RCAviz: Visualizing and Exploring Relational Conceptual Structures

Emile Muller[1,*], Marianne Huchard[1], Pierre Martin[2], Pascal Poncelet[1] and Arnaud Sallaberry[1,3]

[1]LIRMM, Université de Montpellier, CNRS, Montpellier, France
[2]CIRAD, UPR AIDA, F-34398 Montpellier, France; AIDA, Univ Montpellier, CIRAD, Montpellier, France
[3]AMIS, Université Paul-Valéry Montpellier 3, Montpellier, France

Abstract

Formal Concept Analysis (FCA) aims at classifying a set of objects described by Boolean attributes as a concept lattice. Among the various FCA extensions, Relational Concept Analysis (RCA) supports multiple binary relationships between several sets of objects. Using such relational data, RCA builds interconnected conceptual structures, i.e. concept lattices or AOC-posets. Analyzing data can benefit from exploring these interconnected structures by navigating from one concept to another. Navigation can be performed within the same structure through the classification links or from one concept of a conceptual structure to a concept of another one through one of the relations linking the lattices. This paper presents RCAviz[1], an online tool which aims to support such navigation. Once the user has selected a subset of objects and attributes as a starting point for navigation, RCAviz presents the associated concept and its close neighbors. Each concept includes the objects and attributes it introduces. The user can then navigate, i.e. zoom and pan the current view, and move from one concept to another. Additional views show the previous and next conceptual structures. The history allows the user to browse its navigation.

Keywords

Formal Concept Analysis, Relational Concept Analysis, Visualization, Navigation, Concept Lattice

1. Introduction

Formal Concept Analysis (FCA) aims at classifying a set of objects described by Boolean attributes as a concept lattice [1]. Other conceptual structures may also be produced, such as AOC-poset or Iceberg lattice. FCA and its various extensions enable the extraction of frequent patterns, attribute implications, mutual exclusions within attribute groups, and object similarities. The survey of [2] shows the variety of applications where different knowledge discovery methods based on FCA are successfully used [3].

[1]https://rcaviz.lirmm.fr/

Published in Pablo Cordero, Ondrej Kridlo (Eds.): The 16th International Conference on Concept Lattices and Their Applications, CLA 2022, Tallinn, Estonia, June 20–22, 2022, Proceedings, pp. 133–146.

*Corresponding author.
emile.muller.contact@gmail.com (E. Muller); marianne.huchard@lirmm.fr (M. Huchard); ipierre.martin@cirad.fr (P. Martin); pascal.poncelet@lirmm.fr (P. Poncelet); arnaud.sallaberry@lirmm.fr (A. Sallaberry)
0000-0002-6309-7503 (M. Huchard); 0000-0002-4874-5795 (P. Martin); 0000-0002-8277-3490 (P. Poncelet); 0000-0001-7068-176X (A. Sallaberry)

Among the extensions of FCA, Relational Concept Analysis (RCA) supports multiple binary relations between several sets of objects. As output, RCA builds conceptual structures (one per object set) interconnected by relational attributes, that embed the inter-object relations provided as input. Analyzing multi-relational datasets with RCA may benefit from exploring these interconnected structures by navigating from one concept to another. This navigation is therefore performed within the same conceptual structure, through the classification links, or from one concept of a conceptual structure to a concept of another one, through one of the relations linking the lattices.

FCA and RCA present a strong interest when querying a dataset using a set of attributes. In addition to the objects that respond to the query, they provide the classification of these objects [4, 5, 6]. When some objects respond strictly to the query, these objects are part of the extent of the concept, whose intent is constituted by the query attributes. The sub-concepts of this concept then contain the objects having more attributes than those listed in the query. Conversely, the super-concepts contain the objects having less attributes, corresponding to a constraint release. This latter case is of particular interest to domain experts when the dataset is likely to be incomplete, as constraint release may create hypotheses worth to be investigated. Classification also helps to reformulate the query when the answer set is empty, as it shows the concepts highlighting the partially answering objects.

Several authors investigate the potential offered by the visualization for assisting exploring, querying, and analyzing a classification produced by FCA [7, 6, 8]. For RCA, the most advanced tools propose either a sophisticated visualization of the conceptual structure (e.g. Galicia[1]) or a simple visualization of the conceptual structure with the ability to jump from one structure to another through the relational links (e.g. RCAexplore[2]). These tools lack features enabling full visualization support for gradually navigating between concepts without being overwhelmed by the size of the structure. This paper addresses this need by introducing RCAviz[3], an online tool which aims to support such navigation.

