentation is challenging, we suggest that these two goals are not necessarily conflicting, and that certain layouts, including the one implemented in *HiveRel*, may provide an effective balance between them. In order to visualize this, we propose the plane in Fig. 8. The plane's x-axis is related to identification of the most important search results, with systems that support easy identification on the right side of the plane, while the y-axis relates to the level in which knowledge acquisition is enabled, with systems better suited to support it located at the top of the plane.

The dotted circles in Fig. 8 provide generalization, with systems tuned towards supporting lookup tasks fitting the lower right quadrant of the plane, while those tuned towards results clustering fit the higher left quadrant. Systems with high support on both dimensions can be ascribed as enabling *exploratory search* tasks Marchionini (2006). The two approaches discussed thus far are located near the axes. The ranked lists approach (ten blue links) is located near the x-axis, emphasizing only result importance through order—the higher the better—while providing little support for knowledge acquisition. Basic graph clustering over a single attribute ignores the importance of specific results, but allows for stronger knowledge acquisition capabilities. We thus locate single attribute clustering near the y-axis towards the top.

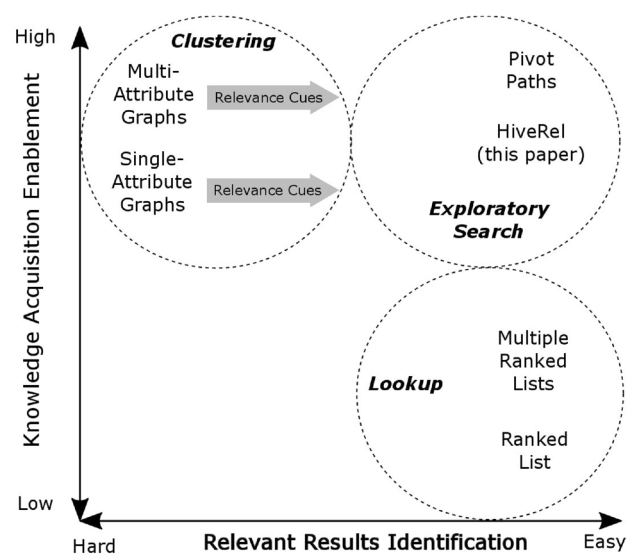


Fig. 8 A classification space for assessing the level of support search methods provide for knowledge acquisition through search, details in the text

Directions for enhancing the knowledge acquisition potential of the two basic result presentation approaches have been suggested in the field of social network retrieval. (Ronen et al. 2009), extend the ranked list approach by offering multiple ranked lists, each containing results from a different type. The ability to identify important results is not diminished as they are located towards the top of each list. However, showing results of multiple types enables presentation of relationships between results, in this case between documents, people and tags. As with other approaches, it is difficult to see how this method can be generalized beyond 2 or 3 different entity types, requiring a separate list for each type. Also, relationships between remote lists may not visualize well. This approach is called *multiple ranked lists*, and we locate it above single ranked list in Fig. 8.

The work by (Perer et al. 2013), proposes a clustering method in which the user controls which attribute is used to cluster the results by manually selecting the attribute of interest. Although this option does not increase the ability to identify important results, it provides knowledge acquisition options for a single set of results, and we therefore locate it at the top left of the plane in Fig. 8. One modification that enables improved identification of important results in any graph-based presentation is indicating importance of certain results by either larger size or stronger color. Usage of such relevance cues is depicted in Fig. 8 by the two horizontal gray arrows.

Researchers have noted that graphical visualizations require clear and meaningful axes to be understandable to users Wilson et al. (2010); Tufte (1991). Ranked list layouts use a single dimension, thus enabling a single option

for the relevance axis, from top to bottom. However, when presenting results in a two-dimensional layout, relevance axes are not trivial. In the following paragraphs we describe two alternatives for such axes implemented in two separate systems.

The *layered layout* of the *PivotPaths* system described above provides strong support for both the identification of important results and for enabling knowledge acquisition. However, we find no straightforward way to extend *PivotPaths* to handle datasets with more than three entity types or more complex relationships. This limits using *PivotPaths* to datasets with a specific data model, and we therefore propose *HiveRel* as a generic solution to the many cases where *PivotPaths* cannot be used.

HiveRel provides a center-out relevance layout. As results are placed by relevance to the search query, *HiveRel* emphasizes relevance. As we support the display of relationships, *HiveRel* also provides strong knowledge acquisition capabilities (although perhaps somewhat less than *PivotPaths*).

3 The HiveRel system

The *HiveRel* system's name is derived from its two main features: a honeycomb-like look and on-demand presentation of relationships. The design of our approach is based on the following assumptions for the knowledge acquisition task:

1. The user wants to gain knowledge on a certain subject.
2. Specification of the information need using a single query is difficult.
3. The information need cannot always be fulfilled by a single search result (document or entity).
4. The user does not know prior to searching the boundaries of the information space containing the data required to acquire the sought knowledge.

We now describe *HiveRel*, starting with how users interact with the system and later explaining the system's architecture with important implementation notes. *HiveRel* is available online at <http://hive-rel-new.appspot.com/>, which contains links to searching in different domains.

3.1 User interface

The design of our graphical user interface attempts to reflect the basic phases that we expect the user to undergo when attempting to acquire knowledge through relationships Ware (2010); Keller et al. (2006). Of the six phases hereby described only the first four are implemented and tested in the current work, leaving the last two for future work:

1. As with any query-based search system, user interaction begins with typing a textual query and submitting it to retrieve results from the search engine.
2. When search results are presented, the user first identifies the most relevant results.
3. When focusing on a certain relevant result, which we call the *active result*, the user may need to identify other results related to the active result.
4. Once related results are found, the relationships between the active result and the results related to it should be observable as an explanation for the relation.
5. Query reformulation can be used if the query given by the user did not provide sufficient results, or if the user identifies in the search results certain concepts that could be transformed into a new query and help fulfilling the information need.
6. Additional information is presented that allows dynamic narrowing or highlighting of search results.

The first design decision stems from the two-dimensional relevance layout. As the relevance of a search result is reflected by its distance from the query, we chose to place the query at the middle of a canvas surrounded by outward radiating circles of search results. In order to use the entire canvas we examined different tiling patterns, and chose a tiled hexagons pattern creating distinct circles around the center using the concepts presented by Degani et al. Degani et al. (2012). This pattern and other interface elements can be seen in Fig. 1.

Our proposed graphical user interface therefore consists of a map-like canvas, with each search result represented by an isosceles hexagon with text describing it. All result hexagons are of identical size, tiled adjacently on the surface, with a small empty space around every hexagon used for interaction purposes described below.

We borrowed from online maps three navigational concepts Harrower and Sheesley (2005). First, the canvas is movable, allowing the user to focus on certain areas of interest. Zooming in and out through either mouse wheel actions, zoom buttons or keyboard shortcuts enable viewing results in either narrow or broad view. Zooming in provides more textual data for each search result. A dedicated button at the top-left of the screen restores the default view in terms of location and zoom. The last concept is an overview window presented at the bottom right corner of the canvas, constantly providing the full view of results on the canvas.

As in some cases an hexagon cannot contain all the data available for its result, there should be a way to view hidden data. We chose to handle long mouse hover as a request for more data, and such hovering results in opening a pop-up window in the center of the canvas containing all the data of the result in focus.

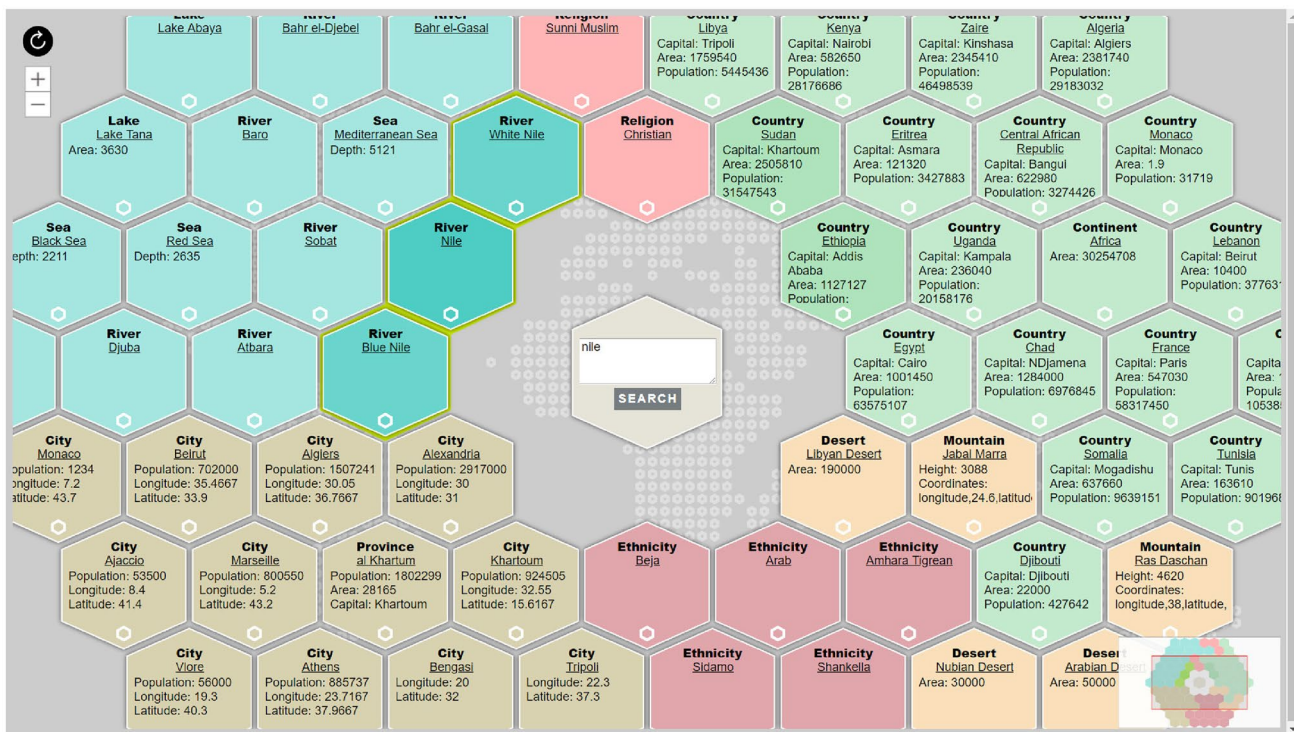


Fig. 9 Example of search results over geographic data for the query “nile”

3.1.1 Search interface design

For the **Initial search phase** of writing and submitting the query, we follow the standard design of a text box with a submission button, both located inside the query hexagon at the center of the canvas. The query hexagon is larger than other hexagons, and it is surrounded by an empty layer of hexagons to create separation between the query and the results. As this ensures easy identification of which hexagon represents the query, the background color of this hexagon is a non-prominent gray.

