Noname manuscript No.
(will be inserted by the editor)

This a preprint of the paper accepted (in May 2016) for publication in the Journal of Intelligent Information Systems (JIIS). DOI: http://dx.doi.org/10.1007/s10844-016-0413-8
The final publication will be available at Springer

Faceted Exploration of RDF/S Datasets: A Survey

Yannis Tzitzikas · Nikos Manolis · Panagiotis Papadakos

Received: date / Accepted: date

Abstract The amounts of available Semantic Web (SW) data (including Linked Open Data) constantly increases. Users would like to browse and explore effectively such information spaces without having to be acquainted with the various vocabularies and query language syntaxes. This paper discusses the work that has been done in the area for the case of RDF/S datasets, with emphasis on session-based interaction schemes for exploratory search. In particular, it surveys the related works according to various aspects, such as assumed user goals, structuring of the underlying information space, generality and configuration requirements, and various (state space-based) features of the navigation structure. Subsequently it introduces a small but concise formal model of the interaction (that captures the core functionalities) which is used as reference model for describing what the existing systems support. Finally the paper describes the evaluation methods that have been used. Overall, the presented analysis aids the understanding and comparison of the various different approaches that have been proposed so far.

Keywords Exploratory search, faceted search, Semantic Web, RDF, RDF Schema.

1 Introduction

It is widely accepted that the structurally rich data organization of the Semantic Web (SW) is not exploited easily by end users. Either the users have to formulate complex queries that require knowing the vocabulary of the sources and the

Y. Tzitzikas (E-mail: tzitzik@ics.forth.gr)
Computer Science Department, University of Crete,
Institute of Computer Science, FORTH-ICS, GREECE

N. Manolis (E-mail: manolisn@ics.forth.gr)
Computer Science Department, University of Crete,
Institute of Computer Science, FORTH-ICS, GREECE

P. Papadakos (E-mail: papadako@ics.forth.gr)
Institute of Computer Science, FORTH-ICS, GREECE

syntax of the query language, or they have to use specialized applications (for a particular domain) that assist users in exploiting such information. The last years a vast amount of structured data has been published as *Linked Open Data (LOD)*. However, they cannot be directly exploited by end users in their current form, since better linking, browsing and presentation is required (*interaction* and *interfaces* is one of the main research challenges of LOD according to [11]). There is a need for general purpose methods for exploring such corpora which do not presuppose knowledge of the underlying vocabulary or query language. Special focus should be given on *session-based* interaction, as opposed to the state-less query and response interaction of current WSEs (Web Search Engines), for enabling the gradual formulation of complex conditions that are needed for locating the desired resources and for aiding decision making. Since plain web browsers support sessional browsing in a very primitive way (just back and forth), there is a need for more effective and flexible methods that allow users to *progressively* reach a state that satisfies them. For example, such methods would allow users to approach gradually to states whose extension corresponds to the answers of complex queries, like: *" German scientists who are known for their work in the field of evolution"*. Figure 1 demonstrates the gradual formulation of the first two conditions of that query in *Faceted Wikipedia Search* that supports interactive search over DBpedia's structured information (over 13.1 billion RDF triples). States corresponding to even more complex queries like: *" Japanese cars for sale which are driven by persons who work at FORTH and know a person who knows Bob"* should also be reachable. Finally, qualitative and quantitative overviews of the current and the possible subsequent states, should be provided at each step of the above process.

This paper surveys the work that has been done in this area. The motivation is that a large part of the related literature adopts a quite heterogeneous conceptualization and terminology, it uses informal descriptions (subject to misinterpretation), and discusses in a mixed way visualization (GUI), state space, and query language issues. Furthermore, existing surveys either focus only on the visualization aspect of the interaction (e.g. [74]), or do not focus on RDF Schema (e.g. [18]). This paper aims at alleviating the aforementioned problems by surveying the work that has been done in the area, according to various aspects, such as assumed user goals, structuring of the underlying information space, generality and configuration requirements, and various (state space) features of the navigation structure. Subsequently, the paper analyzes in detail the case of RDF/S datasets, by defining formally and precisely the *state space of the interaction*, without focusing on the various graphical visualization methods that could be applied upon a state (interested readers can refer to other works like [48] that surveys visualization and interactive visual analysis of multi-faceted scientific data). The benefits of the described "core" model is that (a) it generalizes the main exploration/browsing approaches using a small but precise and formally-defined model, (b) is query language independent (it is based on a small set of fundamental queries which can be implemented in various query languages), and (c) is visualization independent (providing freedom to plug visualization components at implementation). This analysis allows understanding and comparing (and thus assisting in extending) the various different approaches that have been proposed.

Although we confine ourselves to RDF/S, the results of our analysis and our model can be applied to any object-oriented conceptual modeling approach that

supports classes, inter-class associations, specialization/generalization hierarchies (among classes and among inter-class associations) and instantiation.

Fig. 1 An example of faceted exploration over DBpedia

The rest of the paper is organized as follows. Section 2 provides the motivation and the required background. Section 3 introduces aspects and criteria for differentiating the various approaches that have been proposed. Subsequently Section 4 focuses on RDF/S and introduces a "core" state-based interaction model for RDF/S. Section 5 discusses implementation and query language issues and then Section 6 categorizes the most indicative existing works (according to the criteria of §3) and focuses on those that are applicable to RDF/S datasets under the light of the previous model and analysis. Finally, Section 7 discusses evaluation methods, while Section 8 concludes the paper.

2 Background & Motivation

At first we discuss some background, by introducing exploratory search (§2.1) and distinguishing various kinds of information needs (§2.2), while afterwards we motivate this work (§2.3) by refering to related past surveys.

2.1 Exploratory Search

In brief, *exploratory search* [61] is a term that refers to activities carried out by searchers who are unfamiliar with the domain of their goal, the underlying technology and the process they should follow to achieve their goals. Exploratory search systems should inherently support symbiotic human-machine relationships that provide guidance in exploring unfamiliar information landscapes [99]. Therefore, exploratory search refers to a broader class of activities than typical information retrieval, where users combine querying and browsing strategies.

Faceted Exploration (or *Faceted Search*) is a widely used interaction scheme for Exploratory Search. It is the de facto query paradigm in e-commerce [83,92]. In a short (and rather informal) way we could define it as a *session-based interactive*

method for query formulation (commonly over a multidimensional information space) through simple clicks that offers an overview of the result set (groups and count information), never leading to empty results sets.

We should note that the various *graphical query formulators* (e.g. see [31,81] for SPARQL), although related, do not fall in this category. Their main focus is on facilitating the query formulation process through interactive and user friendly ways. They do not assist the exploration of the information base through an interactive process like in faceted search, a process that requires query formulation in accordance with the actual contents of the underlying dataset.

2.2 Kinds of Information Needs

In general, we can classify information needs into two very broad categories: a) *precision-oriented* ones (e.g. find the telephone of a store) and b) *recall-oriented* ones (e.g. decide which car to buy). The current general-purpose Web Search Engines (WSEs) mainly focus on single query precision-oriented needs and do not provide adequate support for the second category ones. Only a number of prototype information systems and WSEs (e.g. the ones described in [68,69,24, 75]), provide means for supporting recall-oriented information needs.

Recall-oriented needs frequently aim at *decision making*, over one or more criteria, and have an exploratory nature, like search tasks in the medical, legal, patent, and academic field, consumer-related tasks like car buying, and recently to species identification tasks [95]. Wildemuth and Freund [100] have identified the following as key attributes for exploratory tasks: a) they are associated with the goals of learning and/or investigation, b) they are general rather than specific, c) they are open-ended, d) they target multiple items, e) they involve uncertainty, f) they elicit through ill-structured information problems, g) they are dynamic, h) they are lengthy, i) they are multi-faceted, j) they are complex and finally k) they are accompanied by other information and cognitive behaviors, like sensemaking.

To the best of our knowledge there are no current reports/insights regarding the number of exploratory vs non-exploratory tasks, although there is a number of recent works focusing on the discovery of search tasks and task sessions from query logs (e.g. [56,63,39]). According to Marchionini [98] though, the majority of information needs are *recall-oriented*. In the same direction, Broder [12] categorizes WSE queries as navigational, informational and transactional (as illustrated in Figure 2). According to him, the queries that are related to recall-oriented needs (i.e. the informational and transactional queries) correspond to 80% of queries (50% for the first and 30% for the latter query category). In a more recent study, Rose and Levinson [79] raise the percentage of informational queries to 60%.

The taxonomy of tasks[1], related to the two different kinds of information needs that we consider in this paper, is illustrated in Figure 3. The left side concerns *precision*-oriented information needs, where the task's objective is to locate one resource and get information about its attributes or metadata. The right side concerns *recall*-oriented information needs. Here the objective is to locate (and get information about) a *set* of resources. In this category we can distinguish goals that require accessing sets of resources just in *groups*, or in groups accompanied

[1] Note that there is a wide variation regarding the identified tasks in the literature (refer to [55] for an overview). In this survey we focus on a faceted-oriented task analysis approach.

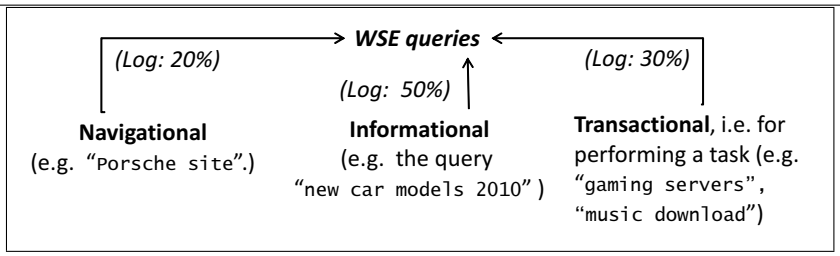

Fig. 2 Analysis of WSE queries in three categories

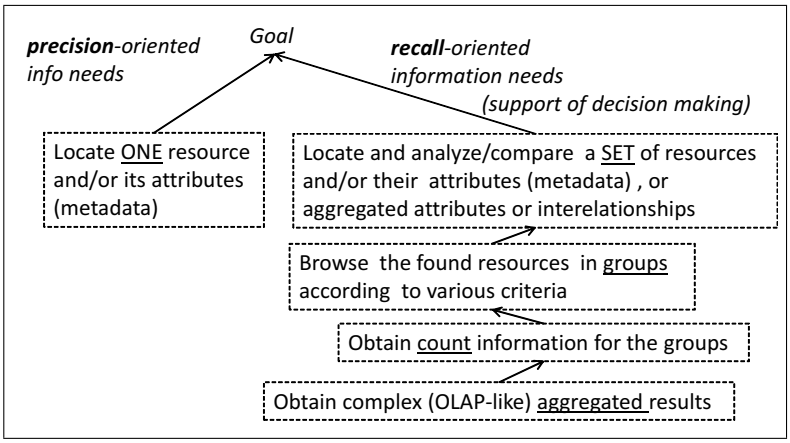

Fig. 3 Kinds of information needs and related information tasks

by *count* information for getting an overview of a set of resources, e.g. as in *Faceted Dynamic Taxonomies* (FDT) [84]. Furthermore, we may have goals that require more complex aggregated results like those provided by data warehouses. For instance, [9] proposes aggregations of arithmetic (min, max, average) and Boolean functions over the numeric attributes of the documents in the answers of free-text queries[2]. Moreover, in [21], counts are computed and displayed over combinations (pairs, triples, quadruplets, etc) of attributes (of grouping criteria in general). In comparison to OLAP (OnLine Analytical Processing) queries, in exploratory search the information demand in unknown a priori (in OLAP it is known and the schema is fixed) and the objective is not only to compute and see various aggregate values (e.g. sales per month and department), but also to support a flexible process for finding the desired individual resources.