Section 2 briefly introduces FCA and RCA basics, and a small RCA navigation example to illustrate the main features expected for a navigation tool. Section 3 presents the visualization and navigation features of RCAviz. in Section 4, we present related work on FCA and RCA visualization and exploration. Section 5 concludes with a summary of the work and a few perspectives.

2. RCA Navigation at a glance

This section briefly introduces FCA and RCA through an example inspired by the Knomana project [9] which motivated the tool presented in this paper. The Knomana dataset deals with the control of pests using plants in agriculture. We then develop the needs addressed by our proposed tool on a small example.

[1] http://www.iro.umontreal.ca/~galicia/
[2] http://dataqual.engees.unistra.fr/logiciels/rcaExplore/
[3] http://rcaviz.lirmm.fr/

FCA in a nutshell FCA input is a formal context (FC) which includes a binary relation that connects each object with each of the attributes it has. E.g. the FC *Plant* presented in Table 1 describes four plants by their family and a few characteristics like being aromatic or comestible. FCA builds formal concepts on top of the FC and organizes these concepts as a conceptual structure. A formal concept is a maximal group of objects (extent) sharing a maximal group of attributes (intent). For instance, in Table 1, as all the plants of the group $E_9 = \{chromolaenaOdorata\ aspiliaAfricana, ageratumConyzoide, \}$ share all the attributes $I_9 = \{\ asteraceae, aromatic, toxic, evergreen, applicEssentialOil, applicExtract\}$, $CPl_9 = (E_9, I_9)$ is a concept. The conceptual structure reflects group inclusion and highlights concept specialization. The top right part of Figure 1 presents an example of specialization of plant concepts. Concept CPl_{11}, which associates $E_{11} = \{ageratumConyzoide\}$ with its common attributes $I_{11} = I_9 \cup \{antidysenteric\}$ is a subconcept of CPl_9, with $I_{11} \supseteq I_9$ and $E_{11} \subseteq E_9$. This figure also presents relational attributes, specific to RCA (i.e. initiated using the quantifier $\exists$), which are explained here after.

RCA in a nutshell RCA extends FCA to datasets that conform to the entity-relationship model [10]. The top left part of Figure 1 shows such a model presented using the UML syntax: A plant treats (controls) a pest; A pest is found in a country; A country possesses plants. As input, RCA takes a set of FCs and a set of *relational contexts* (RC) connecting objects of two FCs. FCs correspond to the classes of the UML model (i.e. *Plant*, *Pest*, *Country*), while RCs correspond to the associations (i.e. *treats*, *isFoundIn*, *possesses*). Each FC is computed as a conceptual structure, which has incorporated iteratively the object clustering of the other FCs thanks to the *relational attributes*. Object clustering propagation applies when a RC (denoted by r) connects objects of a source FC (denoted by FC_s) to objects of a target FC (denoted by FC_t). E.g. The RC $r = treats$ connects objects of FC $FC_s = Plant$ to objects of FC $FC_t = Pest$. Once a concept C_t is built in FC_t, some objects in FC_s may be connected to objects of C_t using r. Different schemes of connection can be adopted, e.g. the connection of an object o_s of FC_s by r to *at least one* object of C_t, the connection of o_s to *all* objects of C_t, or *half* of o_s connections by r are with some objects of C_t. For example in Figure 1, the concept $CPe13$ groups *aspergillusOchraceus* and *aspergillusParasiticus*, two pests that attack peanuts. In addition, *aspiliaAfricana* and *chromolaenaOdorata* control respectively *aspergillusParasiticus* and *aspergillusOchraceus*. Thus both plants control *at least one* pest of $CPe13$. The selection of the scheme is captured in the relational attributes that include a quantifier (e.g. $\exists$, $\supseteq$, $\forall_{\geq 50\%}$), the relation r and the target concept C_t. The use of the quantifier $\exists$ in Figure 1 conducts to express the relational attribute "treats at least one pest of $CPe13$" as follow: $\exists treats(CPe13)$. Relational attributes are added to the description of the objects to respect FCA principle. E.g. *aspiliaAfricana* and *chromolaenaOdorata* both receive $\exists treats(CPe13)$ as an attribute. During the conceptual structure computation, existing concepts may be enriched by these relational attributes or new concepts may be built to group objects that share some of them. In our example $CPl21$ is built to factorize $\exists treats(CPe13)$. The propagation thus follows the paths and cycles of the entity-relationship model until a fix-point is reached. RCA algorithm and theoretical details can be found in [10].