The **Overall results examination phase** deals with identifying the most relevant results among all presented results. In order for an object to stand out when surrounded by similar objects, there should be a single differentiating factor. That factor should be handled in the neurological processing pathway called the *what-system* Keller et al. (2006), such as color or shape. In our system search results all have the same shape, therefore more relevant results are marked using a stronger color.

The multiplicity of search result types presents an important design challenge. As we aim to allow searching in complex data sets, results from different types must be easily discernible. As with relevance, type identification must be quick, therefore each type of search results is assigned a different color. The background color of an hexagon representing a search result is therefore determined according to

its type, and the color strength according to its relevance to the search query.

In order to provide a clear view of the result types, we decided to cluster all results of the same type together. In addition, as there is no prior indication of which type is of higher interest to the user, we chose to put top scored results from each type in the innermost circle of hexagons which is most observable. In most cases, results of a certain type are co-located in a sector of the results, with top scored results in the inner circle, and most relevant results emphasized by their strong color.

Another issue raised during early evaluation of the system was the need for a visual cue to identify results that directly correspond to the textual query. Such “highlighting” effect was added by keeping a small empty space around each hexagon, and setting it to yellow for results directly corresponding to the textual query, as shown for the three results on the top left side of the query hexagon in Fig. 9. Yet, the highlighting effect may be too subtle to provide a first visual cue that draws attention. We therefore sought to use a faster and more accurate neurological pathway termed the *where-system* Keller et al. (2006), which is responsible to motion detection. The *where-system* is triggered by adding a rolling effect to the highlighted hexagons behind entities, thus drawing attention to these results. The effect of this motion is best achieved for a small number of related entities – if many entities are in motion simultaneously the view

user interaction and *Cloud Server* for processing requests from the client using data from the data server. The client is implemented in JavaScript and the other two components in Python. The cloud server is implemented using the *Google App Engine* (<https://cloud.google.com/appengine/>). Currently only Chrome and Microsoft Edge support all features required by the graphical user interface. We now describe in detail the various system components.

3.2.1 Data server

We considered the following factors when selecting the physical storage to be used in our system:

- Data meta-model definitions: entities and relationships are to be stored and indexed, raising the need for flexible data structures.
- Schema flexibility: changes to the schema used to store data may occur when enriching an existing data space with data from new sources.
- Diverse querying ability: the search algorithm requires issuing both textual and structural queries.
- Scalability: although the currently supported datasets are relatively small, the system would be most useful for large datasets, and therefore the physical storage should have inherent scaling support.

Given the above factors we chose *MongoDB* (<http://www.mongodb.org>), a NoSQL database that uses JSON-like documents with dynamic schemes, scales horizontally using sharding, and supports both structural and textual queries. We use the *MongoDB* hosting service mLab (<https://mlab.com>) for storage and retrieval of data.

3.2.2 Data modeling and storage

The data meta-model used in this work and defined hereafter is a simplified version of the conceptual entity relationship (ER) model proposed by Chen Chen (1976), with modifications to support textual search. Let $D = (E, R)$ be a dataset where E is a set of entities, and R a set of relationships between entities in E .

An entity $e \in E$ is defined as $e = (type, A, id, text)$, where $e.type$ is a string literal describing the entities class to which the entity belongs, $e.A$ a set of attributes describing the entity, $e.id$ the entity's unique identifier, and $e.text$ is a textual representing of the entity. Each attribute $a = (name, value) \in e.A$ is a name-value pair, where $a.name$ is a string literal and $a.value$ can be any type of data. $e.id$ is a minimal attribute subset that can be used to uniquely identify the entity. $e.text$ is a concatenation of string values of a subset of the entity attributes, to be used in textual queries. In a movie dataset for example, entity types can be movie,

person and genre. A person entity can have attributes for the person name, gender, date birth. The text representing a person may include the name attribute value.

A relationship $r \in R$ is defined as $r = (type, M, A)$, where $r.type$ is a string literal describing the relationship class to which the relationship belongs, $r.M$ is the set of entities taking part in the relationship and $r.A$ is a set of attributes describing the relationship. A relationship is uniquely identified by its type and combination of the relationship members identifiers. An entity member can have a role within the relationship, therefore $r.M$ is a mapping $r.M : E \rightarrow S$ where S is a set of string literals of possible role definitions, including *Nil* when the member has no role. An attribute $a \in r.A$ is a name-value pair as defined above for entity attributes. A relationship is uniquely identified by its type and the combination of relationship members identifiers. It should be noted that $r.M$ can contain more than two members, allowing non-binary relationships. In a movie dataset, a relationship of type "actor" can link an actor to a movie, with a "ranking" attribute whose value is the actor's precedence in the credentials.

Technically, in order to introduce any data artifact to *MongoDB* it must be translated to a JSON object. Such translation is straight forward for an entity $e = (type, A, id, text)$ using an associative array. The array keys are: `}}entity_type''`, `}}attributes''`, `}}entity_id''` and `}}text''`, respectively. Values are all strings, except for attributes whose value is an associative array by itself containing the pairs of the attribute name and value.

A relationship $r = (type, M, A)$ is also translated to an associative array, with respective keys `}}relationship_type''`, `}}entities''` and `}}rel_attributes''`. Relationship types and attributes are translated like entity types and attributes. Relationship members are translated to an associative array with the entity ids as keys and the entity role as value.

3.3 Information retrieval definitions

The usage of relationships in ranking mechanisms has been extensively studied in the context of the semantic web Bast et al. (2016). Anyanwu et al. Anyanwu et al. (2005) in their *SemRank* algorithm use semantic associations (relationship paths) in knowledge bases to rerank entity results sets. Lamberti et al. Lamberti et al. (2009) tackle the case where the user query contains keywords related to several concepts by linking several concept graphs (of the query, ontology and pages), and ranking by identifying relationship paths in the knowledge graph.

Another research field in which relationships are integral part of search definitions is the field of keyword-based search over relational databases Bergamaschi et al. (2016a). Keyword search aims at retrieving the database tuples matching the user keywords (or their

interpretations), using two basic techniques: schema-based and graph-based Yu et al. (2010). In the schema-based strategy the keyword query is analyzed and converted into SQL queries, while in graph-based techniques the database is treated as a graph, with tuples as nodes and relationships between tuples as edges. The search algorithms' goal is finding most relevant top- k connected tuples, using different graph structures (Steiner trees, rooted trees and others). Latest research work present combinations of the two approaches Bergamaschi et al. (2016b).

However, to the best of our knowledge the challenge of answering a user's textual query over structured or semi-structured data by retrieving the most relevant results, results most related to them and the connecting relationships was not formerly formalized. Using the above meta-model notations, we hereby provide a new set of definitions for this problem.

3.3.1 Entity and relationship weights

The first set of definitions describes relevance score for entities given a textual query, and weights for entities and relationships, to allow ranking entities and relationships given some textual query.

First, the relevance score function allows us to capture the relevance between two strings, e.g., keywords, sentences:

Definition 1 [*Relevance Score Function*] Let $S : (\Sigma^+, \Sigma^+) \rightarrow \mathbb{R}_{\geq 0}$ be a *Relevance Score Function* which assigns a numeric score for a pair of strings over alphabet Σ . In the context of dataset $D = (E, R)$, the relevance score of an entity $e \in E$ for a textual query q is given by $S(e.text, q)$.

The entity weight function is designed to allow us to specify an "importance" ordering over the entities, while the relationship weight function allows us to specify an ordering of relationships:

Definition 2 [*Entity Weight Function*] Given a dataset $D = (E, R)$, let $W_E : E \rightarrow \mathbb{R}_{\geq 0}$ be an *Entity Weight Function* which assigns for each entity $e \in E$ a numeric weight based on the entity's type $e.type$ and attributes $e.A$.

Definition 3 [*Relationship Weight Function*] Given a dataset $D = (E, R)$, let $W_R : R \rightarrow \mathbb{R}_{\geq 0}$ be a *Relationship Weight Function* which assigns for each relationship $r \in R$ a numeric weight based on the relationship's type $r.type$ and attributes $r.A$.

Using these two functions one can decide which results are more important, and hence emphasize them in our visualization by, e.g., stronger colors or larger nodes.

3.3.2 Entity and relationship hypergraph

The next group of definitions connects entities, relationships and a given textual query along with the weights defined above into one weighted hypergraph. The goal is to incorporate in the weighted hypergraph all the information that could be relevant during a search process for the given textual query.

First, we define the set of entities that are relevant to a given search query, given a relevance score function measuring relevance between the textual representation of the entity and the query.

Definition 4 [*Query Related Entities Set*] Given a dataset $D = (E, R)$, a relevance score function S and a textual query q , the *Query Related Entities Set* of q is given by $E_S(q) = \{e \in E \mid S(e.text, q) > 0\}$.

The next definition introduces the query as a pseudo-node, with query induced relationships as edges linking it to the relevant entities.

Definition 5 [*Query Induced Relationship Set*] Given a dataset $D = (E, R)$, a relevance score function S and a textual query q , for each $e \in E_S(q)$, let $r_q(e) = (query, \{q, e\}, \emptyset)$ be the *Query Induced Relationship* of e . Let $r_q(\emptyset) = (query, \{q\}, \emptyset)$ be a relationship containing only q . Then, $R_q = \{r_q(\emptyset)\} \cup \{r_q(e) \mid e \in E_S(q)\}$ is the *Query Induced Relationship Set* of q .