2.3 Past Surveys

There are only a few surveys of the area. [18] proposes ER (entity-relationship) and relational modeling to explain data organization and query models respectively within 8 system's browsing functionality. However, that survey does not analyze or focus on RDF/S-oriented systems, and it does not discuss or analyze how the identified query models are technically supported by the corresponding

[2] For example, instead of just displaying the number of books of an author on a particular topic, also show the average price of the author's books.

systems. [74] surveys several faceted browsers mainly from a visualization point of view and characterizes browsers according to the supported GUI features. From another point of view, [49] at first identifies requirements for various kinds of users (metadata creators and analysts), then describes a general evaluation framework (which however is not formally defined), and finally presents a systems comparison. However, browsing and graphical query formulating tools are being compared using the same set of criteria, which is not clear since, and as already discussed, graphical query formulators target different requirements. [72] reviews approaches where ontologies are used to enhance user interfaces. One of the surveyed approaches is the so-called *ontology-based browsing*, according to which, systems exploit ontologies for browsing purposes. However, only some general characteristics of browsers are described and they are not analyzed or compared with respect to the supported states and transitions. Finally, [36] presents an empirical survey of 98 Semantic Web applications (not only browsers) from a software engineering perspective based on a architectural analysis and a questionnaire about the application functionality. An interesting finding is that 48% of the surveyed applications provide a search service on unstructured and structured data at the same time, and that more than 90% have a user interface. However, that work does not analyze the browsing services offered by the surveyed applications.

3 Aspects of the Landscape

We shall use the term EA to refer to an *exploration approach*. To fill the gaps identified in §2.3, the current survey categorizes the various EAs using criteria that do not concern only the user interface design, but also the underlying information space and the user information needs.

Furthermore it focuses on *states* and *transitions*, for being independent from the various user interface design aspects and the underlying query languages. In particular, we can identify three main aspects (which are analyzed in the subsequent sections):

- *Characteristics of the underlying information space.* The structuring of the underlying information base is an important aspect since each case requires tackling different difficulties.
- *Configuration.* Some approaches can be applied without requiring any form of configuration or application design (regarding the browseable information space), while some others require configuration steps, e.g. specify the contents and structuring of the browsable part through the view-based approach over a DB or an RDF repository. Since the browsable part of the information source is defined by a query, its structure may be different from that of the original source.
- *State Space.* In general we can view the interaction as a state space consisting of *states* and *transitions*, therefore we can characterize, or comparatively evaluate, two EAs by comparing their state spaces, e.g. by identifying properties which are satisfied by their state space.

3.1 Information Space

Some EAs are applicable to simple structures like attribute-value pairs, while others to complex structures, like Knowledge Bases (KBs). Consequently, one important aspect is how the underlying information is structured. We could distinguish various cases, below we list a few commonly occurred:

- attribute-value pairs with flat values, e.g. `name=Tom`,
- attribute-value pairs with hierarchically organized values, e.g. `location = Italy` also knowing (e.g. through a thesaurus or taxonomy) that `Italy` is a narrower term of `Europe`,
- set-valued attributes (with flat values or hierarchically-organized), e.g. `accessories={ABS, ESP, GPS}`,
- multi-entity or object-oriented, e.g. an RDF dataset containing classes like Persons, Cars, Flights etc,
- relational databases, however note that a relational schema does not have an explicit representation of its conceptual schema (and its complexity can vary),
- fuzzy data, where fuzziness can be considered as an orthogonal aspect (there are fuzzy extensions of the RDF model such as [62,60]).

How different or similar the above information spaces are, and what about the corresponding EAs? Figure 4 shows the above categories organized hierarchically where a case A is a child (direct or indirect) of a case B if whatever information can be expressed in B can also be expressed in A. The value of this diagram is that if an EA is applicable to a case A then certainly it is applicable to all cases which are parents of A (since the parents of A have more simple structure than A). For instance, an EA appropriate for RDF datasets is also appropriate for datasets consisting of attribute-value pairs with hierarchically organized values.

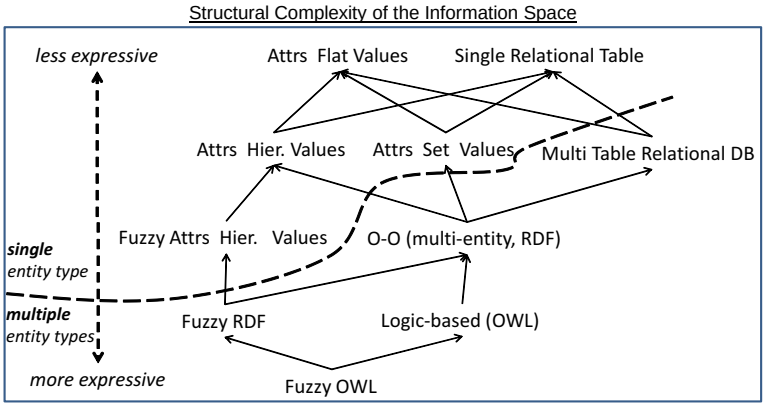

Fig. 4 Categories of information spaces according to their structural complexity

3.2 Configuration

Some approaches can be applied over a dataset without requiring any form of configuration or application design (regarding the browsable information space), while others demand (or just allow) configuration steps. In some approaches the designer should specify the contents and structuring of the browsable part of the dataset by defining a *view* using the query language supported by the DB or the RDF repository. We should make clear that the structure of the actual information space can be different from the structure of browsable part. For instance, for an RDF dataset, one can decide to provide exploration services only over the contents of the answer of a SPARQL query. In that case, the structural complexity of the browsable part is actually the same with that of a relational table. Such

transformations are not necessarily only those that are supported by the query language. For instance, the transformation can be defined by a complex process or workflow, e.g. as in [24] and [50] where the results of keyword queries over Web Search Engines (i.e. the series of records returned each comprising a URL and a textual snippet) are analyzed (using LOD), and the identified entities are used for offering faceted exploration of the search hits. In that case, the information space that is actually explored has the structural complexity of attribute-value pairs.

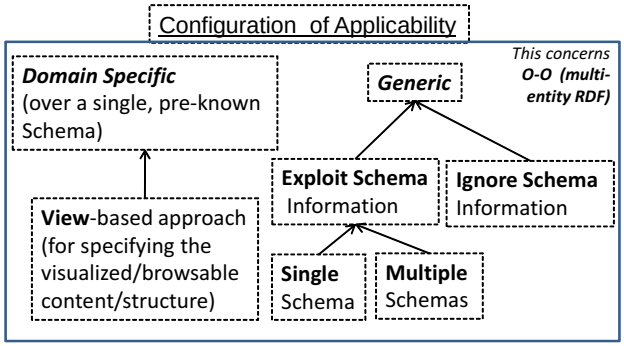

Fig. 5 Distinctions according to configurability/applicability

Figure 5 distinguishes some categories for the O-O case (e.g. RDF/S datasets) regarding the applicability (generality) or context dependency of an EA. The left side corresponds to domain specific approaches, while the right side to generic approaches. The latter can be dichotomized to those that exploit schema information (i.e. RDFS) if available, and those that do not. Those that exploit schema information can be further distinguished to those applicable to triple sets over a single schema and those applicable to triple sets over multiple schemas. All these are analyzed in the core model described in §4.

3.3 State Space (States, Transitions and Transition Markers)

We can view the interaction of an EA as a *state space* consisting of *states* and *transitions*. The user starts from the initial state and moves from state to state. Each *state* has (a) an *extension* being the set of items (or resources) displayed, (b) an *intension* being the condition/query that is satisfied by the items of the extension, and (c) a number of *transitions* each leading to a different state. Each *transition* has a clickable *transition marker* that signifies the existence of the transition and allows the user to trigger a state change. In addition each state has one or more *visualization formats* for its (a) extension, (b) intension, as well as (c) its transitions (e.g. a list, a tree-control, a table, a map). Usually one visualization per state is supported but there are works (e.g. [33]) where a state can have more than one visualization formats (some of these formats may hide some of the transitions of the state). In most cases the left bar is used for the transition markers, the top bar for the intension, and finally the middle right area for the extension, as shown in Figure 1 (DBpedia) and in Figure 7 (from the system described in [24]). Figure 6 shows the aforementioned structural elements in the form of an informal UML class diagram (Figure shows only the important multiplicities of associations).

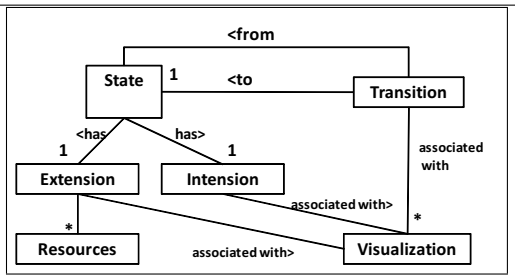

Fig. 6 The schema of the state space (as an informal UML Class Diagram)

Usually, and this is crucial, each transition marker is enriched with information regarding the *target state*. For instance, consider a dataset about hotels and suppose the user is currently at a state containing all hotels located at `Greece`. From that state there are transitions allowing the user to refine his focus, e.g. there is a transition towards a state that shows only the hotels of `Athens`. The marker of that transition is an indication of the extra condition that will be added to the intension of the current state (i.e. the extra condition `location=Athens` will be added), as well as the *size* of the extension of the new state (e.g. the count of hotels located in `Athens` could be `80`). This is why the transition markers that correspond to refinements (restrictions of the current focus) offer a sort of *synopsis* or *summarization* of the current extension, as it has been discussed in the literature (e.g. in [84,28]). Furthermore, and before clicking on `Athens`, the user may be able to see that there are more specialized refinements, such as `Historical Center` and `Olympic Stadium`. This means that the user can inspect transitions which are two or more steps away in the state space. The other way around, transition markers that move to a *broader* focus (e.g. from `Olympic Stadium` to `Athens` or to `Europe`), help the user to understand the *context* of his current focus.

Fig. 7 An indicative GUI of a state

Transitions are a key notion in any kind of *EA* and can be distinguished according to various criteria. We will categorize them on the basis of the user goals shown in Figure 3 that require accessing a *set* of resources. Figure 8 shows one distinction of such transitions, assuming an O-O structuring (since it covers RDF/S). The left part concerns transitions that do not change the entity type (e.g. the user explores cars only and she/he does not start exploring persons) and we can characterize such transitions with respect to the relationship that holds between the extension of current state and that of the target state (e.g. zoom-in/out/side), and

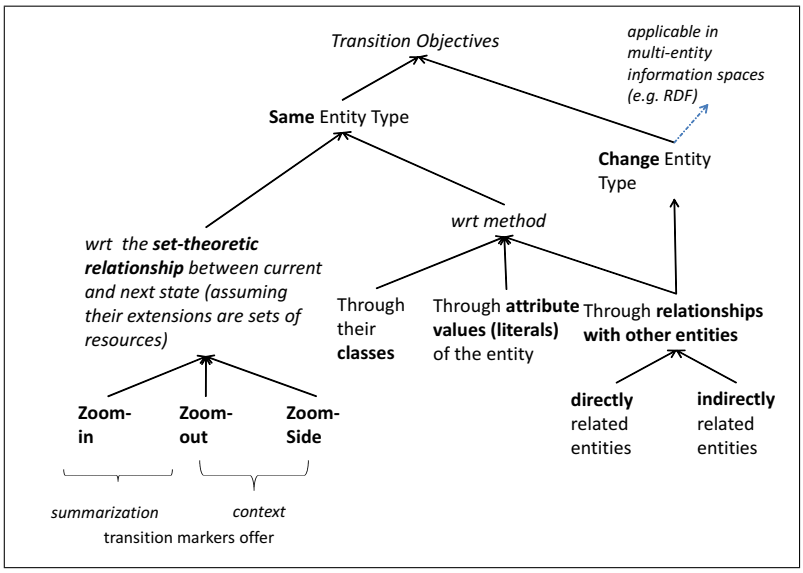

Fig. 8 A taxonomy of transitions according to the objective

with respect to the "handle" that is used for changing the focus (e.g. through their classes, attribute values, or through related entities, etc). The right part concerns transitions that can change the entity type of the current focus. To be more precise, let B denote an information base, and QL denote the query language supported by B. If q is a query in QL, we shall use $q(B)$ to denote its answer over B. A query q is subsumed by q', denoted by $q \leq q'$ if $q(B) \subseteq q'(B)$ in every information base B. Let now $s = (e, q)$ denote a state where e denotes its extension and q its intension. These two elements should satisfy the following constraint: $e = q(B)$. However, note that in approaches like the one described in [68,69], where in the context of a WSE the FDT interaction is offered over the results of a keyword query kq, B is actually the answer of the keyword query, i.e. the list of hits returned where each hit is accompanied by its metadata values (which can be hierarchically organized).