Table 1
Example of relational dataset made of the Formal Contexts (FC) plants (FC ***Plant***), pests (FC ***Pest***) and countries (FC ***Country***). Plants are described by their family and a few characteristics. Pests are described by their family and entities they attack. Countries are described by their place in a continent. Relational Contexts (RC) connect objects of two FCs. RC ***treats*** connects FC Plant and FC Pest. RC ***isFoundIn*** indicates in which country is found a pest. RC ***possesses*** indicates which country possesses a plant.

FC ***Plant***	lauraceae	asteraceae	aromatic	comestible	toxic	evergreen	contraceptive	antidysenteric	applicOil	applicEssentialOil	applicExtract
cinnamomumZeylanicum	x		x	x		x			x		
chromolaenaOdorata		x	x		x	x				x	x
aspiliaAfricana		x	x		x	x	x			x	x
ageratumConyzoides		x	x		x	x		x		x	x

Pest	attacksPeanuts	attacksCheese	attacksRice	aspergillus
aspergillusFlavus		x		x
aspergillusParasiticus	x			x
aspergillusOchraceus	x			x
aspergillusTamarii		x	x	x

Country	australAfrica	westernAfrica
Namibia	x	
SouthernAfrica	x	
Nigeria		x
Benin		x

RC ***treats***	aspergillusFlavus	aspergillusParasiticus	aspergillusOchraceus	aspergillusTamarii
cinnamomumZeylanicum	x			x
chromolaenaOdorata			x	
aspiliaAfricana		x		
ageratumConyzoides	x			

RC ***isFoundIn***	Namibia	SouthernAfrica	Nigeria	Benin
aspergillusFlavus	x			
aspergillusParasiticus				x
aspergillusOchraceus			x	
aspergillusTamarii		x		

RC ***possesses***	cinnamomumZeylanicum	chromolaenaOdorata	aspiliaAfricana	ageratumConyzoides
Namibia				
SouthernAfrica				
Nigeria			x	
Benin		x		

Navigating within linked conceptual structures The above example is used to reveal the gradual navigation needs of the RCA outputs[4]. Let us suppose the user wants information about the pests that attack peanuts. Pest Concept $CPe13$ introduces *attackPeanuts* and is therefore the natural **entry point**. Pests of $CPe13$ extent are Aspergillus ochraceus and Aspergillus parasiticus. To know the localization of these pests, the user needs to **follow the relational attribute** $\exists isFoundIn(CC20)$. This attribute leads to Country concept $CC20$ which groups Western Africa countries, i.e. Benin and Nigeria. In $CC20$, the relational attributes containing *possesses* point to plant concepts. Going towards the Plant concept $CPl21$ indicates that $CC20$ countries host Aspilia africana or Chromolaena odorata. $CPl21$ comports the relational attribute $\exists treats(CPe13)$, meaning that its plants treat at least one of the Aspergillus pest species that attack peanuts. It can be interpreted that way: Aspergillus species that attack peanuts are observed in Western Africa countries and are treated by plants that can be found in the countries of this region. It can be interesting here, for the user, to **go back** by choosing Concept $CC20$ to look at relational attributes $\exists possessesCPl6$ or $\exists possessesCPl9$. Choosing $\exists possessesCPl9$ and going to $CPl9$, the user can see that these Western Africa countries possess plants that have characteristics that are close to the plants treating Aspergillus species attacking peanuts. This is the case of Ageratum conizoïdes which is analyzed by **going to sub-concept** $CPl11$. This plant is toxic and has an antidysenteric effect, it may have antibacterial properties. These characteristics may suggest that common chemical compounds could be shared between these 3 plant species. This navigation therefore brings a research hypothesis that domain experts may study through new experimental research: can Ageratum conizoïdes be used to control Aspergillus species (i.e. Aspergillus ochraceus and Aspergillus parasiticus) attacking peanuts?