The weights of the newly introduced relationships corresponds to the relevance score of the query with each of the relevant entities.

Definition 6 [*Weighted Query Induced Graph*] Given a dataset $D = (E, R)$, a relevance score function S and a textual query q , the *Weighted Query Induced Graph* $Q(D, S, q)$ is the weighted graph with nodes $E_S(q) \cup \{q\}$ and edges R_q , where the weight of edge (q, e) is $S(e.text, q)$.

The weights of the relationships in the entity relationship graph correspond to the relationships weights.

Definition 7 [*Weighted Relationship Hypergraph*] Given a dataset $D = (E, R)$, entity weight function W_E and relationship weight function W_R , let the *Weighted Relationship Hypergraph* $H(D, W_E, W_R)$ be the weighted hypergraph with nodes E , edges R , node weights W_E and edge weights W_R .

Finally, can fully define the weighted hypergraph all the information that could be relevant during a search process for the given textual query, combining a "static" parts from

the dataset and weight functions with a “dynamic” part derived from a specific textual query.

Definition 8 [Weighted Relationship-Query Hypergraph] Given a dataset $D = (E, R)$, entity weight function W_E , relationship weight function W_R , a relevance score function S and a textual query q , the *Weighted Relationship-Query Hypergraph* $HQ(D, W_E, W_R, S, q)$ is defined as $H(D, W_E, W_R) \cup Q(D, S, q)$. The weight of entity $e \in HQ(D, W_E, W_R, S, q)$ is given by:

$$w(e) = \begin{cases} W_E(e), & \text{if } e \in E \\ 1, & \text{if } e = q \end{cases}$$

The weight of relationship $r \in HQ(D, W_E, W_R, S, q)$ is given by:

$$w(r) = \begin{cases} W_R(r), & \text{if } r \in R \\ 0, & \text{if } r = r_q(\emptyset) \\ S(e.text, q), & \text{if } r \in R_q \setminus \{r_q(\emptyset)\} \end{cases}$$

3.3.3 Exploration path and subgraph

In a weighted relationships-query hypergraph as defined above, the focal point is the node representing the query. The following definitions describe exploration options in the graph where the query node serves as starting point:

The basic component in the exploration process is a path, a sequence of relationships starting with a query induced relationship, that does not include a cycle where one node appears in non-adjacent relationships in the sequence.

Definition 9 [Exploration Path] Given a weighted relationship-query hypergraph $HQ(D, W_E, W_R, S, q)$, an *Exploration Path* of length $k > 0$ in HQ is a sequence of relationships of the form $P = (r_0, \dots, r_k)$, such that: (i) $r_0 = r_q(\emptyset)$ (ii) $r_i \neq r_j$ for each $1 \leq i < j \leq k$, (iii) $r_{i-1}.M \cap r_i.M \neq \emptyset$ for each $1 < i \leq k$ (i.e. for each pair of adjacent relationships there is at least one entity that is a member of both relationships).

The union of all exploration paths of certain length k defines a subgraph that includes all the entities and relationships that can be encountered in a random walk of length k starting from the query node.

Definition 10 [Exploration Subgraph] Given a weighted relationship-query hypergraph $HQ(D, W_E, W_R, S, q)$ and a depth parameter k , let P^k be a set of exploration paths of maximal length k in HQ . The *Exploration Subgraph* of P^k , marked $X(P^k)$, is the subgraph of HQ that contains all relationships in P^k and all entities that are members of these relationships. Formally, $X(P^k) = (E_{X(P^k)}, R_{X(P^k)})$, where

$$R_{X(P^k)} = \{r \mid \exists P_i \in P^k : r \in P_i\}$$

$$E_{X(P^k)} = \{e \mid \exists r \in R_{X(P^k)} : e \in r.M\}$$

3.3.4 Exploration paths scores

The following definitions provide scores for ranking exploration paths and subgraphs. These definitions are based on a probabilistic approach, where the query node as the starting point is assigned the full probability (equals 1). This probability is distributed through the hypergraph using the relationships and entities weights to dictate transition weights.

For each path of length smaller than k there is a finite set of relationships that can be used to extend the path. In order to distribute the probability of the path between all possible extensions, we define for each extension a weight that combines the weight of the extending relationship and the added entities. The transition score of each path extension is therefore its extension weight divided by the sum of all extension weights.

Definition 11 [Path Tail Transition Score] Given an exploration path $P = (r_0, \dots, r_m)$ where $1 \leq m \leq k$, let $R_{r_0, \dots, r_{m-1}}^+$ be the set of relationships in location m of all paths starting with r_0, \dots, r_{m-1} . For each $r \in R_{r_0, \dots, r_{m-1}}^+$, the intersection between the entities in r and the entities in previous relationship in the path is given by $E(r_{m-1}) \cap E(r)$. The *Path Tail Transition Score* of P is defined by:

$$S_{TT}(P) = \frac{w(r_m) \cdot \sum_{e \in E(r_{m-1}) \cap E(r_m)} w(e)}{\sum_{r \in R_{r_0, \dots, r_{m-1}}^+} w(r) \cdot \sum_{e \in E(r_{m-1}) \cap E(r)} w(e)} \quad (1)$$

Given the transition scores, the straight forward score of a path of length k corresponds to the probability of selecting this path, i.e. multiplication of the transition scores of its partial paths.

Definition 12 [Exploration Transitions Score] Given an exploration path $P = (r_0, \dots, r_m)$ where $1 \leq m \leq k$, let $P(i) = (r_0, \dots, r_i)$ be the prefix path of P of length i . The *Exploration Transitions Score* of P is defined by:

$$S_{ET}(P) = \prod_{i=1}^m S_{TT}(P(i)) \quad (2)$$

The straight forward score for a set of paths of length k corresponds to the probability of selecting these paths, i.e. sum of the scores of these paths.

Definition 13 [Sum of Paths Exploration Transitions Score] The *Sum of Paths Exploration Transitions Score* of paths $\{P_1, \dots, P_n\}$ is given by

$$S_{SET}(\{P_1, \dots, P_n\}) = \sum_{P \in P_1, \dots, P_n} S_{ET} \quad (3)$$

Based on the path scores, we now define scores for entities. First, each entity that is added to a path receives the path's score multiplied by the entity weight:

Definition 14 [Entity Exploration Transition Score] Given an exploration path $P = (r_0, \dots, r_m)$ where $1 \leq m \leq k$, the *Entity Exploration Transition Score* of $e \in r_{m-1}.M \cap r_m.M$ is defined by:

$$S_{EET}(e|P) = w(e) \cdot S_{ET}((r_0, \dots, r_{m-1})) \quad (4)$$

Then, we sum the scores of an entity across all relevant paths:

Definition 15 [Entity Sum of Exploration Transitions Score] The *Entity Sum of Exploration Transitions Score* of entity e for paths $\{P_1, \dots, P_n\}$ is given by

$$S_{ESET}(e, \{P_1, \dots, P_n\}) = \sum_{P \in \{P_1, \dots, P_n\}} S_{EET}(e|P) \quad (5)$$

3.3.5 The maximal n-bounded exploration subgraph problem

Last, given all the above definitions we define the *Maximal n-Bounded Exploration Subgraph Problem*. The limiting factor in this problem is the number of entities:

Definition 16 [n-Bounded Exploration Subgraph] An exploration subgraph $X(P^k)$ is an *n-Bounded Exploration Subgraph* if $|E_{X(P^k)}| \leq n$.

For a given limit on the number of entities, we wish to find the exploration subgraph with maximal exploration paths score:

Definition 17 [Maximal n-Bounded Exploration Subgraph] Given a weighted relationship-query hypergraph $HQ(D, W_E, W_R, S, q)$, depth parameter k , size parameter n and an exploration score scheme S , the *Maximal n-Bounded Exploration Subgraph* problem is defined as follows: find an n -bounded exploration subgraph $X(P^k)$ with maximal sum of paths exploration transitions score $S_{SET}(P^k)$.

In terms of complexity, the *Maximal n-Bounded Exploration Subgraph* is identical to the cardinality-constrained *maximum weight connected subgraph* problem, which was shown to be NP-hard for general graphs Hochbaum and Pathria (1994). Thus, the *Maximal n-Bounded Exploration Subgraph* is also NP-hard. Further theoretical discussion of

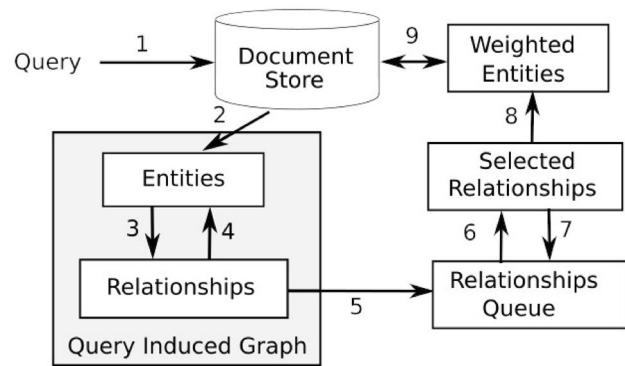


Fig. 11 Phases of the greedy search algorithm

the *Maximal n-Bounded Exploration Subgraph* remains subject to future research.

3.4 Search algorithm

In order to evaluate the interaction concepts that we propose, the results obtained for user queries in the search algorithm must be of acceptable quality. We therefore implemented a heuristic greedy algorithm which tries to find a connected subset with relatively high graph score, i.e. maximize the total score of selected entities and relationships. We provide here a high level description of the algorithm.

One obstacle in applying any optimization on a dataset $D = (E, R)$ stems from the general case of datasets containing non-binary relationships. For such datasets, selecting an edge representing a relationship r whose members are a, b and c requires branching to two different nodes. For example, assuming a was already handled, both b and c need to be processed in parallel. Thus, the complexity of any algorithm applied is bound to become high.