A transition from a state $s = (e, q)$ to a state $s' = (e', q')$ can be characterized according to the relationships that hold between their components. For instance, we can call a transition $s \rightarrow s'$: (a) "refinement" (or zoom-in) if $e' \subseteq e$ or $q' \leq q$, (b) "relaxation" (or zoom-out) if $e \subseteq e'$ or $q \leq q'$, (c) "side-moving" if it is neither refinement nor relaxation but $e \cap e' \neq \emptyset$. If e and e' contain entities of different types (and therefore it certainly holds $e \cap e' = \emptyset$), we can call the transition $s \rightarrow s'$ "entity type switch". Let $trans(s)$ denote all the transitions from s. Each EA actually defines a set of states, transitions and a visualization method for them. For example, in FDT the refinement transitions in $trans(s)$ are in the form of (e', q') such that $e' \neq \emptyset$ and q' is derived by replacing a conjunct of q with a term that is narrower than q (i.e. $q' \leq q$). The case of RDF/S is analyzed in detail in Section 4. Since a key characteristic of interactive search is that it is *session-based*, we can define a *session* as a sequence of states connected through transition markers, and a *refinement session* as a session where for any successive pair of states, s and s', it holds $s'.e \subseteq s.e$. If $s'.e \subset s.e$, we can call the session *strictly restrictive*, while if $s'.e = s.e$ we can call it *extensionally equivalent*. Lastly, we can call a session *single entity* if it does not contain any entity type switch transition.

4 A Core Model for Exploring RDF/S Datasets

In this section we define a precise and concise model that captures the essentials of RDF browsing approaches for recall-oriented information needs, allows accessing resources in groups (with count information) and is applicable to O-O information spaces (RDF/S). The model is part of the interaction model for Fuzzy RDF that was described in [60]. We use this model as a reference model for describing in a systematic manner the various works on exploration of RDF datasets.

RDF Notations. Let K be a set of RDF triples and let $\mathcal{C}(K)$ be its closure (i.e. the set containing also the inferred triples according to the adopted semantics). We shall denote with C the set of classes, with Pr the set of properties, with $\leq_{cl}$ the `rdfs:subclassOf` relation between classes, and with $\leq_{pr}$ the `rdfs:subPropertyOf` relation between properties. We define the instances of a class $c \in C$ as $inst(c) = \{o \mid (o, \texttt{rdf:type}, c) \in \mathcal{C}(K)\}$, and the instances of a property $p \in Pr$ as $inst(p) = \{\ (o, p, o') \mid (o, p, o') \in \mathcal{C}(K)\}$.

Running Example. We shall hereafter use the dataset illustrated in Figure 9 as our running example. An instance of the interaction over this dataset, which is based on the model that will be introduced later on, is sketched at Fig. 10. Specifically, the figure depicts only the part of the UI (usually the left bar) that shows the transition markers (it does not show the object set).

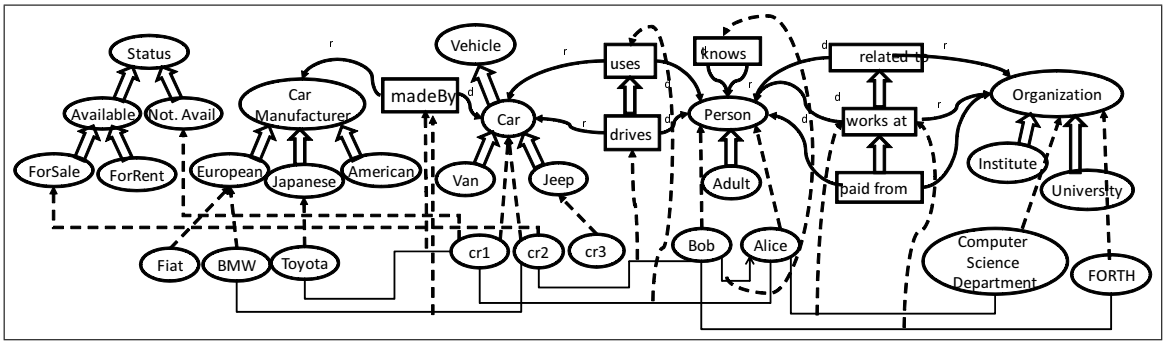

Fig. 9 An RDF/S dataset (running example)
Properties are depicted by rectangles and the letters "d" and "r" are used to denote the domain and the range of a property. Fat arrows denote `subClassOf/subPropertyOf` relationships, while dashed arrows denote `instanceOf` relationships.

Initial States Consider one RDF/S dataset with a single namespace with classes C and properties Pr. The existence of more than one namespaces can be exploited for setting extra filtering conditions throughout the interaction (e.g. to select the namespaces that should be visible or invisible). If s denotes a *state* we shall use $s.e$ to denote its *extension*. Let s_0 denote an artificial (or default) *initial state*. We can assume that $s_0.e = URI \cup LIT$, i.e. its extension can contain every URI and literal of the dataset. Other options are also possible. For instance, the extension of the initial state can be a subset of the full contents, e.g. it can be the result of a

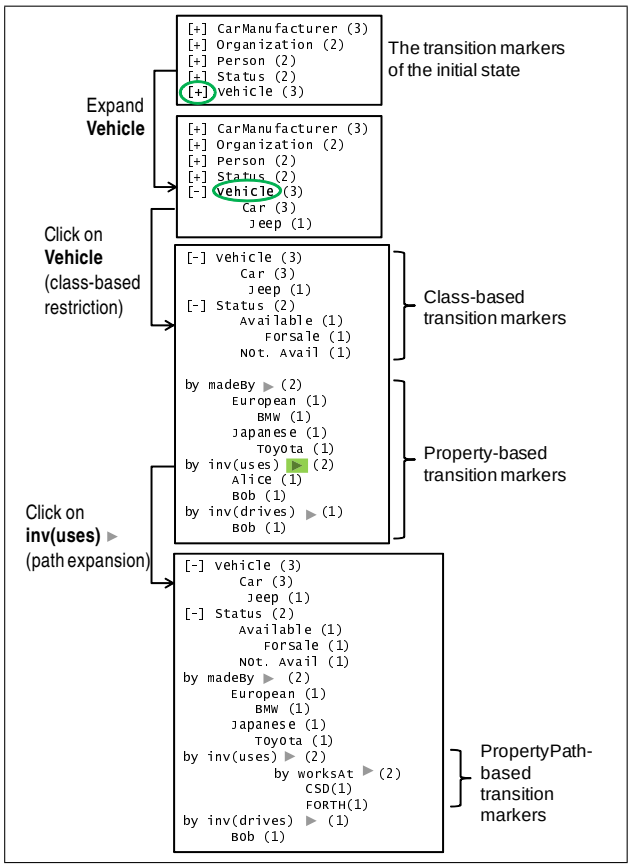

Fig. 10 Sketch of the GUI part (usually left bar) that shows transition markers

keyword query, or a subset of the set of resources of dataset as computed (filtered) by an external access method[3].

Being at a state s below we shall show how to compute the transitions that are available to that state.

From the initial state s_0 it is reasonable to offer transitions towards states corresponding to the *maximal* classes and properties, i.e. to one state for each $maximal_{\leq_{cl}}(C)$ and each $maximal_{\leq_{pr}}(Pr)$. Specifically, each $c \in maximal_{\leq_{cl}}(C)$ (resp. $p \in maximal_{\leq_{pr}}(Pr)$) yields a state with extension $inst(c)$ (resp. $inst(p)$). Below we will define formally the transitions based on the notion of *restriction* and *join*. For doing so, we need some auxiliary definitions. We shall use p^{-1} to denote the *inverse* direction of a property p, e.g. if $(d, p, r) \in Pr$ then $p^{-1} = (r, inv(p), d)$, and let Pr^{-1} denote the inverse properties of all properties in Pr. If E is a set of resources, p is a property in Pr or Pr^{-1}, v is a resource or literal, $vset$ is a set of resources or literals, and c is a class, below we define notations that *restrict* the

[3] One could adopt visualization approaches like in [48] or techniques that derive overviews, like the top-k diagrams proposed in [26], but such works go beyond the scope of this paper.

set of resources E:

$$Restrict(E, p : v) = \{\ e \in E \mid (e, p, v) \in inst(p)\}$$
$$Restrict(E, p : vset) = \{\ e \in E \mid \exists\ v' \in vset \text{ and } (e, p, v') \in inst(p)\}$$
$$Restrict(E, c) = \{\ e \in E \mid e \in inst(c)\}$$

Now we define a notation for *joining* values, i.e. for computing values which are linked with the elements of E:

$$Joins(E, p) = \{\ v \mid \exists e \in E \text{ and } (e, p, v) \in inst(p)\}$$

Now we are ready to define precisely transitions and transition markers. In brief we will define transitions for (a) *class*-based browsing, (b) *property*-based browsing, (c) *property path*-based browsing, and (d) *entity type switch*.

4.1 Class-based transitions

Suppose we are in a state s with extension $s.e$. The classes that can be used as class-based transition markers, denoted by $TM_{cl}(E)$, are those which are used in E, i.e. are defined by:

$$TM_{cl}(E) = \{c \in C \mid Restrict(E, c) \neq \emptyset\} \quad (1)$$

If the user clicks on a $c \in TM_{cl}(s.e)$, then the extension of the targeting state s' is defined as $s'.e = Restrict(s.e, c)$, and its count information is $s'.count = |s'.e|$. For example, suppose the user selects the class `Vehicle`. The user can then view its instances and follow one of the following class-based transition markers: `Vehicle, Car, Jeep, Status, Available, ForSale, Not.Available`. Notice that `ForRent` and `Van` are not included because their extension (and thus their intersection with the current extension) is empty.

The subclass relationships between the elements of $TM_{cl}(s.e)$ can be exploited for organizing them hierarchically. This can be exploited in the visualization, e.g. for producing an appropriate indentation in a text-based visualization like what is shown in Fig. 11(a) for our running example. In general, such a layout should reflect the structure of the *reflexive and transitive reduction* of the *restriction* of $\leq_{cl}$ on $TM_{cl}(s.e)$ (i.e. on $R^{refl,trans}(\leq_{cl\ |\ TM_{cl}(s.e)})$). Furthermore, based on the relationship between the extensions $s.e$ and $s'.e$, a transition (or transition marker) can be characterized as a zoom-in/out/side transition.

4.2 Property-based transitions

Suppose the user has focused on the class `Car` and therefore the extension of this state is the set {`cr1, cr2, cr3`}. The user can further restrict this set through the properties: every property whose domain or range is the class `Car`, or a superclass of `Car` (in general any property that is used in the resources in $s.e$), can be considered as a *facet* of the instances of `Car`. Consider the property `madeBy` whose domain is the class `Car` and suppose its range was the `String Literal` class. In that case the firm names of the current extension can be used as property-based transition markers. Now suppose that the range of the property `madeBy` is not literal, but the class `CarManufacturer` as it holds in our running example. In this case, the firms

(URIs in this case) of the cars in the current extension can be used as transition markers, as shown in Figure 11(b). Notice that `Fiat` is not shown as it is not related to the current focus (i.e. to `cr1`, `cr2` and `cr3`)[4]. Formally, the properties (in their defined or inverse direction) that can be used for deriving property-based transition markers are defined by:

$$Props(s) = \{p \in Pr \cup Pr^{-1} \mid Joins(s.e, p) \neq \emptyset\} \quad (2)$$

For each $p \in Props(s)$, the corresponding transition markers are those in $Joins(s, p)$, and by clicking on a v in $Joins(s, p)$ we get $s'.e = Restrict(s.e, p : v)$.