[4]A navigation for this example using RCAviz is presented at https://rcaviz.lirmm.fr/documentation.html#examples. A video on YouTube is also proposed.

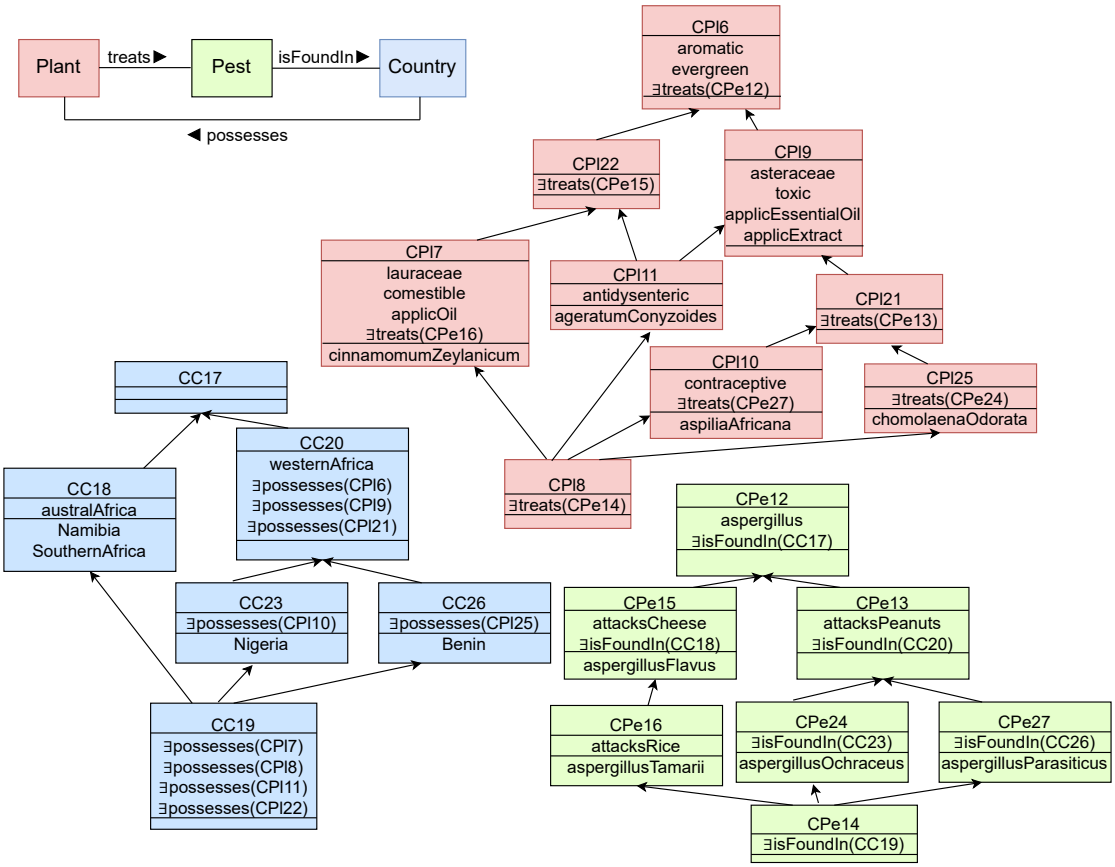

Figure 1: UML model (top left). Classification with RCA of plants (top right), pests (bottom right) and countries (bottom left) of Table 1.

If the answer is true, this then would mean that farmers in Western Africa have an additional local plant that can be used for controlling Aspergillus ochraceus and Aspergillus parasiticus.

This brief navigation example illustrates the main needs of a user: (i) selecting an attribute (or an object) to initiate the navigation, (ii) visualizing the neighborhood of a concept to be able to choose the next step and do not end with an empty solution, (iii) moving from one concept to a sub-concept (or a super-concept) in the same conceptual structure, (iv) jumping to a neighboring conceptual structure using a relational attribute, and (v) backtracking easily.