We chose instead to use the dual graph $\hat{D} = (R, E)$, in which relationships are nodes and entities are edges, and an entity e connects any two relationships r_1, r_2 in which e participates. In terms of ranking, the weight of each node in the dual graph corresponds to the relationship's score. Entity relevance scores for a given query are taken into account by adding query nodes to the graph, with entity relevance score as node weight. Each query node is connected to the graph by a pseudo-relationship with the relevant entity as member. The steps of the algorithm hereby described are depicted in Fig. 11. Steps are numbered to allow referral in the text.

The greedy retrieval algorithm starts with **initialization step**, in which entities whose text matches the query are retrieved using *MongoDB* native text search (with an index on the "text" field), each entity with relevance score (step 1 in Fig. 11). The relationships containing the query and retrieved entities are added to the dual graph as starting points for exploration paths (step 2 in Fig. 11).

The next part of the algorithm is an iterative **expansion step**, using entities that were found in the previous cycle to query for new relationships through entity identifiers and an index on the “entities” field (step 3 in Fig. 11). The retrieved entities and relationships are added to the dual graph with appropriate scores, and the set of new entities encountered is identified to be used in the next cycle (step 4 in Fig. 11). This procedure results in a full dual Weighted Relationship-Query graph including exploration paths up to length k , which is a pre-defined parameter of the algorithm (following tests with different values for k , in both current implementations we set $k = 2$).

In the third step, the **ranking step**, exploration paths are traversed, starting with the shortest ones. All paths of length 1 are added to a priority queue according to the path score (step 5 in Fig. 11). Then, the top-ranked path P is extracted from the queue and added with the contained entities to the result set (step 6 in Fig. 11). All paths of which P is a prefix and are longer than P by one relationships are then added to the priority queue with the appropriate path score (step 7 in Fig. 11). This iteration continues until there are no more paths that can be added to the result set without exceeding the maximal number of entities n . Once the result set entities are finalized, another pass through the exploration paths is done, this time adding all relationships that include only entities from the result set entities. As part of the iterative procedure, entity scores (S_{SET} , see Definition 15) are updated with the addition of each relationship, to be reported as part of the result set and later used in the layout computation (step 8 in Fig. 11).

After the final set of relationship and entity results is determined, there is an **attributes acquisition step** where a single query fetches the attributes of these entities to be presented in the GUI using entity identifiers and an index on the “entity_id” field (step 9 in Fig. 11).

3.4.1 Layout computation

As described in the description of the second search phase in Sect. 3.1, the layout of entities on the canvas should fulfill three goals: order the results by decreasing relevance order from the center out, cluster all results of the same type together and locate top scored results from all types in the most inner circle of hexagons. To achieve this, the layout is built from the inner circle outside. Results are bucketed by type, and the results of each type are sorted by decreasing score order. For each circle, the number of hexagons allocated to each type is proportional to the fraction of results from that type within all unassigned results, with a minimal value of 1 to assure the presence of all relevant types in the circle.

Once the number of hexagons for each type in the current layer is decided, the types are sorted using a

predefined ordering, and hexagons are allocated to the ordered types starting from the left most hexagon in the circle and moving clockwise. Since the fraction of each type may vary between adjacent circles, in some cases the hexagons of the current circle for a certain type are not connected to the ones in the previous circle, as happened to the Desert/Mountain type in Fig. 9. To minimize these errors, an adjustment procedure is applied which rotates the hexagons in the current circle, calculates a “disconnectedness score” for each formation, and selects the formation with lowest score. Assuming type t was allocated k hexagons in the current circle, the top remaining k unassigned results in the ordered results list of t are assigned to these hexagons. It should be noted that this method does not guarantee clustering all results from the same type together.

4 Evaluation

The central hypothesis of this work is that the coupling of two-dimensional layout of search results with a layer of relationship presentation provides a viable option for knowledge acquisition tasks. We examine this hypothesis in two user studies in which we compared the *HiveRel* system with available existing search systems, all of them using a ranked list layout (ten blue links). In the hereby described experiments, participants performed one set of search tasks using *HiveRel*, and another set using search engine of their choice. All participants chose to use Google, which is perhaps the toughest competitor existing, as current users have vast experience using Google, and have learned to formulate and reformulate queries rapidly and easily. On the other hand, searching with *HiveRel* requires users to learn a new interaction paradigm. It is therefore reasonable to assume that users will be biased towards the more familiar search system (Google). Thus, providing at least as good experience with an unfamiliar system would attest to the merits of the new system. A similar approach of comparing novel retrieval algorithms to Google as an existing state of the art available search algorithm, was used to test algorithms optimized for human learning Syed and Collins-Thompson (2017), educational content Cortinovis et al. (2019); Kammerer et al. (2021), query reformulation Kaur and Aggarwal (2020), and translation Safdar et al. (2020).

The evaluation is based on two similar experiments that were designed to measure accuracy, time and subjective opinions while performing specific relationship based knowledge acquisition tasks on both Google and *HiveRel*. The studies were approved by an institutional IRB expedited review subcommittee.

Table 1 Experiment groups with group sizes for the two experiments

	System1 : Data-set1	System2 : Data-set2	Size Exp. 1	Size Exp. 2
1	Google : MovieLens	<i>HiveRel</i> : Mondial	22	19
2	Google : Mondial	<i>HiveRel</i> : MovieLens	24	18
3	<i>HiveRel</i> : Mondial	Google : MovieLens	20	18
4	<i>HiveRel</i> : MovieLens	Google : Mondial	23	15

4.1 Datasets

We used two datasets from different domains—the MovieLens dataset Cantador et al. (2011), containing data on movies, actors, directors, and genres, and the Mondial dataset May (1999); Coffman and Weaver (2010), containing geographical data on countries, mountains, rivers, population and much more.

The datasets used for this study by *HiveRel* can be publicly found on the web, and thus the database that Google searches over subsumes our datasets. Moreover, for the specific tasks that we choose, we ensured that Google contains the needed information for accomplishing the tasks. For each of the datasets, we created for each experiment a set of 4 tasks with varying levels of difficulty.

4.2 Protocol and experimental design

Both experiments took place in a computer laboratory in groups of up to 15 participants per session. Participants were assigned randomly to one of the four experimental groups described in Table 1. Of the four possible domain times system combinations, each experimental group performed search tasks in two non-overlapping combinations, as reported in Table 1. This was done to prevent cases in which users repeat the search in the same domain or the same system. Counterbalancing the order of the combinations resulted in four experimental groups (i.e., Groups 1 and 3, and Groups 2 and 4 performed the same combinations in counter balanced order, respectively).

The study began with signing a consent form, followed by a short questionnaire to collect demographic data and general questions over the user expertise with web search. Then, the users started to complete the experimental tasks. The users first interacted with one system and one domain, and then with the other system on the other domain. No additional questions were asked between the two parts. While the participants were highly familiar with Google search, *HiveRel* was completely new to them. To accommodate that lack of familiarity we created two short (2.5 min) demonstration

videos. Users who used the *HiveRel* system with the MovieLens data were shown a demonstration with the Mondial data, and vice versa. The short movie was the only tutorial the users received regarding *HiveRel*. After completing the entire tasks, the users were asked to answer 5 likert-type items concerning their experience, and two open questions to express additional thoughts.

4.3 Experiment 1: simple search tasks

The first evaluation of the *HiveRel* system was performed on an early version of the system, and with simple search tasks, in order to establish the system's usability. The main differences between the early version and the current version described in Sect. 3 are as follows:

- Algorithm: the early version of *HiveRel* was based only on the definitions of Sect. 3.3.1 where weights are defined, and not on the definitions of Sects. 3.3.3 and 3.3.4 where exploration paths and scores are defined. Therefore, the search algorithm was optimizing for maximal sum of weights of the entities and relationships. The solution space for this problem is much larger than the *Maximal n-Bounded Exploration Subgraph* problem, and there is no possibility to run a greedy algorithm to solve it. Instead, we applied a search algorithm based on a different technique—random walk with restarts. This resulted in longer search time and in some cases results of lesser quality than the current version.
- Highlighting related results: in the early version, highlighting of related results was not done by shading the hexagons of unrelated entities. Instead, the background hexagon surrounding related entity hexagons was turned yellow, same as when highlighting relevant results that include the textual query.
- Querying for a certain entity: the option to directly run a new query of the entity name from the was not part of the early version.

The first two issues were mentioned by experiment participants as major hurdles in using the *HiveRel* system, and were therefore improved during the period between the two experiments.

4.3.1 Tasks

For each of the datasets we created a set of 4 tasks with low difficulty level. For the MovieLens dataset, we requested the users to: (i) list 5 movies directed by Francis Ford Coppola, (ii) list 3 actors who acted in at least one movie in which Francis Ford Coppola was involved, (iii) find the number of movies directed by Francis Ford Coppola in which Robert

Duvall played (between 1 and 6), and (iv) list 4 genres of movies in which Clint Eastwood was involved.

For the Mondial dataset the tasks were: (i) Find the countries that have a common border with Cameroon, (ii) find the country with longest common border with Cameroon, (iii) list the main ethnic groups in Cameroon, (iv) List the rivers that flow from Russia into the Arctic Ocean.

4.3.2 Sample

Ninety-three 3rd-year student volunteers from two Engineering programs started the experiment. They received academic credit for their participation. In addition, a money reward was given to the 2 best performing participants. Four participants (from 3 different groups) failed to provide valid answers to some of the tasks and were excluded from further analysis. The remaining 89 participants consisted of 26 females and 63 males, with average age of 26.

All participants reported extensive daily experience with standard web search, but only 23 of them reported similar experience with search over structured datasets, such as libraries or e-commerce websites. On average the participants rated their usage of web search as 4.93 and their use of structured search as 2.83 (1 = once every few weeks; 5 = many times a day). There were no statistically significant differences between the experimental groups on any of the background items.

4.3.3 Results: objective measures

We begin with evaluating the objective measures (response accuracy, and task completion times) obtained by the different experimental groups. We examine the accuracy of the answers based on several criteria. Most tasks had correct or incorrect answers. For these, we count the number of false positives (*#FP*) and the number of missing answers (*#MA*) with respect to the expected number. In some cases, the experimental datasets had missing data. For example, the Mondial dataset contains only the larger rivers, while in Google one can find a few more smaller rivers. When the number of correct answers differs, we use the portion of missing answers and false positives, denoted *%MA* and *%FP*, respectively. For questions with numeric answers (or single result that can be transformed into Boolean value), we report the magnitude of error—the absolute distance from the correct answer. As such, all of the metrics that we report are minimization metrics, and lower values are preferred.