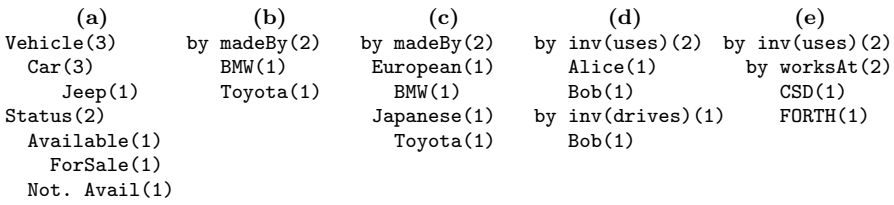

Fig. 11 Examples of transition markers

If the set of transition markers $Joins(s.e, p)$ is big, this set can be categorized according to the classes of its elements. This means that in our example, the set of firms can be categorized based on the subclasses of the class `CarManufacturer`. These classes can be shown as *intermediate nodes* of the hierarchy that lead to particular car firms, as shown in Figure 11(c). These classes are actually the classes in $TM_{cl}(Joins(s.e, p))$. This approach, apart from controlling the magnitude of the displayed information, it can also be exploited as a kind of *composite property-based transition markers*, specifically as shortcuts allowing the user to select a *set of values* with disjunctive interpretation. In our example, this means that the user can click on `Japanese` instead of clicking to every Japanese firm. Formally, clicking on a value vc ($vc \in TM_{cl}(Joins(s.e, p))$) drives to a state s' with the following extension:

$$s'.e = Restrict(s.e, p : Restrict(Joins(s.e, p), vc)) \quad (3)$$

We should note that apart from the `madeBy`-based transitions, the user can follow transitions based on the properties `inv(drives)` and `inv(uses)`, as shown in Figure 11(d). Finally, if $|Props(s)|$ is high, then the elements of $Props(s)$ could be hierarchically organized based on the `subProperty` relationships among them (analogously to the case of classes). In our running example, `paid from` could be under `works at`, which in turn could be under `related to`.

4.3 Property Path-based transitions

It is of prominent importance in semantically-structured datasets to be able to extend property-based transitions to *property path*-based transitions where paths

[4] Since `cr3` does not participate to a `madeBy` property, an alternative approach that one might follow is to add an artificial value, say `NonApplicable/Unknown`, whose count would equal 1, for informing the user that one element of the focus has no value to the `madeBy` property.

can have unlimited length. This is required for allowing users to restrict the focus through the values of *complex attributes* (e.g. addresses that may be represented as RDF blank nodes [59]) or through the relationships (direct or indirect) with other resources (e.g. provenance trails like those in [90]). In our example, a user might want to restrict the set of cars to those which are used by persons working for the `CSD` (`Computer Science Department`). In other words, the user would like transition markers of the form shown in Figure 11(e) that corresponds to path length equal to 2. It should also be possible the *successive application* of the same property (e.g. for recursive properties). For example, the user may want to focus to all friends of the friends of `Bob`, or to all friends of Bob at distance less than 5.

Below we define precisely *property path*-based transitions (expansion and cascading restriction). Let $p_1, \ldots, p_k$ be a sequence of properties. We call this sequence *successive in* s if $Joins(Joins(\ldots(Joins(s.e, p_1), p_2)\ldots p_k) \neq \emptyset$. Obviously, in alignment with the principles of faceted search, such a sequence does not lead to empty results and can be used to restrict the current focus. Let $M_1, \ldots M_k$ denote the corresponding set of transition markers at each point of the path. Assuming $M_0 = s.e$, the transition markers for all i such that $1 \leq i \leq k$, are defined as:

$$M_i = Joins(M_{i-1}, p_i) \tag{4}$$

What is left to show is how selections on such paths restrict the current focus. Suppose the user selects a value v_k from M_k. This will restrict the set of transitions markers in the following order $M_k, \ldots, M_1$ and finally it will restrict the extension of s. Let $M'_k, \ldots M'_1$ be the restricted set of transitions markers. They are defined as follows: $M'_k = \{v_k\}$, while for each $1 \leq i < k$ we have:

$$M'_i = Restrict(M_i, p_{i+1} : M'_{i+1}) \tag{5}$$

for instance, $M'_1 = Restrict(M_1, p_2 : M'_2)$. Finally, the extension of the new state s' is defined as $s'.e = Restrict(s.e, p_1 : M'_1)$. Equivalently, we can consider that M'_0 corresponds to $s'.e$ and in that case Eq. 5 holds also for $i = 0$. For example, consider the schema path `Car` $\overset{hasFirm}{\rightarrow}$ `Firm` $\overset{ofCountry}{\rightarrow}$ `Country` and three cars `cr1, cr2, cr3`, the first being `BMW`, the second `VW`, and the third `Renault`. The first two firms come from `Germany` the last from `France`. Suppose the user is on `Cars` and expands the path `hasFirm.ofCountry`. If he selects `Germany`, the previous list will become {`BMW, VW`}, since `Renault` will be excluded, and the focus will be restricted to {`cr1`,`cr2`} (as shown in Figure 12). It follows that path clicks require disjunctive interpretation of the matched values in the intermediate steps.

The above can be applied also for successive applications of the same property, e.g. `inv(drives).knows.knows.paidFrom` is a property path that could be used to restrict cars to those cars whose drivers know some persons who in turn know some persons who are paid from a particular organization.

4.4 Entity Type Switch

At the simplest case, from one specific resource the user should be able to move to one resource which is directly or indirectly connected to that. In general, if the current focus is a *set* of resources (e.g. a set of cars), the user should be able to move to one or more resources which are directly or indirectly connected

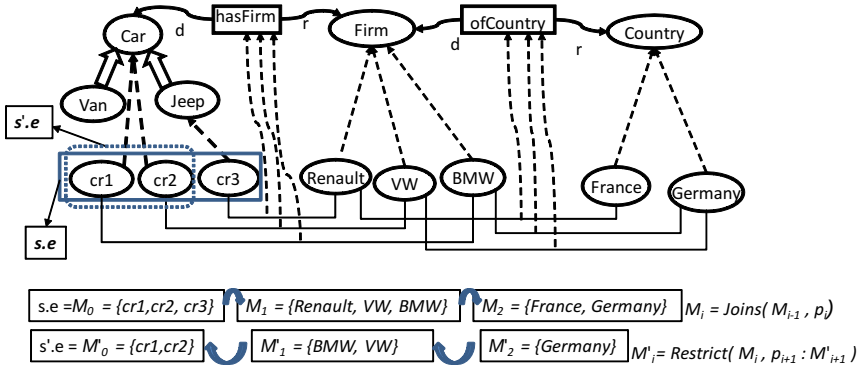

Fig. 12 An example of property path-based restriction

(to all, or at least one) of the resources of the current focus. For example, while viewing a set of cars the user should be able to move to (and focus on) the list of their firms (in this case we interpret disjunctively, i.e. we take the union of the elements associated with every object of the focus). We could capture this requirement by allowing users to move to a state whose extension is the current set of transition markers. Since these elements can belong to different classes (in comparison to the elements of the focus), we can call such transitions *entity type switch* transitions. However the classes of the transition markers are not necessarily different than those of the current focus (e.g. in the case of cyclic properties). Note that the notion of entity type corresponds to the notion of RDF class, and from this perspective, class-based transitions could also be considered as entity type switch too. However in this case the source and target type may be `subClassOf` related. To clarify the interaction, consider a user who starts from the class `Persons`, and then restricts the focus to those persons who `workAt FORTH`. Subsequently the user further restricts the focus through the property `drives`, specifically the user selects the transition marker `European`. The focus so far contains persons working at FORTH who drive European cars. At that point the user requests to change the entity type to `Cars`. This means that the entity type of the extension of the new state should be `Cars`, and the extension of the new state will contain *European cars which are driven by persons working at FORTH*. Consequently, the property `drives` (actually its inverse direction), is now a possible facet of the current focus (and a condition is already active, based on the session of the user). The user can then proceed and restrict the focus as he/she wishes to, e.g. *European cars which are driven by persons working at FORTH* and are `ForSale`, and so on.

4.5 Discussion

Notice that although the described model is minimal (the user provides plain single clicks) it includes actions which have *disjunctive* nature. In particular, property paths, as well as complex transition markers (Eq. 3) are the means to express disjunction. Also, note that the `subclassOf/subpropertyOf` semantics are disjunctive in nature (i.e. the set of instances of a class/property is the union of the instances of its subclasses/subproperties). One can easily see that it is not difficult

to provide the user with a means to select *all* possible transition markers for a given property. In that case a *property-based* transition would return all resources of the initial focus that have this property independently of its values[5]. Finally, as already explained, when an *entity type switch* takes place, the model interprets disjunctively the elements associated with every object in the current focus.

5 Implementation and Query Language Issues

Here we comment implementation approaches and query language specific issues. It is not hard to see that the key capability required for implementing the model is that of traversals (due to the `subClassOf` and `subPropertyOf` hierarchies). We can dichotomize implementation approaches, on the basis of the assumed application scenario. In the first scenario updates are not frequent and emphasis is given on maximizing the speed of exploration services. At the other extreme, we have a scenario where the updates are frequent and emphasis is given on reducing the storage space and the effort/cost for maintaining the integrity of the data after updates. A general implementation approach for the first scenario is to materialize inferred information for avoiding traversals at run time at the cost of extra memory space and more costly updates. A general implementation approach for the second scenario is to avoid storing any inferable information so that updates are supported efficiently at the cost of less efficient exploration services (due to the required traversals). In general, we can say that implementations are dichotomized to *forward* and *backward chaining* approaches. For instance, and for the case of taxonomy-based sources (Attrs with Hier. Values), dedicated indexes have been proposed (e.g. [82, 9]) for scenarios with rare updates. For scenarios with frequent updates DBMSs have also been used (e.g. [103]). However, we should mention that the latter approach requires knowing the depth of the hierarchies, or adopting query languages that support recursion[6] for computing the transition markers with one query (alternatively we can use *SQL with while*). Analogous approaches (stored closure versus `subClassOf/subPropertyOf` inference at query level) are applicable on RDF/S. For instance, and for bypassing the prohibiting (for interaction purposes) slow-down caused by RDFS and OWL-based reasoning at run time, /facet [37] (which is *a generic* browser) pre-computes and stores the closures of the transitive and inverse properties to a triplestore (supporting Prolog-based querying). We further analyze the query language perspective below.

We have defined the interaction model using the *extensions* of the states, not their *intensions*. This approach is more generic in the sense that it is independent from the particular Query Languages (QLs) which are usually used for expressing the intensions. However along the way towards implementation one may confront an information source that is accessible through a particular QL. For instance, in the context of Semantic Web and LOD, the RDF/S dataset can be stored in a local triplestore (like SESAME[7], Virtuoso[8], RDFox[9], etc.) that offers a particular

[5] This kind of transition is called *existential selection* in [66].

[6] TriQ [6] is a datalog based QL that offers a general form of recursion, reasoning and navigational capabilities, that incorporates the main RDF QLs.

[7] http://www.openrdf.org/

[8] http://docs.openlinksw.com/virtuoso/

[9] http://www.cs.ox.ac.uk/isg/tools/RDFox/

Notation	Expression in SPARQL

$Restrict(E, p : vset)$, $vset = \{v_1, ..., v_k\}$	`select ?x where {`
	`?x rdf:type temp; p ?V.`
	`Filter (?V=v_1 ||...|| ?V=v_k)}`

$Restrict(E, c)$	`select ?x where {` `?x rdf:type temp; rdf:type c.}`
$Joins(E, p)$, where $E = \{e_1, ..., e_k\}$	`select Distinct ?v where { ?x p ?v.` `Filter (?x = e_1 ||...|| ?x = e_k)}`

$TM_{cl}(s.e)$ and counts	`select Distinct ?c count(*) where{`
	`?x rdf:type ?c; rdf:type temp.}`
	`group by ?c`

$Props(s)$	`select Distinct ?p`
	`where{ {?x rdf:type temp; ?p ?v.}`
	`UNION {?m rdf:type temp. ?n ?p ?m. }}`

$Joins(s.e, p)$ and counts	`select Distinct ?v count(*)`
	`where{ ?x rdf:type temp; p ?v.}`
	`groupby ?v`

Table 1 SPARQL-expression of the model's notations (the extension of the current state is stored in temporary class `temp`)

query language (e.g. the W3C standard SPARQL [2]), or it may be remote but it is accessible through an SPARQL endpoint as it is usually the case in the LOD (Linked Open Data) cloud. Table 1 shows the notations that we have used so far in the interaction model and their expression in SPARQL, enabling in this way a straightforward "plain vanilla" implementation[10]. To aid the readability of queries, in the table we assume that the extension of the current state is stored in a temporary class with name `temp`. In Table 1 we also assume that all inferred triples are available (materialized) in the storage level. However, we should note that the general purpose RDBMS *Virtuoso* [22], supports an extended SPARQL version with `subClassOf` and `subPropertyOf` inference at query level. This means that triples entailed by `subclass` or `subproperty` statements in an inference context (built from one or more graphs containing RDF schema triples) are not physically stored, but are added to the result set during query answering. This is nowadays referred as SPARQL 1.1 RDFS entailment regime. As a result, we only have to define the inference context[11] for the SPARQL expressions of Table 1. Details about the implementation of existing systems, and the query languages they use, are given in §6.3.4.