3. Visualization

In the previous section, we presented the extracted data structure using an RCA classification. As can be seen in Figure 1, this structure can be modeled as a set of Directed Acyclic Graphs (DAG), each DAG corresponding to a context (here, one DAG for plants, one for pests and one for countries). There is an extensive literature on the visualization of DAG (see [11] for an introduction). Most articles present algorithms for positioning vertices and drawing edges to

limit the number of edge crossings. This type of approach is particularly interesting for small datasets, like the one presented in Figure 1. However, when datasets have hundreds of vertices, visualizing the entire structure induces visual cluttering issues. Moreover, in our context, it can be useful to be able to switch from one DAG to another interactively via the relational attributes. This is why we opted for a local visualization system with a step-by-step navigation.

Requirement analysis Generally, visualization design is done by iterating a process consisting of (1) defining needs and suitable data structures, (2) proposing visual encodings and interactive features meeting these needs, (3) providing the results to the users so that they can refine their needs, etc. (see for instance [12, 13]). In our case, we designed RCAviz by organizing regular meetings with the three visualization experts in charge of providing the visual platform, an RCA expert and a dataset expert. Here is the final list of the needs identified during these meetings: **[R1] Selection of a concept.** A step-by-step navigation requires a starting point. Our tool must therefore initially allow the selection of a concept, depending on the objects and attributes of interest for the user. **[R2] Visualization of a concept and its neighbors.** The user must be able to see a concept (with its objects and attributes) as well as its neighbors, i.e. the lower and higher level concepts in the DAG and the neighboring concepts in the other DAGs. **[R3] Navigation step-by-step.** The user must be able to explore neighboring concepts of the displayed concept (whether they belong to the same DAG or to another one). **[R4] History.** The users must be able to keep a history of their navigation, i.e. the list of the previously explored concepts. They must also be able to navigate through this history and to save a state to be able to resume their exploration later.

RCAviz is designed to meet these requirements. The visualization consists of an initial view allowing to select the starting concept, the *Concept Selector*, and two coordinated views: (1) the *Explorer* allows to navigate step-by-step and (2) the *History* allows to visualize the navigation history.

Concept Selector The *Concept Selector* is the first view displayed when one launches RCAviz. It is designed to meet the requirement **[R1]**. It first allows to upload a data file. Once the file is uploaded, the user can choose a context, i.e. one of the DAGs of the dataset. Then appears the list of objects, the list of attributes and the list of concepts, as shown in Figure 2. The user can select one or more objects and one or more attributes. The selection of these elements updates the other lists according to the logical operators used, and in particular updates the list of the corresponding concepts on the right. One can for example display the concepts containing *(object A and object B) and (attribute C or attribute D)*. When the list of objects and attributes is long, a search field allows the user to easily find the elements. The user also can select or deselect all the elements of each list. The list of selected items is displayed when hovering over *Selected*. When an object or an attribute is selected, the corresponding introducer concept number on the right panel appears in bold face. Once the user has found an interesting concept, they can select it from the list on the right and launch the *Explorer* and the *History*. It may seem critical to select a concept according to its index number and the selected associated objects and attributes. Thus, in a future work, we plan to show additional information on the listed concepts, such as their introduced objects and introduced attributes, or their whole extent and

intent.

Figure 2: The *Concept Selector*. The user can select *Objects* and *Attributes* of interest. Then, the list of the available *Concepts* is updated on the right, and the user can select one of them to launch the *Explorer* and the *History*. Concept 20 is in bold face as it introduces the selected attribute *westernAfrica*.

Explorer Figure 3 shows the *History* (top panel) and the *Explorer* (bottom). The *Explorer* consists of three sub-views. The central panel displays the selected concept in the center and its neighborhood above (children in the DAG) and below (parents in the DAG) **[R2]**. Each concept is represented as a colored rectangle composed of stacked boxes. The id is shown at the left corner of the top box. The introduced objects are presented in the box entitled "Objects". Simple (i.e. non relational) introduced attributes appear in another box. The introduced relational attributes, when they exist, appear in the box at the bottom of the concept. When there are several relational attributes with the same relation, they are grouped by relation. In addition, when there are many relational attributes with the same relation, only their number is shown in order to save space, and thus they are not clickable.

For example, the concept 20 in the country DAG in Figure 3 has links to the concepts 6, 9, and 21 in the plant DAG and 13 in the pest DAG. The user can navigate step-by-step by clicking on the concepts **[R3]**. For example, if they click on the concept 23, it is positioned in the middle

of the view and its parents and children are displayed below and above. $\top$, for the top (resp. $\bot$ for the bottom) at the top right of a concept means that it has no parents (resp. no children). To facilitate the navigation, an opposite link, e.g. "isFoundIn-opposite", is provided for each relational attribute. These opposite links do not correspond to a `isFoundIn-opposite` formal context which would be the reverse of `isFoundIn`. In a future work, additional information on each concept (intent, extent, etc.) will be provided to make the navigation easier.