Table 2 shows the average results for each system broken down by domain, task and measure. The right column lists the parameters for which the differences were statistically significant, using a between-subjects ANOVA type III test ($p \leq 0.05$, with Bonferroni correction for multiple tests of the response accuracy metrics). The factors are O—the order

Table 2 Experiment results for knowledge acquisition tasks comparing Google and *HiveRel*

	Metric	Google	<i>HiveRel</i>	$p \leq 0.05$
MovieLens	Time	11.405	9.511	S
Task (i)	<i>%FP</i>	0.090	0.034	
Task (i)	<i>%MA</i>	0.005	0.022	
Task (ii)	<i>#FP</i>	0.024	0.340	
Task (ii)	<i>#MA</i>	0.143	0.043	
Task (iii)	<i>Error</i>	0.190	0.170	
Task (iv)	<i>#FP</i>	0.452	0.447	
Task (iv)	<i>#MA</i>	0.024	0.362	S
Mondial	Time	8.319	7.714	O
Task (i)	<i>#FP</i>	0.000	0.048	
Task (i)	<i>#MA</i>	0.234	0.905	
Task (ii)	<i>Error</i>	0.067	0.095	
Task (iii)	<i>#FP</i>	0.234	0.429	
Task (iii)	<i>#MA</i>	1.319	1.381	
Task (iv)	<i>%FP</i>	0.347	0.014	S
Task (iv)	<i>%MA</i>	0.290	0.086	S

in which systems were used (Google first vs. *HiveRel* first) and S—the system that was used for the task (Google vs. *HiveRel*). Interaction between order and system was also tested with no significant results found.

In both domains search was faster using *HiveRel*, although not considerably. The difference was statistically significant in the MovieLens domain ($p = 0.004$). The relatively small difference in task completion time may be explained by the existence of a Wikipedia entry as a top result in Google's search. The Wikipedia article on Cameroon, for example, contains answers to questions about its neighboring countries as well as its ethnic groups. Detailed Wikipedia articles typically contain answer to knowledge acquisition questions attesting to the importance of such questions, but it is probable that in less popular domains where Wikipedia articles are less detailed, or in domains where relationships are more subtle, efficiency (i.e., completion time) differences would be more pronounced in favor of *HiveRel*.

In the movies domain response accuracy was similar on six of the seven measures. However, in the other two measures, Google's accuracy was better. The one significant difference due to type of system used was found in Task (iv) where using *HiveRel* resulted in significantly more missing answers. We looked into this and found a problem in the heuristic retrieval algorithm, which caused the system to fail in providing results for the query “clint eastwood genre”. For some users, this failure have terminated the search process early, resulting in missing answers.

In the geographical domain (the Mondial dataset), average time of task completion was similar for both systems. However, there was a significant order effect due

Table 3 User evaluation of different aspects of the *HiveRel* system following experiment 1

Group	1	2	3	4	All
Clarity	4.18	3.50	3.65	3.78	3.78
Interaction	3.68	3.25	3.30	3.48	3.43
Usability	3.86	3.50	3.65	3.91	3.73
Aesthetics	4.00	4.04	3.25	3.86	3.80
Usefulness	4.36	3.25	3.75	3.74	3.76
Average	4.02	3.51	3.52	3.75	

to difference in completion times between participants who started with Google (average time = 7.2 minutes) and those who started with *HiveRel* (average time = 8.9 minutes). This might be because the experiment posed two different challenges—understanding the nature of exploratory questions, and learning how to use *HiveRel*. Participants who started with Google faced these two challenges separately: the first set of tasks posed only the first challenge. On the other hand, participants who started with *HiveRel* faced both challenges at the same time while responding to the first set of tasks.

In terms of accuracy metrics, there were no statistically significant differences due to system and order in the first three Mondial tasks. However, performance of the fourth task was significantly better in *HiveRel* compared to Google. We assume that the reason is that Task (iv) was the only task in which Google search requires manual processing in order to provide a complete answer, as there is no single list of relevant rivers in the corresponding Wikipedia article. In order to validate this assumption, in the second experiment we tried to select only new search tasks that require manual processing.

4.3.4 Results: subjective measures

Following the study we asked users to evaluate various aspects of the *HiveRel* system on a scale of 1 (negative) to 5 (positive). Table 3 provides the experimental groups' average responses to these questions. The responses are overall positive, which is encouraging for a proof of concept system. We further asked the participants to provide general comments about the system. Ninety-one of the 93 participants provided such comments. This shows a high level of user engagement. Most participants, being engineering students, were comfortable enough to suggest improvements, such as stronger emphasis for links or the use of different colors. As mentioned above, two of the most common suggestions were implemented in the

system following the first experiment, and were in place before the second experiment.

4.3.5 Discussion

Considering the previous experience of experiment participants with the two systems—years of daily usage for Google against 2.5 min of introductory video for *HiveRel*, the results are encouraging. Accuracy measures of *HiveRel* are on par with those obtained by the participants using the most popular commercial search engine, while search times were faster. Search times were even faster in both domains when *HiveRel* was the second system used, suggesting that performance with the system can improve further once users master the concept of knowledge acquisition.

As with any empirical study, this study had several limitations. First, two practical constraints prevented us from genuinely testing whether *HiveRel*, a system that was designed to support knowledge acquisition in exploratory search, is indeed superior to a one-dimensional, yet extremely familiar, simple search engine—Google. One constraint was a large discrepancy in terms of users' familiarity, which could not have been bridged within the framework of a single study. Another constraint was that the experimental tasks were mostly not tailored for exploratory search (i.e., the results could have been found on Wikipedia using simple search). These constraints have put *HiveRel* at an inferior position vis a vis Google. Yet, for the single task that required non-trivial processing (Task iv in the Mondial data set), *HiveRel* decidedly outperformed Google. We take this overall virtual draw as an encouraging sign for *HiveRel*'s potential.

Another limitation of the study is that *HiveRel* was not compared to other exploratory search systems. However, this was a byproduct of the lack of available, functioning such systems. We expect that as the field matures and progresses other systems will be developed and such comparisons will become feasible. Finally, our questionnaire referred only to the participants' evaluations of *HiveRel* and not of Google. We found it pointless to ask participants to evaluate Google since they are all highly familiar with that universal search engine. Indeed, it would be very difficult to properly compare any new interface to such a well-known system. Still, users naturally report scores in comparison with the other system with which they are familiar. For example, if the usability of our system was much lower in comparison with Google, we would have expected substantially lower evaluation scores.

4.4 Experiment 2: Complex Search Tasks

As described in Sect. 4.3, following the results of the first experiment and the feedback received in the questionnaires, a few improvements were made in *HiveRel*, by introducing

new Information Retrieval definitions and corresponding algorithm, along with changes to the interface (highlighting of related results and issuing entity name query).

Once these improvements were implemented, a second experiment was designed and carried out using the new system. Given that the basic design of the first experiment was successful in comparing *HiveRel* to Google despite the limitations described above, the second experiment used a similar design. The goal performing a second experiment was to evaluate users' ability to handle more complex search tasks using *HiveRel*. The tasks of most interest in that regard are tasks that cannot be fulfilled by finding a pre-built list, and require gathering information from several web pages and identification of relationships between entities.

In this experiment for each of the two combinations of search system and data set each participant received, the time to complete the tasks (including the demonstration video) was limited to 25 min. As we expected that 25 min may not be enough to finish the search tasks, participants were instructed to mark unanswered questions as either "could not find an answer" for tasks they attempted but did not complete, or "did not attempt" for tasks they had no time to try answering. In order to balance the number of participants that attempt each search task, the tasks were rotated within each data set in different sessions.

4.4.1 Tasks

As with the first experiment, for each dataset we created a set of 4 tasks with varying levels of difficulty. For the MovieLens dataset, we requested the users to: (i) list people that worked with both Sean Connery and Arnold Schwarzenegger (in separate movies), (ii) list movies that were shot in locations in Manchester (England) during the years 2001 to 2005, (iii) list 3 actors/actresses and 3 directors that participated in films produced in West Germany, (iv) list 3 actors/actresses and 3 directors who cooperated in more than one movie with Paul Newman.

For the Mondial dataset the tasks were: (i) which country in Africa has parts of most deserts of area 35000 square kilometers or more? (ii) list 4 geographic and demographic attributes shared between Germany and Hungary, (iii) in countries in which at least 45% of the population is of Hindu religion, what other religions are held by at least 1% of the population? (iv) which city on the shores of the Pacific Ocean has the largest population?

4.4.2 Sample

74 student volunteers from two Engineering programs started the experiment. They received academic credit for their participation. In addition, a money reward was given

to the two best performing participants. Two participants failed to provide valid answers to some of the tasks and were excluded from further analysis. The remaining 72 participants consisted of 23 females and 49 males, with average age of 23.

Participants reported extensive daily experience with standard web search, but only 21 of them reported similar experience with search over structured datasets, such as libraries or e-commerce websites. On average the participants rated their usage of web search as 4.81 and their use of structured search as 2.89 (1 = once every few weeks; 5 = many times a day). There were no significant differences between the experimental groups on any of the background items.

4.4.3 Results: objective measures

We begin with an evaluation of generic objective measures— response accuracy, number of attempted tasks and task completion times—obtained by the different experimental groups. We examine the accuracy of the answers for each task based on percent of true positive results of all participants that attempted to answer the task (*%TP*), and on percent of false positive results of all participants that provided answer for the task (*%FP*). For each data set, we compute the average over all four tasks of both *%TP* and *%FP*. Number of attempted tasks and task completion times are computed for entire domain tasks.

Table 2 shows the average results for each system broken down by domain and generic measures. The significance of difference between results for the two systems was tested using Welch t-test (unpaired, unequal variances). When the difference is significant ($p < 0.05$), the result of the better performing system is given in bold face. In the movies domain *HiveRel* shows better results in all generic measures presented on Table 2—on average participants attempted more tasks in less time with significantly better *%TP* and better *%FP*. In the geographical domain (the Mondial dataset), participants using *HiveRel* performed significantly better on all measures except for completion time (Table 4).