6 Categorizing Existing Works

In this section we categorize the most indicative existing works according to the aspects introduced in §3 and the functionalities of the model introduced in §4. In particular, §6.1 concerns systems for exploring *single* entity type datasets, §6.2 is about systems for exploring *multiple* entity types datasets, and §6.3 further focuses on works for RDF/S datasets. We have done our best to include the most indicative works, however we may have missed some.

[10] This "mapping" can be implemented over any web accessible RDF/S dataset by exploiting the SPARQL extension described in [27], even if no triplestore or SPARQL endpoint is installed.

[11] Instructions are available at `http://docs.openlinksw.com/virtuoso/rdfsparqlrule.html`

System	Goal	Information space
Dynamic Faceted Search syst.[21]	Aggregation	AHV
Elastic Lists [88]	Count	AFV
Facete [87]	Count	AFV
Faceted search impl. [9]	Aggregation	AFV
Flamenco [103]	Count	AHV
Flexplorer [94] and Mitos [69]	Count	AHV
Google Scholar/Seach Tools	Group	AFV
Hippalus [70]	Count	AHSV
IOS/XSearch [24,25]	Count	AHV
product-search[89]	Count	AFV
RB++[106]	Count	AFV

Aggregation: Aggregated complex results — **AFV**: Attribute Flat Values
Count: Groups & Count info — **AHSV**: Attribute Hierarchical & Set Values
Group: Groups only (no count info) — **AHV**: Attribute Hierarchical Values

Table 2 List of *single entity* exploration approaches

System	Goal	Inform. space	Config.	Data Sources
BrowseRdf[66]	Set	O-O	I	one
Camelis2 [28]	Count	O-O	V	one
/facet [37]	Count	O-O	E	one
Faceted Data Explorer[23]	Count	O-O	V	one
Faceted Wikipedia [32]	Count	O-O	I	one
Fuzzy view based search[38]	Set	FRDF	V	one
gFacet: [35]	Set	O-O	E	one
GRQL [7]	Set	O-O	V	one
Humboldt [51]	Set	O-O	E	one
Longwell [1,73]	Count	O-O	I	one
MediaFaces [78]	Count	O-O	I	multiple
mspace [85]	Set	O-O	V	one
MuseumFinland [42]	Count	O-O	V	multiple
Odalisque [4]	Count	OWL	V	one
Ontogator [58]	Count	O-O	V	one
Parallax [41]	Count	O-O	E	multiple
Rhizomer[13]	Count	O-O	V	one
SemFacet [5]	Set	OWL 2	E	one
SPARKLIS [29]	Count	O-O	E	one
VisiNav [33]	Count	O-O	E	multiple

FRDF: Fuzzy RDF — **E**: Exploit schema
OO: Object Oriented — **I**: Ignore schema
V:View-based

Table 3 List of *multi-entity* exploration approaches

6.1 Case: Single Entity Type Datasets

Table 2 characterizes eleven (11) single entity type browsing approaches based on the user goal categories of Figure 3 and the structuring of the information space as shown in Figure 4. The indicated information space corresponds to the structure of the browseable part of the information space, which can be different from the structure of the original information space as mentioned in §3.2. Notice that the majority of systems group the hits and offer count information, however only some of them support hierarchically structured values.

6.2 Case: Multi Entity Type Datasets

Table 3 focuses on multi-entity type approaches and characterizes twenty one (21) of them according to user goals, the structuring of the information space, the configuration requirements, and multitude of underlying data sources. We can observe that 8 of them do not provide count information, only 6 exploit schema information, 9 of them follow the view-based approach and most of them are applicable to a single dataset.

6.3 Works for RDF/S datasets

Here we focus on works for RDF/S datasets and we characterize them according to the introduced core model (of §4). Section §6.3.1 focuses on transitions, §6.3.2 focuses on state space and ranking, and finally §6.3.5 discusses the differences with graphical query formulators and linked data browsers.

Current model's terminology	Literature Terminology
transitions	*navigation modes* [28], *navigation links* [4]
transition markers	*zoom points* [68], *index term* [28], *restriction value* [66]
property-based browsing	*basic selection* [66]
property path-based browsing	*indirect facets* [18], *join selection* [66]
entity type switch	*pivoting* [51,35], *refocusing* [40], [5] *reversal* [4,28], *path traversal* [33]
complex transition markers	*complex class* [28]

Table 4 Terminology mappings

System	Supported Transitions					
	Class based	Property based	Complex TMs	Property Path based	Entity Type Switch	Cycle Detection
BrowseRDF	✓	✓	×	✓	×	×
Camelis2	✓	✓	✓	×	✓	×
/facet	✓	✓	×	×	✓	×
Facete	✓	✓	×	✓	×	×
Faceted Data Explorer	✓	×	×	×	×	×
Faceted Wikipedia	✓	✓	×	×	×	×
Fuzzy view-based Search	✓	×	×	×	×	×
gFacet	✓	✓	×	✓	✓	×
GRQL	✓	✓	×	×	✓	×
Humboldt	×	✓	×	×	✓	×
Longwell	×	✓	×	×	×	×
MediaFaces	×	✓	×	×	×	×
mspace	×	✓	×	✓	×	?
MuseumFinland	✓	×	×	×	×	×
Odalisque	✓	✓	✓	×	✓	×
Ontogator	✓	×	×	×	×	×
Parallax	×	✓	×	×	✓	×
Rhizomer	✓	✓	×	×	×	×
SemFacet	✓	✓	✓	✓	✓	×
SPARKLIS	✓	✓	×	✓	✓	×
VisiNav	×	✓	×	×	✓	×

Table 5 Transitions supported by systems for exploring RDF datasets

6.3.1 Supported Transitions

Since the related literature uses a quite heterogeneous terminology, Table 4 lists some of the terms that we have used in this paper, and alternative terms which have been used in the literature. This makes evident one "difficulty" of the area, i.e. the discrepancy of the used terminology. Table 5 lists 21 systems and indicates whether they support (up to some degree) each kind of transition where ✓ means yes, × no, and ? not specified. Notice that almost all systems support class and property-based restrictions. Complex transition markers are supported only by the prototypes Odalisque [4], Camelis2 [28] and SemFacet [5] (to the best of our knowledge). Property path-based restrictions are supported by some systems, but none of them detects cycles and prevents users from making redundant steps. Also notice that in average, the number of supported transition types is rather low and only one system receives five ✓ (the maximum is six). This indicates that there is room for improvement in several of the existing systems.

6.3.2 State Space and Ranking

Since the number of states and transitions can be numerous, there is a need for various ranking methods. This is important for datasets that contain several classes and properties since in such cases the transitions markers can be too many to fit on the screen. To tackle this problem *ranking methods* are employed for identifying the more important transition markers (according to various criteria). The top-ranked transition markers are then shown, while the rest can become visible on demand, usually by pressing a "more" button or link. Figure 13 shows a taxonomy as regards what it is ranked and on what basis it is ranked, and then Table 6 categorizes some of the related works, mainly those corresponding to running systems (the rest works are described below). In particular, the left part Figure 13 concerns what is ranked, and the right part concerns the information upon which ranking is performed. For instance, BrowseRDF [66] introduces metrics for automatically ranking facets (actually properties in the RDF language) based on their *frequency* and the number of values (property values) associated with them. A frequency-based ranking method is also adopted in [33, 32] (VisiNav and Faceted Wikipedia) for ranking properties and property values. In eBay.com the most important facets (attributes) are determined in advance through *query and click logs* (as also discussed in [34]). Facetedpedia [54] proposes metrics for ranking facets (attributes) hierarchies based on a navigational cost model based on structural characteristics. In MediaFaces [78] (Yahoo's image search engine), the candidate facets (properties) after a text query are being ranked according to a statistical analysis of image search *query logs*. Faceted Data Explorer [23] offers interactive browsing over billions of triples, combining full text search and structured querying, where classes are ordered according to the number of instances that have a property value matching with a particular text query. From another point of view, [38] presents a fuzzy view-based approach in which navigation results are ranked according to resource's fuzzy descriptions (based on manually defined fuzzy concept inclusion axioms). Finally, [71] shows that it is beneficial if the systems let the users to explore the available choices and at the same time, i.e during browsing/exploration of the information base, allow them to express their preferences as they are formulated according to the available choices. In this direction, the

system Hippalus [70] that implements the preference-aware extension of faceted search introduced in [96] allows the user to rank the facets, the facet terms and the objects according to *preferences* defined directly by the user, through right-click triggered actions.

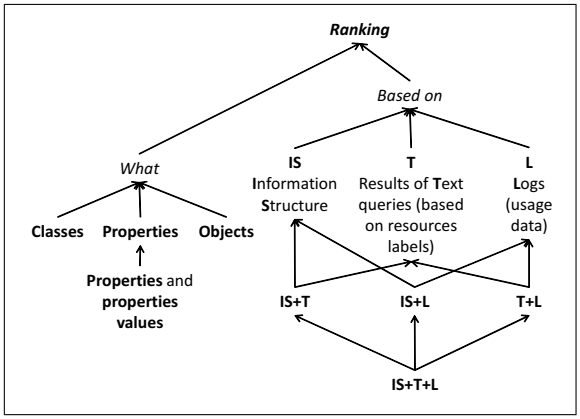

Fig. 13 Taxonomy of ranking methods for exploration approaches for O-O information spaces

System	Ranking of	Based on
BrowseRdf [66]	P	IS
VisiNav [33]	PV	IS
Faceted Wikipedia [32]	P	IS
MediaFaces [78]	P	IS+L
Faceted Data Explorer [23]	C	IS+T
Facetedpedia[54]	CP	IS
ebay	P	L
Fuzzy view based search[38]	CP	IS
Hippalus[70]	COPV	IS + UP

C: Classes	**IS**: Information Structure
O: Objects	**L**: Logs
P: Properties	**T** : Text queries
V: Property-values	**UP** : User Preferences

Table 6 Exploration approaches and their ranking capabilities

Additionally, a number of approaches that try to automatically present the most "useful" facets and zoom-points according to various other criteria have been proposed. Such criteria include *set-cover ranking* of indexed objects [20], *interestingness* over a number of criteria [21], or use *collaborative* [52] and *content* filtering [93] to rank facet-values pairs. Minimum-effort driven navigational techniques for enterprise databases, that rapidly drill down to the most prominent tuples are described in [80]. In the same manner, but for zoom-points, [47] proposes a system for faceted navigation using a cost model of user navigation. A browsing-oriented approach for facet ranking and grouping of facets and their values according to different intuitions and metrics is provided in [97]. Finally, various other works are discussed in [67]. In general we observe an increasing interest on ranking methods

which is justified by the increase of datasets' sizes; the bigger the dataset's size is, the more important (and difficult) the problem of ranking becomes.

Apart from ranking methods, for aiding the users to find the sought facet (either corresponding to a class, property, or metadata element), *keyword search* is usually supported by several systems (listed in §6.3.3). In addition, *autocompletion* services during typing are very convenient and are supported by some systems (e.g. SPARKLIS).