Each DAG is represented by a color. We use a categorical color scale adapted to the representation of nominal variables [14]. In our example, the country DAG is shown in blue, the plant DAG in red, and the pest DAG in green. When the user hovers over a link, the concept in the corresponding DAG is displayed on the right panel. In Figure 3, the mouse is positioned on the red link 21, the corresponding concept is displayed with its neighborhood on the right panel. When the user clicks on the link, the left DAG is positioned in the central panel **[R3]** and that of the central panel is positioned in the right panel. We thus keep track of the previous exploration step **[R4]**.

The concepts that have already been explored appear with lower brightness than the other concepts **[R4]**. In Figure 3, we can see that all the concepts have already been explored except one of the red DAG at the bottom left.

Finally, it is common to have a high number of parents or children. Showing all of them induces visualization and navigation issues. Therefore, if the number exceeds 5, we display a single "super-concept". When the user clicks on it, a pop-up displays a view similar to the *Concept Selector*. The user can then select one or more concepts which, after validation, are displayed in the central panel. In this release, we choose a top to bottom representation of the specialization relation as this is the common representation of partial orders by Hasse diagrams. Alternatively, we may explore a left to right representation, where a concept will be on the left of its sub-concepts, to show more sibling concepts as they will be vertically organized.

History We have seen that a part of the path leading to the concept displayed is available in the left panel of the *Explorer*. The brightness of the concepts also makes it possible to know if a concept has already been explored. However, these functionalities do not fully meet the requirement **[R4]**. We therefore complete them with a view entirely dedicated to **[R4]**: the *History*. As shown in Figure 3, the history consists of two parts: a colored line at the top and a list of concepts below.

The colored line is made up of segments, each segment representing an explored concept. The color represents the DAG of the concept. The length of the line adapts to the width of the visualization so that all the concepts explored are visible. Therefore, the more concepts the user has explored, the smaller the segments representing those concepts are. The user can see if the current concept of the *Explorer* has already been explored with a small triangle placed above each segment representing this concept. For example, in Figure 3, we see two triangles, one on the last segment representing the concept displayed in the *Explorer* and one on a previous segment: this means that the current concept has already been explored once during the navigation.

The list of concepts explored with their id is displayed under the colored line. This list can be long, a horizontal slider is therefore activated when the available space is not enough to display

all of them in the view. The concepts are represented by colored rectangles according to the DAG they belong to. An arrow connects two concepts if the user has moved from one to the other by navigating step-by-step. If the user clicks, in the *History*, on a previously explored concept, it is displayed in the *Explorer* and a vertical bar is used in the *History* to show that the sequence was not produced by a navigation step-by-step. When the user hovers over a concept, a small red cross appears at the corner. It can be used to remove this concept from the *History*. As selecting a concept only according to its id may be tricky, additional information (e.g. intent/extent) will be presented in a future release.

Finally, the user can save the state of their navigation as well as the history by clicking on the save button in the top bar. They can then load the file thus produced to resume their exploration.

Implementation RCAviz is a web platform developed in HTML, CSS and JavaScript. The panels and the visualizations are made using the D3.js[5] [15] library. Saving the history file is made possible with the FileSaver[6] library. The tool is available online[7]. An extensive documentation containing sample datasets is available to allow anyone to use the tool with the samples or with their own datasets.