Table 5 further shows the average results for each system broken down by domain with *%TP* and *%FP* for each task. The significance of difference between results for the two systems was tested against the difference between results for the two domains using a between-subjects ANOVA type III test ($p < 0.05$). All significant results were found for the system breakdown.

Deeper examination of the specific tasks in Table 5 revealed that while in tasks (i) and (iii) results for *HiveRel* were significantly better, in task (ii) which required listing movies that were shot in locations in Manchester (England) during the years 2001 to 2005 the percent of false positive answers when searching in Google was significantly lower

Table 4 Results for knowledge acquisition tasks comparing Google and *HiveRel* of the following measures: average time to complete the tasks (*Time*), average number of tasks each participant attempted (*#Attempted*), average number of true positive results (*%TP*) and average number of false positive (*%FP*)

	Measure	Google	<i>HiveRel</i>
MovieLens	Time	16.892	16.333
	<i>#Attempted</i>	3.667	3.865
	<i>%TP</i>	0.320	0.436
	<i>%FP</i>	0.156	0.145
Mondial	Time	15.879	18.081
	<i>#Attempted</i>	3.545	3.919
	<i>%TP</i>	0.432	0.588
	<i>%FP</i>	0.479	0.360

In rows with statistically significant difference the number of the better performing system is given in bold face

Table 5 Results for individual knowledge acquisition tasks comparing Google and *HiveRel*

	Metric	Google	<i>HiveRel</i>
MovieLens			
Task (i)	<i>%TP</i>	0.009	0.152
	<i>%FP</i>	0.257	0.156
Task (ii)	<i>%TP</i>	0.746	0.766
	<i>%FP</i>	0.043	0.109
Task (iii)	<i>%TP</i>	0.140	0.458
	<i>%FP</i>	0.148	0.098
Task (iv)	<i>%TP</i>	0.387	0.394
	<i>%FP</i>	0.189	0.227
Mondial			
Task (i)	<i>%TP</i>	0.242	0.278
	<i>%FP</i>	0.485	0.417
Task (ii)	<i>%TP</i>	0.515	0.811
	<i>%FP</i>	0.159	0.088
Task (iii)	<i>%TP</i>	0.758	0.676
	<i>%FP</i>	0.667	0.568
Task (iv)	<i>%TP</i>	0.212	0.595
	<i>%FP</i>	0.606	0.379

than with *HiveRel*. When investigating this matter, we found that there is a pre-defined list we were not aware of containing films shot in Manchester ordered by production year. We assume that a non-negligible number of participants found this list, causing the statistics to turn in favor of Google searchers.

4.4.4 Results: subjective measures

Following the experiment we asked users to evaluate various aspects of the *HiveRel* system on a scale of 1 (negative)

Table 6 User evaluation of different aspects of the *HiveRel* system following experiment 2

Aspect	Group				All
	1	2	3	4	
Clarity	3.21	3.44	3.61	3.13	3.36
Interaction	3.26	3.44	3.39	3.33	3.36
Usability	2.95	3.44	3.00	3.13g	3.13
Aesthetics	3.68	3.50	3.50	3.20	3.49
Usefulness	3.00	3.44	3.28	3.07	3.20
Average	3.22	3.46	3.36	3.17	

to 5 (positive). Average user evaluation of different aspect of the *HiveRel* system by group are given in Table 3. The evaluation items (in the order given in the table) were: (1) How clear are the components of *HiveRel*'s result pages? (2) How clear are the interaction options of *HiveRel*'s result pages? (3) How easy is it to use *HiveRel*? (4) What do you think of *HiveRel*'s aesthetics? (5) To what degree would you like to search using *HiveRel* outside of this experiment? The responses are overall mildly positive, but lower than the corresponding results in the first experiment (Table 6).

We further asked the participants to provide real life scenarios in which they could have used *HiveRel* and suggest improvements to the system. Thirty scenarios outside the experiment domains were mentioned, with repeating answers about news, travel and sports. Being engineering students, 51 of 74 participants were comfortable enough to suggest improvements. The main directions for improvement (in decreasing order of frequency) were: interaction, query tooling, UI design, explaining the system components and capabilities and performance.

4.4.5 Discussion

Following the first experiment results, the goal of the second experiment was to further evaluate *HiveRel*'s ability to assist in performing complex exploratory search tasks. The objective measures results of this experiment strengthen the assumption that *HiveRel* provides better support than existing search engines for completing such tasks. In the Mondial search tasks *HiveRel* outperformed Google in all quality measurement. In the MovieLens search tasks the true positive results were better using *HiveRel*, and the false positive results were mixed, mainly due to one search task that despite our efforts had a pre-defined list of results available through search in Google. In terms of search completion times, no system had clear advantage.

It is further likely that for more complex tasks and in less popular domains (e.g., scientific, engineering and medical information), in which less data is readily available

and organized for search on Wikipedia, the advantages of *HiveRel* for relationship based knowledge acquisition will become even more salient. We leave the examination of this conjuncture to future experiments.

While in the objective measures results comparing *HiveRel* to Google in the second experiment yielded better results than the first experiment, in the subjective measures there was a noticeable decrease in users' evaluation of different aspects of the system in the second experiment. The average evaluation for the five different items changed from value between 3.43 and 3.80 to values between 3.13 and 3.49. This phenomenon is even more surprising in light of the improvements made to the system between the two experiments. However, the experiments were different in one major aspect—the difficulty level of the search tasks. This factor could be the cause for the decrease in users' satisfaction with *HiveRel*—users in the first experiments were given search tasks that could be performed faster and with less effort than the search tasks in the second experiment, and therefore had more positive view of the system.

5 Discussion

The core question raised in this work considers the construction of a search system for non-expert users to search in data from information domains containing complex data models, and be able to gain knowledge as part of the search process. In the background section (Sect. 2), we positioned this question within the broader field of *exploratory search*—a *learning* activity termed *knowledge acquisition* (see Fig. 3). In particular, the research question aims at activities of *knowledge acquisition through search*, where the user articulates the starting point of the knowledge acquisition process through textual query.

In order to better understand the task and possible solutions, we examined existing search systems and the level of support these systems provide for knowledge acquisition, both commercially available systems and systems built in academic context. This examination process resulted in a classification space for search systems and their level of support of knowledge acquisition through search, depicted in Fig. 8.

There are two main conclusions we drew from this classification space. First, the commonly used layouts for search results—ranked list layout and graph layout—are more suited for lookup search and clustering search results

by attribute values, respectively, and do not provide clear ways to gain knowledge. Therefore, any viable solution to the challenge of knowledge acquisition through search must be based on a different layout of search results.

Given this understanding, we sought to find search systems with non-traditional search results layout, and came up with one system that clearly achieves the goal of knowledge acquisition through search: *PivotPaths* (see Sect. 2.2.3 for details). However, the layout of search results in *PivotPaths* is limited in terms of the data complexity it can support (only 3 result types), making it irrelevant for many information spaces of interest to wide publics, such as the movies and geographic datasets used in this work.

At this stage of the research, when it was clear that the major challenge at hand is finding a suitable search layout, we took a step back to portray the mental process that a searcher would undergo during a search session geared towards knowledge acquisition. The resulting process contains six phases (detailed in Sect. 3.1): textual query submission, identifying most relevant search results, locating results related to specific relevant result, viewing relationships between results of interest, query reformulation and results narrowing. With these steps in mind we developed a user interface where search results are displayed as isosceles hexagons on a map-like canvas, allowing the user to focus on certain results and view on demand the relations between the selected results and other results (see Sect. 3.1.1).

In addition to the novel search interface, the proposed search process poses a new challenge in the retrieval of search results. Given a single search query, not only relevant search results are required, but also other results which are related to those relevant search results, and the relationships that provide the connections. As no prior formulation of such requirements exists, we built a set of definitions that augments the entity-relationship data model with weights for both entities and relationships, and introduced the information retrieval concept of relevance for a given query as part of an entity relationship hypergraph. Modeling the search process components into one hypergraph allowed defining the new search problem formally, which we call the *Maximal n-Bounded Exploration Subgraph* problem (all definitions are given in Sect. 3.3).

In order to validate the proposed search process, interface and formal definitions, we implemented a full system which we call *HiveRel*. The system is written in Python and JavaScript, and uses *MongoDB* data storage and querying, and the *Google App Engine* cloud service for all other aspects of the system, in particular a greedy search algorithm providing suboptimal solutions to the *Maximal n-Bounded Exploration Subgraph* problem (described in Sect. 3.4). As part of the implementation, we modeled, parsed and created ranking schema to two datasets that were previously used in

academic research, one in a geographic and demographic domain, and the other with movie related data.

In a user study, search with *HiveRel* was compared to standard web search using Google (details in Sect. 4.3). Performance measures showed a virtual tie between the two systems, for simple search tasks despite users' lack of familiarity with *HiveRel*. In an additional user study in which participants were required to perform more complex knowledge acquisition search tasks, quality measures for task completion using *HiveRel* were better than using Google (details in Sect. 4.4). We take these results as an indication that the approach taken in this research poses a viable alternative for performing knowledge acquisition search tasks in domains with complex data, in comparison to existing methods. The potential is highest in situations that require the completion of more elaborate knowledge acquisition tasks in knowledge-rich, yet less popular domains, such as scientific publications and medical information.

5.1 Future work

There are a number of future research directions stemming from the work described in this paper:

5.1.1 Retrieval algorithm

The ranking scheme and the greedy retrieval algorithm were developed based on a relatively small number of sample queries. A major research effort to conduct following this work is creation of an evaluation framework for the new retrieval problem, by curating a list of queries and expected results along with a set of quality measurements to be applied on suggested result sets Coffman and Weaver (2010). Given such benchmark, it would be possible to try improve the retrieval algorithm to provide better optimization than the current greedy algorithm,

5.1.2 Interface

Two important phases that are currently not fully implemented are narrowing of search results and query reformulation. Narrowing of search results is a well studied field with proven methods Ben-Yitzhak et al. (2008a), therefore the challenge we see in that regard is to implement and test relevant methods in order to identify the most effective. Regarding query reformulation, we envision enabling selection of an object on the screen and performing an action that will turn this object into the next query. Among objects that can be transformed into query are textual attribute values, attribute names, entity types and even an entire entity. As with other novel design-related aspects of the system,

such interaction should be subjected to thorough empirical evaluation.