6.3.3 Initial States and Exploitation of Namespaces

Initial States. As regards the initial states in most systems the topmost elements (classes, categories and properties) are shown, allowing the user to expand to more specialized classes or properties on demand, as in BrowseRDF, Camelis2, /facet, Facete, GRQL, MuseumFinland, Ontogator, Rhizomer, and SPARKLIS. In some systems *ranking* is used (recall ranking was discussed in §6.3.2), specifically ranked facets are used in Faceted Wikedia and MediaFaces. In most systems *keyword searching* is (or can be) the first step of the process, as in Facete, Faceted Data Explorer, Faceted Wikipedia, Museum Finland, Ontogator, Parallax, Rhizomer, SemFacet, SPARKLIS, VisiNav. Moreover various *graphical representations* are also used in a few cases, e.g. facet graphs in gFacet and tag clouds in Humboldt. For elements with geocoordinates some systems provide view maps, e.g. the system Facete.

Namespaces. As mentioned in §4, the existence of more than one namespaces can be exploited for setting extra filtering conditions throughout the interaction. Although simple and not hard to implement, most systems do not allow the user to select the desired namespace(s). Some papers just mention that they filter out the elements of "internal" namespaces (RDF, RDFS and OWL). We should mention however that it was not feasible to check this functionality in various systems first because some of them are not online, and second because the demos that are usually deployed provide access to small in size datasets and the prefixes of the URIs are not shown. However, namespace management is definitely something useful and modern systems should exploit them. According to some measurements that we have performed over datasets available in the LOD cloud in 2016[12], we have seen that each dataset's URIs contain on average 212 different prefixes.

6.3.4 Implementation

In Section 5 we described how the core model can be implemented using the current features of SPARQL. Here we describe the technologies used by the systems mentioned earlier.

Front-End. The front-end is mainly based on client side technologies (HTML, CSS and JavaScript).

Server-Side Data Management. The server-side usually relies on a framework for managing semantic data. As regards the query language most systems use SPARQL as provided by the used triplestore (Sesame, Virtuoso, RDFox, etc). Parallax, that provides access to Freebase, uses MQL[13], Camelis2 uses LISQL, while

[12] Measurements performed b Michalis Mountantonakis

[13] `http://mql.freebaseapps.com/`

/facet and Ontogator (Ontogator is one of the components of MeseumFinland) use Prolog. GRQL uses RQL [46], while mspace uses RDQL[14]. Special indexes are used by some systems: Ontogator uses a memory-resident prefix label index, VisiNav is based on dedicated indexes, while Faceted Wikipedia relies on full text retrieval system and dedicated tree indexes. Finally, some systems like Faceted Wikipedia and SemFacet also have an IR system (Lucene) for supporting flat keyword search (useful for the initial state of the interaction).

Server-Side Programming Languages and Frameworks. The server side relies on Java technologies (GRQL, Facete, Longwell, SemFacet, Ontogator, VisiNav), Ruby on Rails (BrowseRDF), or C# (Humboldt). Camelis2 is based on OCaml (a PL supporting functional, imperative and object-oriented features). Faceted Data Explorer (which is based on Virtuoso cluster edition) also uses application code in SQL procedures. Ontogator apart from Java it uses Prolog, and MuseumFinland also uses the Cocoon framework. Several systems use XML and XSLT transformations to generate XHTML. It is worth noting that some systems, like gFacet and SPARKLIS do not have a dedicated server side component for the exploration, meaning that the implementation is at client side (JS), i.e. the client sends directly queries to SPARQL endpoints (SPARKLIS uses JS, gFacet is implemented using Adobe Flex).

6.3.5 Graphical Query Formulators and Linked Data Browsers

There are some tools, usually called *graphical query formulators* or *visual query systems*, that offer interactive and graphical methods to formulate SPARQL or SQL queries like [31] which is based on extended filter-flow graphs, NITELIGHT [81], SEWASIE [15] and others. Although related, such tools provide only an interactive process for formulating queries; they do not offer an interactive process for query formulation in accordance with the actual contents of the underlying dataset. Instead, the surveyed in this paper systems offer an *information seeking* process for satisfying *recall-oriented* information needs in the form of a sequence of browsing actions that includes inspection or overview of the dataset. None of the systems we surveyed is geared towards graphical query formulation only.

Linked Data browsers allow users to navigate between data sources by following links expressed at published RDF triples. Browsers that support such a functionality include Tabulator [10], Disco[15], Marbles[16](provenance of resources is also indicated) and the OpenLink Data Explorer[17], a browser extension for viewing Data Sources associated with Web Pages. A common characteristic of these browsers is that they support *one resource-to-one resource* transitions (navigation along in/outgoing links) while at each step of browsing only the directly connected resources are accessible. Thus these approaches do not exploit the structuring of data for providing exploration services. However, Tabulator allows the user to formulate queries in a query-by-example style: the user can highlight a formulated pattern of interest, query for any similar patterns and further analyze the results through maps, timeline or other conventional data presentation methods.

[14] `https://www.w3.org/Submission/RDQL/`
[15] http://sites.wiwiss.fu-berlin.de/suhl/bizer/ng4j/disco/
[16] http://beckr.org/marbles
[17] http://ode.openlinksw.com

7 Evaluating Exploration Approaches

An important aspect of information systems, which is not well studied for exploratory search systems, is their evaluation and metric-based comparison. Below we focus on the creation of evaluation tasks and metrics, and categorize various systems based on their evaluation criteria. Finally, we discuss information retrieval (IR) metrics that could be exploited for evaluating exploratory search systems.

The first step in the evaluation of a system is the creation of the appropriate evaluation tasks, which can be either user-based or non user-based ones. A number of works provide guidelines for creating new exploratory tasks specifically designed for faceted search and user-based evaluations. For example [53] summarizes that the desired characteristics of exploratory tasks include low initial topic familiarity, multiple items consideration and ambiguity to the final answers. In the same manner, the work described in [100] recommends that exploratory tasks should be open-ended and oriented towards learning and investigation, targeting multiple items and including multiple facets, but in a clear context. On the other hand, a number of initiatives from the IR world like TREC[18] and CLEF[19], provide relevant tasks (e.g. TREC's Session track[20], TREC's Interactive track[21], CLEF-IP track[22]) that could be adapted and used for evaluating exploratory systems.

After the creation of the tasks, the systems at hand can now be compared or evaluated according to a number of metrics. As shown in Table 7, most systems are evaluated using metrics from the Human Computer Interaction (HCI) domain; either qualitative (e.g. Task Success, Evaluation Forms that users have to fill-in) or quantitative (e.g. number of clicks/actions, time to make an action or finish a task, data gathered from an eye-tracker, etc.). Only a few systems (e.g. the Hippalus system [70]) exploit metrics proposed and used in the IR domain, since there are no standard datasets that can be easily exploited by exploratory systems.

In this manner, below we describe a number of metrics that could be used in the evaluation of exploratory systems and capture different aspects of the search and exploration process. We dissever the available metrics in four categories (shown in Table 8), based on a recent survey on IR metrics [14].

The first group of metrics assumes no user model (UM) and binary relevance. As a result, instead of a ranked list of results, these metrics return the set of relevant results. This category includes the traditional Cranfield metrics of *Precision* and *Recall*, and their variations and combinations. For example, for precision-oriented tasks *R-Precision* is a widely used metric, while *Normalized Recall (Rnorm)* and *PRES* [57] are used for recall-oriented and exploratory tasks. Metrics like *Average Precision (AP)*, *Mean Average Precision (MAP)* and *F-Measure* consider both precision and recall. Additionally, *F-Measure* can be adjusted to be either more recall-oriented or more precision-oriented.

The second group of metrics provides simple UMs for user behaviour and assumes a ranked list of results, which is more relevant for tasks related to web searching or question answering. For example the *Expected Search Length (ESL)*

[18] http://trec.nist.gov/
[19] http://www.clef-initiative.eu
[20] http://ir.cis.udel.edu/sessions/
[21] http://trec.nist.gov/data/interactive.html
[22] http://www.clef-initiative.eu/track/clefip

System	Metrics	
	HCI	**IR**
BrowseRDF	T, F	-
Elastic Lists	S, T, F, E, C	-
FleXplorer & Mitos	S, T, F, C	-
Flamenco	S, T, F	-
Fuzzy view based search	F	-
gFacet	S	-
Hippalus	S, T, F, C	R, P, AP
IOS/XSearch	F, T	-
Rhizomer	S, T	-

C: Clicks & Log info
E: Eye-Tracking
F: Evaluation Forms
S: Task Success
T: Timings
AP: Average Precision
P: Precision
R: Recall

Table 7 Evaluation metrics used by systems

Type		Metric
Traditional		*Precision*, *Recall*, *R-precision*, *AP, MAP*, *F-Measure*, *Rnorm*, *PRES* [57]
User Models	Simple	*ESL* [19], *Utility based Precision & Recall* [76], *GAP* [77], *RBR* [64], *DCG & nDCG* [43], *nDCG for facets & nrDCG* [86], *ERR* [16], *EBU* [104], *Bayes procedure & click data* [14], *Twist* [30]
	Novelty & Diversity	*Subtopics based Precision & Recall* [105], *Intent-aware metrics* [3], *Intent-aware ERR* [16], α-*nDCG* [17]
	Session -based	*nsDCG* [44], *Session-based Recall, Precision & AP* [45] *Probabilistic multi-session metrics* [102]
Other		*Stream-based metrics* [8],IR & HCI blended metrics [101] *Statistical methods FA* [91], *Statistical methods SEM* [65]

Table 8 Categories of evaluation metrics

metric [19], assumes that the user walks down a ranked list of results, observing each result until a stop point. A redefinition of the *Precision* and *Recall* using the above user model and the utility of each result is described in [76], while *Graded Average Precision (GAP)* [77] assumes that the user is only interested for results with a ranking value over a specific threshold. *Rank Biased Precision (RBR)* [64] assumes that the user only examines a specific number of results, the *Discounted Cumulated Gain* family metrics [43] accumulates the gain of a results set from the top, with the gain of each result discounted at lower ranks, *Expected Reciprocal Rank (ERR)* [16] assumes a user that accumulates utility by stepping down the ranked list and decides whether to stop or not according to the accumulated utility, while [14] simulates user behaviour by using click data and a Bayes procedure. Finally, the *Twist* measure described in [30], evaluates the effectiveness of a system according to the required user effort for retrieving the desired information.

The third group of metrics uses more advanced UMs and consists of two families. The first one takes into account the novelty and diversity of the results. For example, Recall and Precision have been generalized to include subtopics [105] (i.e. utility is increased when the user gets an answer with a new topic), intent-aware family measures [3] assume a probability distribution over subtopics, intent-aware ERR [16] computes the weighted average of ERR over intents and α-nDCG [17] penalizes duplicates. The second family of metrics assumes a session-based inter-

action, which is a distinct feature of faceted exploratory systems. Recall that these systems offer a dialogue between the user and the system, in such a way that the response of the system (e.g. answer, branch shown, etc.), does not depend only on the current user request (e.g. query, selector click), but also on his previous requests and session history in general. As a result session-based metrics, like nsDCG [44], generalizations of the traditional measures of *Precision*, *Recall* and *Average Precision* for sessions [45], as well as theoretical probabilistic frameworks that take into consideration user interactions over multi-session ordered lists [102] are rather important for evaluating exploratory systems.

The last category of metrics includes works like the one described in [8], where the usage of a system is represented as a stream of results and its performance is studied based on time and usage, while [101] provide an evaluation framework by blending IR and HCI design. Finally, statistical methods for examining interrelationships between multiple evaluation criteria, such as Factor Analysis (FA) and Structural Equation Models (SEM) are provided in [91] and [65] respectively.

8 Concluding Remarks

Since the number of RDF datasets constantly increases, the value of general purpose methods for browsing and effectively exploring such datasets increases as well. The exploration of such datasets can aid a plethora of tasks that rely on decision making and range a wide spectrum of activities of our life from the medical domain to the academic and research domain. This paper introduced aspects and criteria for surveying the works in the area of browsing and exploring such datasets with emphasis on session-based interaction schemes. It identified different kinds of information needs, information structures and configuration requirements. Apart from clarifying the tasks at hand, and the quite diverse terminology that is used in the literature, the paper provided a generalization of the main exploration/browsing approaches using a small but precise model comprising states and transitions (of various objectives) between states. The model is suitable for comparing the existing approaches and can be used also as a guide for designing, implementing and evaluating new systems, APIs (Application Programming Interfaces) or protocols. Although we focused on approaches applicable to RDF/S datasets, the analysis and the introduced model can be applied to any object-oriented conceptual modeling approach that supports classes, inter-class associations, specialization/generalization hierarchies (among classes and among inter-class associations) and instantiation. Subsequently, the paper described more than 30 systems in total (11 of them for single entity and 21 for multi entity type datasets like RDF/S datasets) according to the introduced aspects and the core model. Finally, the paper focused on HCI and IR methods for evaluating exploration approaches and discussed what kind of evaluation results are usually reported in the literature.