4. Related Work

Visualizing and exploring in FCA tools There have been a lot of proposals to visualize and explore the conceptual structures. For space reasons, we only mention some of them to highlight the main followed directions. Some proposals visualize the structure itself such as Conexp [16], Latviz [17], Carve (based on a decomposition tree) and DAnCe (with concept enumeration guided by the user) [18], and the approach by expandable and collapsible concept trees [19]. Additional views can be provided, like in RV-xplorer [20], with pie-charts showing local view on a concept, its content and its neighboring, and statistics on the next levels to guide the user. Other approaches do not disclose the DAG structure, such as CREDO [21] or SearchSleuth [22] for web queries which show links and terms that can be clicked to navigate on the underlying lattice, e.g. for specializing or generalizing the query; SORTeD [18] where terms can be added or removed; SPARKLIS [23] for RDF data which uses lists of urls, types, operators; or ConceptCloud [8] which represents a focus concept by a tag cloud showing objects and attributes (with size and colors for highlighting importance and categories) and tag selection to change the focus concept. Some approaches use objects representations other than terms, such as photos [24] or movie posters [25]. As presented in this paper, RCAviz shows the structure, enables selecting a focus concept from initial objects or attributes, enables going step-by-step towards sub-concepts and super-concepts, and backtracks to a previous step.

[5] https://d3js.org/
[6] https://github.com/eligrey/FileSaver.js/
[7] https://rcaviz.lirmm.fr/

Visualizing in RCA tools Only four tools implement RCA: Galicia, eRCA[8] improved in Galatea[9], RCAexplore, and a Cogui[10] plugin. This last plugin was also converted as a Java library named FCA4J[11]. RCAexplore and Cogui have the largest functional features regarding the number of input/output and accepted quantifiers. Regarding the visualization, Galicia provides an advanced lattice representation (tridimensional, graph layout built with a force algorithm) but no specific operators to navigate within a structure or from one structure to another. RCAexplore provides a visualization tool relevant for navigating small-size structures. Compared to Galicia, eRCA/Galatea, and Cogui, RCAexplore provides a browser enabling the navigation between concepts textually described (no visualization), shows the whole structure but does not enable zooming. Concepts and relational attributes are clickable to go from one structure to a neighboring structure, but selecting a focus concept and going step-by-step using local views is not proposed. Finaly eRCA/Galatea, RCAexplore, and Cogui produce outputs in the dot file format, which can be visualized or transformed using Graphviz[12]. In RCAviz, several visualization panels show local views, and allow the user to know which neighboring structure has been previously accessed or can be accessed through relational attributes. The step-by-step navigation is done by clicking on the relational attributes. The colors help to distinguish the different sets of objects. Table 2 synthesizes the main features of RCA tools regarding our requirements.

Table 2
Comparison with other RCA tools on the visualization requirements.

	R1 Initial concept selection by O/A	R2 Local visu with only concept neighbors	R3 Step-by-step nav. among concepts with clicking	R4 History
Galicia				
eRCA/Galatea				
RCAexplore		Textual descr. only (concept browser)	In concept browser and Along relational attr. in global visu.	
RCAviz	x	x	x	x

5. Conclusion

This paper presents RCAviz, a web platform devoted to visualize and navigate within relational conceptual structures computed using RCA. First uses showed the relevance of the features adopted to navigate within and between conceptual structures. These uses also enabled to identify a need for additional features to be integrated, e.g. specifying the path length to be shown, presenting the complete conceptual structures when they are small, local computation as in [26], visualizing the ER model, showing a larger concept neighborhood, or expressing

[8]https://code.google.com/archive/p/erca/
[9]https://github.com/jrfaller/galatea
[10]http://www.lirmm.fr/cogui/
[11]https://www.lirmm.fr/fca4j/
[12]https://graphviz.org/

complex queries to initiate the exploration. Additional testings will be conducted to consolidate the reliability of the software.

This release is a first step in the building of a general platform to support exploratory search in relational datasets. To this end, future development will also focus on considering implications visualization. Large user experiments will be carried out in the context of the Knomana project.

Acknowledgments

The authors wish to thank the anonymous reviewers for their valuable comments that helped improving the paper. They also warmly thank Alain Gutierrez for his strong support during this work and for enabling FCA4J to generate output files with the JSON data format as required by RCAviz. This work was supported by the French National Research Agency under the Investments for the Future Program, referred as ANR-16-CONV-0004.

References

[1] B. Ganter, R. Wille, Formal Concept Analysis: Mathematical Foundations, Springer, 1999.

[2] J. Poelmans, D. I. Ignatov, S. O. Kuznetsov, G. Dedene, Formal concept analysis in knowledge processing: A survey on applications, Expert Syst. Appl. 40 (2013) 6538–6560.

[3] J. Poelmans, S. O. Kuznetsov, D. I. Ignatov, G. Dedene, Formal concept analysis in knowledge processing: A survey on models and techniques, Expert Syst. Appl. 40 (2013) 6601–6623.