The layout computation algorithm can be further improved to ensure that results of the same type are always connected. An aspect of layout computation that is not taken into account in the current algorithm is the distance between members of each relationship. It can be argued that minimizing this distance over all relationships while maintaining ordering by type and relevance could result in inter-type clustering of results and reveal new traits of the query-related space, allowing additional knowledge acquisition options.

5.1.3 New domains

As described in the introduction of this work (Sect. 1), there are several domains in which the need arises for a search system that would assist non-expert users gain knowledge through search. Some examples for such domains are scientific publications networks, social media, biomedical data and e-commerce. Based on the effectiveness of using *HiveRel* for the movies and geographic domains, further efforts should be made to enable searching in other domains.

5.1.4 Evaluation

An important aspect of the system evaluation that was not addressed in this work is long term usage of *HiveRel*. Performing a well-designed experiment to examine the improvement in knowledge acquisition by long term users can provide deeper understanding of the benefits and disadvantages of the system.

6 Conclusion

The goal of the research done in this work was to provide means for non-expert users to perform exploratory search tasks aiming at knowledge acquisition in domains that contain complex relationships between entities of different types. While striving to achieve this goal, we made several significant contributions detailed hereby:

Information Retrieval: we provided new definitions that support exploratory search over data in domains that contain both significant textual attributes to be searched through textual query and complex relationship schema between entities. Given these definitions we proposed scores for ranking exploration paths, and a new optimization problem of these scores, the *Maximal n-Bounded Exploration Subgraph* problem. We further implemented a greedy search algorithm aiming at finding solutions to the new search problem, and used it in the *HiveRel* system using datasets from two different domains.

Search Interface: we proposed a new six-phase paradigm for supporting knowledge acquisition search tasks in domains containing entities with textual attributes and complex relationship schema between entities. We designed and implemented a new search interface based on the new paradigm, in which entities are presented on a map-like canvas in center-out relevance ranking. Presentation of relationships in which an entity or entities of interest participate is done on user demand.

System and Evaluation: by combining the previous contributions, we developed a search system termed *HiveRel*, and tested users' ability to perform knowledge acquisition tasks using *HiveRel* in two separate user studies. During the user studies, participants performed knowledge acquisition search tasks in either Google or *HiveRel*, measuring the quality of their work and the time to complete different tasks. Overall, task results quality and completion time were similar in both search systems used, and in more complex search tasks results quality of users using *HiveRel* was better than those using Google. Given that all user participants were unfamiliar with *HiveRel* beforehand and used it based on a short introductory video (2.5 min), we take these results as a viable demonstration that the approach proposed in this work can indeed assist in knowledge acquisition through search.

Appendix

Following the exploratory search concepts defined by Marchionini, White and Roth¹ proposed a set of eight features as ones that must be present in systems that support exploratory search activities. In order to simplify the assessment of exploratory search systems, we group the proposed features into three feature sets, and add a fourth criterion. The features are as follow:

1. Search activities supported by the system, including:
 - (a) Support querying and rapid query refinement,
 - (b) Offer facets and metadata-based result filtering,
 - (c) Leverage search context,
2. Presentation of search results, including:
 - (a) Offer visualizations to support insight and decision making,
 - (b) Support learning and understanding,

3. Search task management support, including:
 - (a) Facilitate collaboration,
 - (b) Offer histories, workspaces, and progress updates,
 - (c) Support task management.
4. Although not covered by White and Roth, several attributes of user interaction collectively covered by the term "User Experience" have been shown to play an important role in facilitating user engagement with interactive systems O'Brien and Toms (2010).

The following table contains comparative assessment of the different layout types described above, using the feature sets defined at the beginning of the section: search activities, presentation, task management and user experience.

	Search activities
Vertical List	Textual search supported, with options to narrow results, use search context and show documents related to a certain search result. Very limited ability to create new queries out of search results.
Graph	Not all graph based systems support keyword queries, and query reformulation is not straight forward. However, using an entity as query is easy.
Other Layouts	The three surveyed systems do not support keyword search. Two of them (the UNESCO search system and <i>PivotPaths</i> provide strong query reformulation options.
	Presentation
Vertical List	Limited presentation, mostly since the space required for each result is relatively large, causing a limit on the total number of presented results. Thus, even if there is a good way to present relationships between results, only a very small number of relationships could be shown. As a result, learning and investigation cannot be properly supported by this layout.
Graph	Flexible presentation, allowing use of size, color, line width and transparency in ways that support identification of patterns and gaining insights. One obstacle which hinders some systems is data overload, where too much information is presented without proper tools to focus on local phenomena.

¹ R. W. White, R. A. Roth, Exploratory search: Beyond the query-response paradigm, *Synthesis Lectures on Information Concepts, Retrieval, and Services* 1 (1) (2009) 1–98.

	Search activities
Other Layouts	Flexible presentation, allowing use of size, color, line width and transparency to support identification of patterns and gaining insights, with visualization strongly linked to the data model.
Vertical List	Task Management Good task management support provided by <i>querium</i> , including both retrieval of previous queries and sharing of search tasks.
Graph/Other Layouts	Task management was not addressed in any of the systems other than <i>querium</i> , probably since the developers of these systems currently focus on other challenges.
Vertical List	User Experience Very similar to web search engines, which makes the first steps in the system easier for new users. However, exploratory system are required to support complex actions which are not part of common web search, which would command significant changes to the user experience, and thus require more learning by new users.
Graph	Can be very appealing in terms of visual traits, and incorporation of new technologies (such as HTML5) could improve it even more. On the other hand, complex actions such as query reformulation are not enabled. Finding ways to enable such actions while maintaining ease of use is a major challenges.
Other Layouts	User experience in the three surveyed systems is unique in terms of graphics and functionality, which many users would find appealing. They present a viable alternative in cases where the data model deviates from the document or graph model.

Acknowledgements This paper was partially supported by the ISF fund, under grant number 1210/18.

References

- Ahmed, R.F.M., Salama, C., Mahdi, H.: Clustering research papers using genetic algorithm optimized self-organizing maps. Presented at the (2020)
- Amal, S., Adam, M., Brusilovsky, P., Minkov, E., Kuflik, T.: Personalized multifaceted visualization of scholars profiles. Presented at the (2020)

- Anyanwu, K., Maduko, A., Sheth, A.: Semrank: ranking complex relationship search results on the semantic web. Presented at the (2005)
- Bast, H., Buchhold, B., Haussmann, E., et al.: Semantic search on text and knowledge bases. *Foundations and Trends in Information Retrieval* **10**(2–3), 119–271 (2016)
- Ben-Yitzhak, O., Golbandi, N., Har'El, N., Lempel, R., Neumann, A., Ofek-Koifman, S., Sheinwald, D., Shekita, E., Sznajder, B., Yogev, S.: Beyond basic faceted search. In *Proceedings of WSDM*, pages 33–44. ACM, 2008a. ISBN 978-1-59593-927-2. <https://doi.org/10.1145/1341531.1341539>
- Ben-Yitzhak, O., Golbandi, N., Har'El, N., Lempel, R., Neumann, A., Ofek-Koifman, S., Sheinwald, D., Shekita, E., Sznajder, B., Yogev, S.: Beyond basic faceted search. Presented at the (2008b)
- Bergamaschi, S., Ferro, N., Guerra, F., Silvello, G.: In: Keyword-based search over databases: a roadmap for a reference architecture paired with an evaluation framework, pp. 1–20. Springer (2016a)
- Bergamaschi, S., Guerra, F., Interlandi, M., Trillo-Lado, R., Velegarakis, Y.: Combining user and database perspective for solving keyword queries over relational databases. *Information Systems* **55**, 1–19 (2016b)
- Cantador, I., Brusilovsky, P., Kuflik, T.: 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA, 2011. ACM
- Chau, D.H., Kittur, A., Hong, J.I., Faloutsos, C.: Apolo: interactive large graph sensemaking by combining machine learning and visualization. Presented at the (2011)
- Chen, P.: The entity-relationship model ?- toward a unified view of data. *ACM Transactions on Database Systems (TODS)* **1**(1), 9–36 (1976)
- Coffman, J., Weaver, A.C.: A framework for evaluating database keyword search strategies. Presented at the (2010)
- Cortinovis, R., Mikroyannidis, A., Domingue, J., Mulholland, P., Farrow, R.: Supporting the discoverability of open educational resources. *Educ. Inform. Technol.* **24**(5), 3129–3161 (2019)
- Date, C.J., Darwen, H.: A guide to the SQL Standard: a user's guide to the standard relational language SQL. Addison-Wesley (1989)
- Degani, A., Asherov, R., Lu, P.J.: In: Tiles and patterns of a field: From byzantine churches to user interface design, pp. 175–182. Tesselations Publishing (2012)
- di Sciascio, C., Brusilovsky, P., Veas, E.: A study on user-controllable social exploratory search. Presented at the (2018)
- Dork, M., Henry Riche, N., Ramos, G., Dumais, S.: Pivotpaths: Strolling through faceted information spaces. *Visualization and Computer Graphics*, *IEEE Transactions on* **18**(12), 2709–2718 (2012)
- Fang, L., Sarma, A.D., Yu, C., Bohannon, P.: Rex: explaining relationships between entity pairs. *Proceedings of the VLDB Endowment* **5**(3), 241–252 (2011)
- Felix, C., Pandey, A.V., Bertini, E.: Texttile: An interactive visualization tool for seamless exploratory analysis of structured data and unstructured text. *IEEE Trans. Vis. Comput. Graphics* **23**(1), 161–170 (2017)
- Fluit, C., Sabou, M., Van Harmelen, F.: Ontology-based information visualization: toward semantic web applications. In: *Visualizing the semantic web*, pp. 45–58. Springer (2006)
- Gazni, A., Sugimoto, C.R., Didegah, F.: Mapping world scientific collaboration: authors, institutions, and countries. *J. Am. Soc. Inform. Sci. Technol.* **63**(2), 323–335 (2012)
- Gretarsson, B., J. O'donovan, S. Bostandjiev, T. Höllerer, A. Asuncion, D. Newman, and P. Smyth.: Topicnets: Visual analysis of large text corpora with topic modeling. Presented at the (2012)
- Harb, Y., Sarnikar, S., El-Gayar, O.: A clustering and treemap-based approach for query reuse and visualisation in large data repositories. *Int. J. Business Intell. Data Mining* **19**(3), 267–290 (2021)