Promising directions for further work and research include novel implementation approaches focusing on scalability for enabling the exploration of big data, also harmonized with the distributed nature of Linked Open Data. Another promising direction is the investigation of possible refinements of the interaction for datasets that include geographic and social data, as well as refinements of the interaction according to various quality aspects of the data, like completeness, uncertainty, provenance issues, and trust in general. Finally, a rather important direction is the

creation and standardization of evaluation methods, tasks and datasets, specifically designed for exploratory search systems.

References

1. SIMILE: Longwell RDF Browser (2003-2005). `http://simile.mit.edu/wiki/Longwell/`.
2. SPARQL Query Language for RDF, W3C Candidate Recommendation, 15 January 2008.
3. R.h Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying Search Results. In *Procs of the Second ACM Intern. Conf. on Web Search and Data Mining (WSDM'09)*, pages 5–14, New York, NY, USA, 2009. ACM.
4. P. Allard and S. Ferré. Dynamic Taxonomies for the Semantic Web. In *Procs of the 19th Intern. Conf. on Database and Expert Systems Application (DEXA)*, 2008.
5. M. Arenas, B. Cuenca Grau, E. Kharlamov, S. Marciuska, and D. Zheleznyakov. Faceted Search over Ontology-Enhanced RDF Data. In *Procs of the 23rd ACM Intern. Conf. on Information and Knowledge Management*, pages 939–948. ACM, 2014.
6. M. Arenas, G. Gottlob, and A. Pieris. Expressive languages for querying the semantic web. In *Procs of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS'14, Snowbird, USA, June, 2014*, pages 14–26, 2014.
7. N. Athanasis, V. Christophides, and D. Kotzinos. Generating On the Fly Queries for the Semantic Web: The ICS-FORTH Graphical RQL Interface (GRQL). In *Intern. Semantic Web Conf. (ISWC)*, 2004.
8. L. Azzopardi. Usage Based Effectiveness Measures: Monitoring Application Performance in Information Retrieval. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 631–640, New York, USA, 2009. ACM.
9. O. Ben-Yitzhak, N. Golbandi, N. Har'El, R. Lempel, A. Neumann, S. Ofek-Koifman, D. Sheinwald, E. Shekita, B. Sznajder, and S. Yogev. Beyond Basic Faceted Search. In *Procs of the Intern. Conf. on Web Search and Web Data Mining (WDSM'08)*, pages 33–44, 2008.
10. T. Berners-lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets. Tabulator: Exploring and Analyzing Linked Data on the Semantic Web. In *Procs of the 3rd Intern. Semantic Web User Interaction Workshop (SWUI06)*, 2006.
11. C. Bizer, T. Heath, and T. Berners-Lee. Linked Data-The Story so Far. *Intern. Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.
12. A. Broder. A Taxonomy of Web Search. *SIGIR Forum*, 36(2):3–10, 2002.
13. J. M. Brunetti, R. Garcia, and S. Auer. From overview to facets and pivoting for interactive exploration of semantic web data. *Intern. Journal on Semantic Web and Information Systems*, 9(1):1–20, 2013.
14. B. Carterette, E. Kanoulas, and E. Yilmaz. Evaluating Web Retrieval Effectiveness. In *Web Search Engine Research*, pages 105–137. Emerald Books, 2012.
15. T. Catarci, T. Di Mascio, E. Franconi, G. Santucci, and S. Tessaris. An Ontology Based Visual Tool for Query Formulation Support. In *On The Move to Meaningful Internet Systems 2003: OTM 2003Workshops*, pages 32–33. Springer Berlin / Heidelberg, 2003.
16. O. Chapelle, S. Ji, C. Liao, E. Velipasaoglu, L. Lai, and S.-L. Wu. Intent-Based Diversification of Web Search Results: Metrics and Algorithms. *Information Retrieval*, 14(6):572–592, 2011.
17. C. L. A. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and Diversity in Information Retrieval Evaluation. In *Procs of the 31st Annual Intern. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR'08)*, pages 659–666, New York, NY, USA, 2008. ACM.
18. E. C. Clarkson, S. B. Navathe, and J. D. Foley. Generalized Formal Models for Faceted User Interfaces. In *JCDL '09: Procs of the 9th ACM/IEEE-CS joint conference on Digital libraries*, pages 125–134, New York, NY, USA, 2009. ACM.
19. W. S. Cooper. Expected Search Length: A Single Measure of Retrieval Effectiveness Based on the Weak Ordering Action of Retrieval Systems. In *American Documentation*, pages 30–41, 1968.
20. W. Dakka, P. Ipeirotis, and K. R. Wood. Automatic Construction of Multifaceted Browsing Interfaces. In *Procs of CIKM'05*, pages 768–775, Nov. 2005.
21. D. Dash, J. Rao, N. Megiddo, A. Ailamaki, and G. Lohman. Dynamic Faceted Search for Discovery-driven Analysis. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 3–12, 2008.

22. O. Erling and I. Mikhailov. RDF Support in the Virtuoso DBMS. In *Procs of 1st Conf. on Social Semantic Web*, 2007.
23. O. Erling and I. Mikhailov. Faceted Views over Large-Scale Linked Data. In *Procs of the WWW2009 Workshop on Linked Data on the Web*, 2009.
24. P. Fafalios, I. Kitsos, Y. Marketakis, C. Baldassarre, M. Salampasis, and Y. Tzitzikas. Web Searching with Entity Mining at Query Time. In *Multidisciplinary Information Retrieval*, pages 73–88. Springer Berlin Heidelberg, 2012.
25. P. Fafalios and Y. Tzitzikas. X-ENS: Semantic Enrichment of Web Search Results at Real-time. In *Procs of the 36th intern. ACM SIGIR conference on Research and development in information retrieval*, pages 1089–1090. ACM, 2013.
26. P. Fafalios and Y. Tzitzikas. Post-Analysis of Keyword-based Search Results using Entity Mining, Linked Data and Link Analysis at Query Time. In *2014 IEEE Eighth Intern. Conf. on Semantic Computing (ICSC 2014)*, California, USA, June 2014. IEEE.
27. P. Fafalios and Y. Tzitzikas. SPARQL-LD: A SPARQL Extension for Fetching and Querying Linked Data. In *The Semantic Web–ISWC 2015 (Posters & Demonstrations Track)*, Bethlehem, Pennsylvania, USA, 2015.
28. S. Ferré. Conceptual Navigation in RDF Graphs with SPARQL-Like Queries. *Formal Concept Analysis*, pages 193–208, 2010.
29. S. Ferré. Sparklis: a SPARQL Endpoint Explorer for Expressive Question Answering. In *The Semantic Web–ISWC 2014*. Springer, 2014.
30. N. Ferro, G. Silvello, A. Keskustalo, H. abd Pirkola, and K Järvelin. The Twist Measure for IR evaluation: Taking User's Effort into Account. *Journal of the Association for Information Science and Technology*, 20.
31. F. Haag, S. Lohmann, S. Bold, and T. Ertl. Visual SPARQL Querying Based on Extended Filter/Flow Graphs. In *Procs of the 2014 Intern. Working Conf. on Advanced Visual Interfaces*, AVI '14, pages 305–312, New York, NY, USA, 2014. ACM.
32. R. Hahn, C. Bizer, C. Sahnwaldt, C. Herta, S. Robinson, M. Bürgle, H. Düwiger, and U. Scheel. Faceted Wikipedia Search. In *Business Information Systems*, volume 47 of *Lecture Notes in Business Information Processing*, pages 1–11. 2010.
33. A. Harth. VisiNav: Visual Web Data Search and Navigation. In *Procs of the 20th Intern. Conf. on Database and Expert Systems Applications (DEXA '09)*, 2009.
34. M. A. Hearst. UIs for Faceted Navigation: Recent Advances and Remaining Open Problems. In *Workshop on Computer Interaction and Information Retrieval, HCIR'08*, 2008.
35. P. Heim, T. Ertl, and J. Ziegler. Facet Graphs: Complex Semantic Querying Made Easy. In *The Semantic Web: Research and Applications*, pages 288–302. Springer, 2010.
36. B. Heitmann, S. Kinsella, C. Hayes, and S. Decker. Implementing Semantic Web Applications: Reference Architecture and Challenges. *5th Intern. Workshop on Semantic Web-Enabled Software Engineering*, 2009.
37. M. Hildebrand, J. Ossenbruggen, and L. Hardman. /facet: A Browser for Heterogeneous Semantic Web Repositories. In *Procs of ISWC '06*, 2006.
38. M. Holi and E. Hyvönen. Fuzzy View-Based Semantic Search. In *ASWC*, 2006.
39. W. Hua, Y. Song, H. Wang, and X. Zhou. Identifying Users' Topical Tasks in Web Search. In *Procs of the Sixth ACM Intern. Conf. on Web Search and Data Mining*, WSDM '13, pages 93–102, New York, NY, USA, 2013. ACM.
40. D. Huynh. Nested Faceted Browsing, 2010.
41. D. Huynh and D. Karger. Parallax and Companion: Set-based Browsing for the Data Web. *(submitted to WWW '09)*, 2009.
42. E. Hyvönen, E. Mäkelä, M. Salminen, A. Valo, K. Viljanen, S. Saarela, M. Junnila, and S. Kettula. MUSEUMFINLAND - Finnish Museums on the Semantic Web. *Journal of Web Semantics*, 3(2-3):224–241, 2005.
43. K. Järvelin and J. Kekäläinen. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Transactions Information Systems*, 20(4):422–446, October 2002.
44. K. Järvelin, S. L. Price, L. M. L. Delcambre, and M. L. Nielsen. Discounted Cumulated Gain Based Evaluation of Multiple-Query IR Sessions. In *ECIR*, pages 4–15, 2008.
45. E. Kanoulas, B. Carterette, P. Clough, and M. Sanderson. Evaluating Multi-Query Sessions. In *Procs of the 34th Intern. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR'11)*, pages 1053–1062, 2011.
46. Gregory Karvounarakis, Sofia Alexaki, Vassilis Christophides, Dimitris Plexousakis, and Michel Scholl. Rql: a declarative query language for rdf. In *Proceedings of the 11th international conference on World Wide Web*, pages 592–603. ACM, 2002.