- Harrower, M., Sheesley, B.: Designing better map interfaces: A framework for panning and zooming. *Trans. GIS* **9**(2), 77–89 (2005)
- Hearst, M.: Search user interfaces. Cambridge University Press (2009)
- Hochbaum, D.S., Pathria, A.: Node-optimal connected k-subgraphs. manuscript. UC Berkeley (1994)
- Hope, T., Portenoy, J., Vasan, K., Borchardt, J., Horvitz, E., Weld, D. S., Hearst, M. A., West, J.: Scisight: Combining faceted navigation and research group detection for covid-19 exploratory scientific search. *arXiv preprint arXiv:2005.12668*, 2020
- Jacksi, K., Zeebaree, S., Dimililer, N.: Design and implementation of lod explorer: A lod exploration and visualization model. *J. Appl. Sci. Technol. Trends* **1**(2), 31–39 (2020)
- Jacobsen, B., Wallinger, M., Kobourov, S., Nöllenburg, M.: Metrosets: Visualizing sets as metro maps. *IEEE Trans. Vis. Comput. Graph.* **27**(2), 1257–1267 (2020)
- Kammerer, Y., Gottschling, S., Bråten, I.: The role of internet-specific justification beliefs in source evaluation and corroboration during web search on an unsettled socio-scientific issue. *J. Educ. Comput. Res.* **59**(2), 342–378 (2021)
- Kaur, N., Aggarwal, H.: Query reformulation approach using domain specific ontology for semantic information retrieval. *Int. J. Inform. Technol.*, 1–9 (2020)
- Keller, T., Gerjets, P., Scheiter, K., Garsoffky, B.: Information visualizations for knowledge acquisition: The impact of dimensionality and color coding. *Comput. Hum. Behavior* **22**(1), 43–65 (2006)
- Kohonen, T.: Essentials of the self-organizing map. *Neural Netw.* **37**, 52–65 (2013)
- Kohonen, T., Somervuo, P.: Self-organizing maps of symbol strings. *Neurocomputing* **21**(1), 19–30 (1998)
- Lamberti, F., Sanna, A., Demartini, C.: A relation-based page rank algorithm for semantic web search engines. *Knowl. Data Eng. IEEE Trans.* **21**(1), 123–136 (2009)
- Liu, S., Wang, X., Collins, C., Dou, W., Ouyang, F., El-Assady, M., Jiang, L., Keim, D.A.: Bridging text visualization and mining: a task-driven survey. *IEEE Trans. Visualization Comput. Graph.* **25**(7), 2482–2504 (2018)
- Mahdi, M.N., Ahmad, A.R., Ismail, R., Natiq, H., Mohammed, M.A.: Solution for information overload using faceted search—a review. *IEEE Access* **8**, 119554–119585 (2020)
- Marchionini, G.: Exploratory search: from finding to understanding. *Commun ACM* **49**(4), 41–46 (2006)
- May, W.: Information extraction and integration with florid: The mondia case study. Technical report, Technical Report 131, Universität Freiburg. Institut für Informatik (1999)
- Munzner, T.: Visualization analysis and design. AK Peters/CRC Press (2014)
- Novak, J.D.: Learning, creating, and using knowledge: Concept maps as facilitative tools in schools and corporations. Routledge (2010)
- O’Brien, H.L., Toms, E.G.: The development and evaluation of a survey to measure user engagement. *J. Am. Soc. Inform. Sci. Technol.* **61**(1), 50–69 (2010)
- Perer, A., Guy, I., Uziel, E., Ronen, I., Jacovi, M.: The longitudinal use of sandvis: visual social network analytics in the enterprise. *Vis. Comput. Graph. IEEE Trans.* **19**(7), 1095–1108 (2013)
- Perkhofer, L., Conny, W., Hofer, P.: Does design matter when visualizing big data? an empirical study to investigate the effect of visualization type and interaction use. *J. Manag. Control* **31**(1–2), 55–95 (2020)
- Prud, E., Seaborne, A., et al. Sparql query language for rdf. 2006
- Ronen, I., Shahr, E., Ur, S., Uziel, E., Yogeve, S., Zwerdling, N., Carmel, D., Guy, I., Har’El, N., Ofek-Koifman, S.: Social networks and discovery in the enterprise (sand). Presented at the (2009)
- Ruecker, S., Shiri, A., Fiorentino, C.: Interactive visualization for multilingual search. *Bull. Am. Soc. Inform. Sci. Technol.* **38**(4), 36–40 (2012)
- Russell-Rose, T., Tate, T.: Designing the search experience: the information architecture of discovery. Newnes (2012)
- Safdar, Z., Bajwa, R.S., Hussain, S., Abdullah, H.B., Safdar, K., Draz, U.: The role of roman urdu in multilingual information retrieval: A regional study. *The J. Acad. Librarianship* **46**(6), 102258 (2020)
- Sarrafzadeh, B., Lank, E.: Improving exploratory search experience through hierarchical knowledge graphs. Presented at the (2017)
- Sarrafzadeh, B., Vechtomova, O., Jokic, V.: Exploring knowledge graphs for exploratory search. Presented at the (2014)
- Sathyanarayanan, M., Burlutskiy, N.: Design and evaluation of euler diagram and treemap for social network visualisation. Presented at the (2015)
- Shneiderman, B.: Dynamic queries for visual information seeking. *IEEE Softw.* **11**(6), 70–77 (1994)
- Shneiderman, B.: The eyes have it: A task by data type taxonomy for information visualizations. In: *The craft of information visualization*, pp. 364–371. Elsevier (2003)
- Shneiderman, B., Wattenberg, M.: Ordered treemap layouts. In *Proceedings of the IEEE Symposium on Information Visualization* **73078**, 2001 (2001)
- Shukla, S., Hoeber, O.: Visually linked keywords to support exploratory browsing. Presented at the (2021)
- Syed, R., Collins-Thompson, K.: Retrieval algorithms optimized for human learning. Presented at the (2017)
- Tufte, E.R.: Envisioning information. *Optometry Vis. Sci.* **68**(4), 322–324 (1991)
- Verbert, K., Parra, D., Brusilovsky, P., Duval, E.: Visualizing recommendations to support exploration, transparency and controllability. Presented at the (2013)
- Wang, Y., Shi, C., Li, L., Tong, H., Qu, H.: Visualizing research impact through citation data. *ACM Trans. Interactive Intell. Syst. (TiiS)* **8**(1), 5 (2018)
- Wang, Y., Cheng, D., Wang, Z., Zhang, J., Zhou, L., He, G., Deussen, O.: F2-bubbles: Faithful bubble set construction and flexible editing. *IEEE Trans. Vis. Comput. Graph.* (2021)
- Ware, C.: Visual thinking: For design. Morgan Kaufmann (2010)
- Ware, C.: Information visualization: perception for design. Morgan Kaufmann (2019)
- White, R.W., Roth, R.A.: Exploratory search: Beyond the query-response paradigm. *Synthesis Lectures on Information Concepts, Retrieval, and Services* **1**(1), 1–98 (2009)
- Wickens, C.D., Helton, W.S., Hollands, J.G., Banbury, S.: *Engineering psychology and human performance*. Routledge (2021)
- Wicks, P., Massagli, M., Frost, J., Brownstein, C., Okun, S., Vaughan, T., Bradley, R., Heywood, J.: Sharing health data for better outcomes on patientslikeme. *J. Med. Internet Res.* **12**(2), (2010)
- Wilson, M.L., Kules, B., Shneiderman, B., et al.: From keyword search to exploration: Designing future search interfaces for the web. *Foundations Trends Web Sci.* **2**(1), 1–97 (2010)
- Yogev, S., Shani, G., Tractinsky, N.: Hiverel: towards focused knowledge acquisition. In: Glowacka, D., Milios, E.E., Soto, A.J., Pavlovich, F.V. (eds.) *Proceedings of the 2017 ACM Workshop on Exploratory Search and Interactive Data Analytics, ESIDA@IUI 2017*, Limassol, Cyprus, March 13, 2017, pp. 17–24. ACM (2017). <https://doi.org/10.1145/3038462.3038473>.
- Yokoyama, T.T., Okada, M., Taniguchi, T.: Panacea: Visual exploration system for analyzing trends in annual recruitment using time-varying graphs. *Plos one* **16**(3), e0247587 (2021)
- Yu, J.X., Qin, L., Chang, L.: Keyword search in relational databases: A survey. *IEEE Data Eng. Bull.* **33**(1), 67–78 (2010)
- Zobel, J., Moffat, A.: Inverted files for text search engines. *ACM Comput. Surveys (CSUR)* **38**(2), 6 (2006)

Sivan Yogev is a researcher and software engineer at Igentify Inc., working on genetic data integrations. Previously, he worked at IBM research on various problems in information retrieval, and at Outbrain Inc. on Automatic Machine Learning. He is an expert on search problems involving entity relationship data.

Guy Shani is a Professor of Software and Information Systems Engineering at Ben-Gurion University of the Negev. Previously, he was a research at Microsoft Research, working in the machine learning group.

He is interested in recommendation systems, automated decision making under uncertainty, and applied machine learning in agriculture.

Noam Tractinsky is a Professor of Software and Information Systems Engineering at Ben-Gurion University of the Negev. He studies various aspects of human-computer interaction, with a special interest in the visual aesthetics of user interfaces.