47. A. Kashyap, V. Hristidis, and M. Petropoulos. FACeTOR: Cost-Driven Exploration of Faceted Query Results. In *Procs of CIKM'10*, pages 719–728. ACM, 2010.
48. J. Kehrer and H. Hauser. Visualization and visual analysis of multifaceted scientific data: A survey. *Visualization and Computer Graphics, IEEE Transactions on*, 19(3):495–513, 2013.
49. S. Khurso and A.M. Tjoa. Fulfilling the Needs of a Metadata Creator and Analyst. In *The Past and Future of Information Systems: 1976-2006 and Beyond*, volume 214, pages 177–188. Springer Boston, 2006.
50. I. Kitsos, K. Magoutis, and Y. Tzitzikas. Scalable Entity-based Summarization of Web Search Results using MapReduce. *Distributed and Parallel Databases*, pages 1–42, 2013.
51. G. Kobilarov and I. Dickinson. Humboldt:Exploring linked data. In *Linked Data on the Web Workshop at WWW2008*, Beijing, China, 2008.
52. J. Koren, Y. Zhang, and X. Liu. Personalized Interactive Faceted Search. In *WWW'08: Procs of the 17th Intern. Conf. on World Wide Web*, pages 477–486, New York, NY, USA, 2008. ACM.
53. B. Kules and R. Capra. Creating Exploratory Tasks for a Faceted Search Interface. In *in the Workshop on Computer Interaction and Information Retrieval, HCIR 2008*, pages 18–21e, 2008.
54. C. Li, N. Yan, S. B. Roy, L. Lisham, and G. Das. Facetedpedia: Dynamic Generation of Query-Dependent Faceted Interfaces for Wikipedia. In *WWW*, pages 651–660, 2010.
55. Y. Li. Exploring the Relationships Between Work Task and Search Task in Information Search. *Journal of the American Society for Information Science and Technology*, 60(2):275–291, 2009.
56. C. Lucchese, S. Orlando, R. Perego, F. Silvestri, and G. Tolomei. Identifying Task-based Sessions in Search Engine Query Logs. In *Procs of the 4th ACM Intern. Conf. on Web Search and Data Mining*, WSDM '11, pages 277–286, New York, NY, USA, 2011. ACM.
57. W. Magdy and G. J. F. Jones. PRES: A Score Metric for Evaluating Recall-oriented Information Retrieval Applications. In *Proceeding of the 33rd Intern. ACM SIGIR Conf. on Research and Development in Information Retrieval, SIGIR 2010, Geneva, Switzerland, July 19-23, 2010*, pages 611–618, 2010.
58. E. Mäkelä, E. Hyvönen, and S. Saarela. Ontogator - A Semantic View-Based Search Engine Service for Web Applications. In *Procs of ISWC '06*, pages 847–860, 2006.
59. A. Mallea, M. Arenas, A. Hogan, and A. Polleres. On blank nodes. In *The Semantic Web–ISWC 2011*, pages 421–437. Springer, 2011.
60. N. Manolis and Y. Tzitzikas. Interactive Exploration of Fuzzy RDF Knowledge Bases. In *Procs of ESWC'11*, 2011.
61. G. Marchionini. Exploratory Search: from Finding to Understanding. *Communications of the ACM*, 49(4):41–46, April 2006.
62. M. Mazzieri. A Fuzzy RDF Semantics to Represent Trust Metadata. In *1st Workshop on Semantic Web Applications and Perspectives (SWAP2004)*, 2004.
63. J. Ming, Y. Jun, G. Siyu, H. Jiawei, H. Xiaofei, Z. Wei Vivian, and C. Zheng. Learning Search Tasks in Queries and Web Pages via Graph Regularization. In *Procs of the 34th Intern. ACM SIGIR Conf. on Research and Development in Information Retrieval*, SIGIR '11, pages 55–64, New York, NY, USA, 2011. ACM.
64. A. Moffat and J. Zobel. Rank-Biased Precision for Measurement of Retrieval Effectiveness. *ACM Transactions Information Systems*, 27(1):2:1–2:27, December 2008.
65. H. L. O'Brien, E. G. Toms, E. K. Kelloway, and E. Kelley. Developing and Evaluating a Reliable Measure of User Engagement. *Procs of the American Society for Information Science and Technology*, 45(1):1–10, April 2008.
66. E. Oren, R. Delbru, and S. Decker. Extending Faceted Navigation for RDF Data. In *Procs of ISWC '06*, 2006.
67. P. Papadakos. *Interactive Exploration of Multi-Dimensional Information Spaces with Preference Support*. PhD thesis, University of Crete, November 2013.
68. P. Papadakos, S. Kopidaki, N. Armenatzoglou, and Y. Tzitzikas. Exploratory Web Searching with Dynamic Taxonomies and Results Clustering. In *ECDL '09: Procs of the 13th European Conf. on Digital Libraries*, September 2009.
69. P. Papadakos, S. Kopidaki, N. Armenatzoglou, and Y. Tzitzikas. On Exploiting Static and Dynamically-mined Metadata for Exploratory Web Searching. *Knowledge and Information Systems*, 2011.
70. P. Papadakos and Y. Tzitzikas. Hippalus: Preference-enriched Faceted Exploration. In *EDBT/ICDT Workshops*, pages 167–172, 2014.

71. P. Papadakos and Y. Tzitzikas. Comparing the Effectiveness of Intentional Preferences versus Preferences over Specific Choices: A User Study. *Internation Journal of Information and Decision Sciences*, 2015. (to appear).
72. H. Paulheim and F. Probst. Ontology-Enhanced User Interfaces: A Survey. *Int. J. Semantic Web Inf. Syst.*, 6(2):36–59, 2010.
73. E. Pietriga, C. Bizer, D. Karger, and R. Lee. Fresnel - A Browser-Independent Presentation Vocabulary for RDF. In *Procs of the Second InterN. Workshop on Interaction Design and the Semantic Web*, pages 158–171. Springer, 2006.
74. J. Polowinski. Widgets for Faceted Browsing. In *Procs of the Symposium on Human Interface 2009 on Conf.Universal Access in Human-Computer Interaction. Part I*, pages 601–610, 2009.
75. B. Qarabaqi and M. Riedewald. User-driven refinement of imprecise queries. In *IEEE 30th Intern. Conf. on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014*, pages 916–927, 2014.
76. S. Robertson. A New Interpretation of Average Precision. In *Procs of the 31st Annual Intern. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR'08)*, pages 689–690, New York, NY, USA, 2008. ACM.
77. S. E. Robertson, E. Kanoulas, and E. Yilmaz. Extending Average Precision to Graded Relevance Judgments. In *Procs of the 33rd Intern. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR'10)*, pages 603–610, New York, NY, USA, 2010. ACM.
78. Z. Roelof, B. Sigurbjornsson, R. Adapala, L. Garcia Pueyo, A. Katiyar, K. Kurapati, M. Muralidharan, S. Muthu, V. Murdock, A. Ng, P.and Ramani, A. Sahai, S. T. Sathish, H. Vasudev, and U. Vuyyuru. Faceted Exploration of Image Search Results. In *WWW'10: Procs of the 19th intern. conference on World wide web*, 2010.
79. D. E. Rose and D. Levinson. Understanding User Goals in Web Search. In *Procs of the 13th intern. conference on World Wide Web*, WWW '04, 2004.
80. S. B. Roy, H. Wang, G. Das, U. Nambiar, and M. Mohania. Minimum-Effort Driven Dynamic Faceted Search in Structured Databases. In *Procs of CIKM'08*, pages 13–22, 2008.
81. A. Russell, P. R. Smart, D. Braines, and N. R. Shadbolt. NITELIGHT: A Graphical Tool for Semantic Query Construction. In *Semantic Web User Interaction Workshop (SWUI 2008)*, April 2008.
82. G. M. Sacco. Efficient Implementation of Dynamic Taxonomies. Technical report, Univ. di Torino, 2004.
83. G. M. Sacco and Y. Tzitzikas. *Dynamic Taxonomies and Faceted Search: Theory, Practice, and Experience*. Springer, 2009.
84. G. M. Sacco and Y. Tzitzikas. *Dynamic Taxonomies and Faceted Search: Theory, Practise and Experience*. Springer, 2009. ISBN = 978-3-642-02358-3.
85. M. C. Schraefel, D. A. Smith, A. Owens, A. Rusell, C. Harris, and M.L. Wilson. The Evolving mSpace Platform: Leveraging the Semantic Web on the Trial of the Memex. *In Procs of Hypertext 2005*, pages 174–183, 2005.
86. A. Schuth and M. Marx. Evaluation Methods for Rankings of Facetvalues for Faceted Search. In *Procs of the Second Intern. Conf. on Multilingual and Multimodal Information Access Evaluation (CLEF'11)*, pages 131–136, Berlin, Heidelberg, 2011. Springer-Verlag.
87. C. Stadler, M. Martin, and S. Auer. Exploring the Web of Spatial Data with Facete. In *Procs of the Companion Publication of the 23rd Intern. Conf. on World Wide Web Companion*, WWW Companion '14, pages 175–178, Republic and Canton of Geneva, Switzerland, 2014. Intern. World Wide Web Conf.s Steering Committee.
88. M. Stefaner, T. Urban, and M. Seefelder. Elastic Lists for Facet Browsing and Resource Analysis in the Enterprise. In *Procs of the 19th Intern. Conf. on Database and Expert Systems Application (DEXA'08)*, pages 397–401, 2008.
89. A. Stolz and M. Hepp. An adaptive faceted search interface for structured product offers on the web. In *Procs of the 4th Intern. Workshop on Intelligent Exploration of Semantic Data (IESD 2015), At Bethlehem, Pennsylvania, USA*, October 2015.
90. C. Strubulis, G. Flouris, Y. Tzitzikas, and M. Doerr. A case study on propagating and updating provenance information using the cidoc crm. *Intern. Journal on Digital Libraries*, 15(1):27–51, 2014.
91. E. G. Toms, H. L. O'Brien, R. W. Kopak, and L. Freund. Searching for Relevance in the Relevance of Search. In *CoLIS*, pages 59–78, 2005.
92. D. Tunkelang. *Faceted Search*. Morgan & Claypool, 2009.

93. M. Tvarožek, M. Barla, G. Frivolt, M. Tomša, and M. Bieliková. Improving Semantic Search Via Integrated Personalized Faceted and Visual Graph Navigation. In *SOFSEM*, volume 4910 of *Lecture Notes in Computer Science*, pages 778–789. Springer, 2008.
94. Y. Tzitzikas, N. Armenatzoglou, and P. Papadakos. FleXplorer: A Framework for Providing Faceted and Dynamic Taxonomy-Based Information Exploration. In *Procs of 19th Intern. Conf. on Database and Expert Systems Application (DEXA'08)*, pages 392–396, 2008.
95. Y. Tzitzikas, N. Bailly, P. Papadakos, N. Minadakis, and G. Nikitakis. Species identification through preference-enriched faceted search. In *Procs of the 9th Metadata and Semantics Research Conf. (MTSR '15), Manchester, UK*, September 2015.
96. Y. Tzitzikas and P. Papadakos. Interactive Exploration of Multi-Dimensional and Hierarchical Information Spaces with Real-time Preference Elicitation. *Fundamenta Informaticae*, 20:1–42, 2012.
97. A. Wagner, G. Ladwig, and T. Tran. Browsing-Oriented Semantic Faceted Search. In *DEXA (1)*, pages 303–319, 2011.
98. R. W. White, B. Kules, S. M. Drucker, and M. C. Schraefel. Supporting Exploratory Search, Introduction, Special Issue, Communications of the ACM. *Communications of the ACM*, 49(4):36–39, April 2006.
99. Ryen W White and Resa A Roth. Exploratory search: beyond the query-response paradigm (synthesis lectures on information concepts, retrieval & services). *Morgan and Claypool Publishers*, 3, 2009.
100. B. M. Wildemuth and L. Freund. Assigning Search Tasks Designed to Elicit Exploratory Search Behaviors. In *Procs of the Symposium on Human-Computer Interaction and Information Retrieval*, HCIR '12, pages 4:1–4:10, New York, NY, USA, 2012. ACM.
101. M. L. Wilson and M. C. Schraefel. Bridging the Gap: Using IR Models for Evaluating Exploratory Search Interfaces. In *SIGCHI 2007 Workshop on Exploratory Search and HCI*. ACM, April 2007.
102. Y. Yang and A. Lad. Modeling Expected Utility of Multi-session Information Distillation. In *Procs of the 2nd Intern. Conf. on Theory of Information Retrieval: Advances in Information Retrieval Theory (ICTIR'09)*, pages 164–175, Berlin, Heidelberg, 2009. Springer-Verlag.
103. K.-P Yee, K. Swearingen, K. Li, and M. Hearst. Faceted Metadata for Image Search and Browsing. *In Procs of the SIGCHI Conf. on Human factors in Computing Systems*, pages 401–408, 2003.
104. E. Yilmaz, M. Shokouhi, N. Craswell, and S. Robertson. Expected Browsing Utility for Web Search Evaluation. In *Procs of the 19th ACM intern. conference on Information and knowledge management (CIKM'10)*, pages 1561–1564, 2010.
105. C. X. Zhai, W. W. Cohen, and J. Lafferty. Beyond Independent Relevance: Methods and Evaluation Metrics for Subtopic Retrieval. In *Procs of the 26th Annual Intern. ACM SIGIR Conf. on Research and Development in Informaion Retrieval (SIGIR'03)*, pages 10–17. ACM, 2003.
106. J. Zhang and G. Marchionini. Evaluation and Evolution of a Browse and Search Interface: Relation Browser++. In *Procs of the national conference on Digital government research (DG. '05)*. Digital Government Society of North America, 2005.