[4] N. Messai, M. Devignes, A. Napoli, M. Smaïl-Tabbone, Querying a bioinformatic data sources registry with concept lattices, in: ICCS, volume 3596 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 323–336.

[5] Z. Azmeh, M. Huchard, A. Napoli, M. Rouane-Hacène, P. Valtchev, Querying Relational Concept Lattices, in: CLA'11, 2011, pp. 377–392.

[6] S. Ferré, Reconciling Expressivity and Usability in Information Access, Habilitation à diriger des recherches, Université de Rennes 1, 2014. URL: https://hal.inria.fr/tel-01100292.

[7] C. Carpineto, G. Romano, Concept data analysis - theory and appl., Wiley, 2005.

[8] G. J. Greene, M. Esterhuizen, B. Fischer, Visualizing and exploring software version control repositories using interactive tag clouds over formal concept lattices, Information and Software Technology 87 (????).

[9] P. Silvie, P. Martin, P. Marnotte, M. Huchard, Projet Knomana. KNOwledge MANAgement on pesticidal plants in Africa for a safer food and a better environmental health. Rapport d'extractions, Technical Report, Métaprog. Glofoods INRA-CIRAD, 2019.

[10] M. R. Hacene, M. Huchard, A. Napoli, P. Valtchev, Relational concept analysis: mining concept lattices from multi-relational data, Ann. Math. Artif. Intell. 67 (2013) 81–108.

[11] P. Healy, N. S. Nikolov, Hierarchical drawing algorithms, in: R. Tamassia (Ed.), Handbook on Graph Drawing and Visualization, Chapman and Hall/CRC, 2013, pp. 409–453.

[12] T. Munzner, A nested process model for visualization design and validation, IEEE Transactions on Visualization and Computer Graphics 15 (2009) 921–928.

[13] M. Sedlmair, M. D. Meyer, T. Munzner, Design study methodology: Reflections from the

trenches and the stacks, IEEE Transactions on Visualization and Computer Graphics 18 (2012) 2431–2440.
[14] T. Munzner, Visualization Analysis and Design, A.K. Peters visualization series, A K Peters, 2014.
[15] M. Bostock, V. Ogievetsky, J. Heer, D^3 data-driven documents, IEEE Transactions on Visualization and Computer Graphics 17 (2011) 2301–2309.
[16] S. A. Yevtushenko, Conexp, 2018. URL: http://conexp.sourceforge.net/.
[17] M. Alam, T. N. N. Le, A. Napoli, Latviz: A new practical tool for performing interactive exploration over concept lattices, in: CLA'16, 2016, pp. 9–20.
[18] T. Pattison, Interactive visualization of formal concept lattices, in: ED/GViP@Diagrams, volume 1244 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2014, pp. 78–89.
[19] S. Andrews, L. Hirsch, A tool for creating and visualising formal concept trees, in: CSTIW@ICCS, volume 1637 of *CEUR Workshop Proceedings*, 2016, pp. 1–9.
[20] M. Alam, M. Osmuk, A. Napoli, RV-Xplorer: A Way to Navigate Lattice-Based Views over RDF Graphs, in: CLA'15, 2015, pp. 23–34.
[21] C. Carpineto, G. Romano, Exploiting the Potential of Concept Lattices for Information Retrieval with CREDO, J. UCS 10 (2004) 985–1013.
[22] J. Ducrou, P. Eklund, SearchSleuth: The conceptual neighbourhood of an Web query, in: CLA'07, ????, p. 14.
[23] S. Ferré, Sparklis: An expressive query builder for SPARQL endpoints with guidance in natural language, Semantic Web 8 (2017) 405–418.
[24] S. Ferré, CAMELIS: Organizing and Browsing a Personal Photo Collection with a Logical Information System, in: CLA'07, Montpellier, France, 2007, pp. 112–123.
[25] M. Crampes, M. Plantié, Visualizing and Interacting with Concept Hierarchies, in: Proc. of WIMS14, ACM, 2014.
[26] A. Bazin, J. Carbonnel, M. Huchard, G. Kahn, P. Keip, A. Ouzerdine, On-demand relational concept analysis, in: ICFCA 2019, volume 11511 of *LNCS*, Springer, 2019, pp. 155–172.

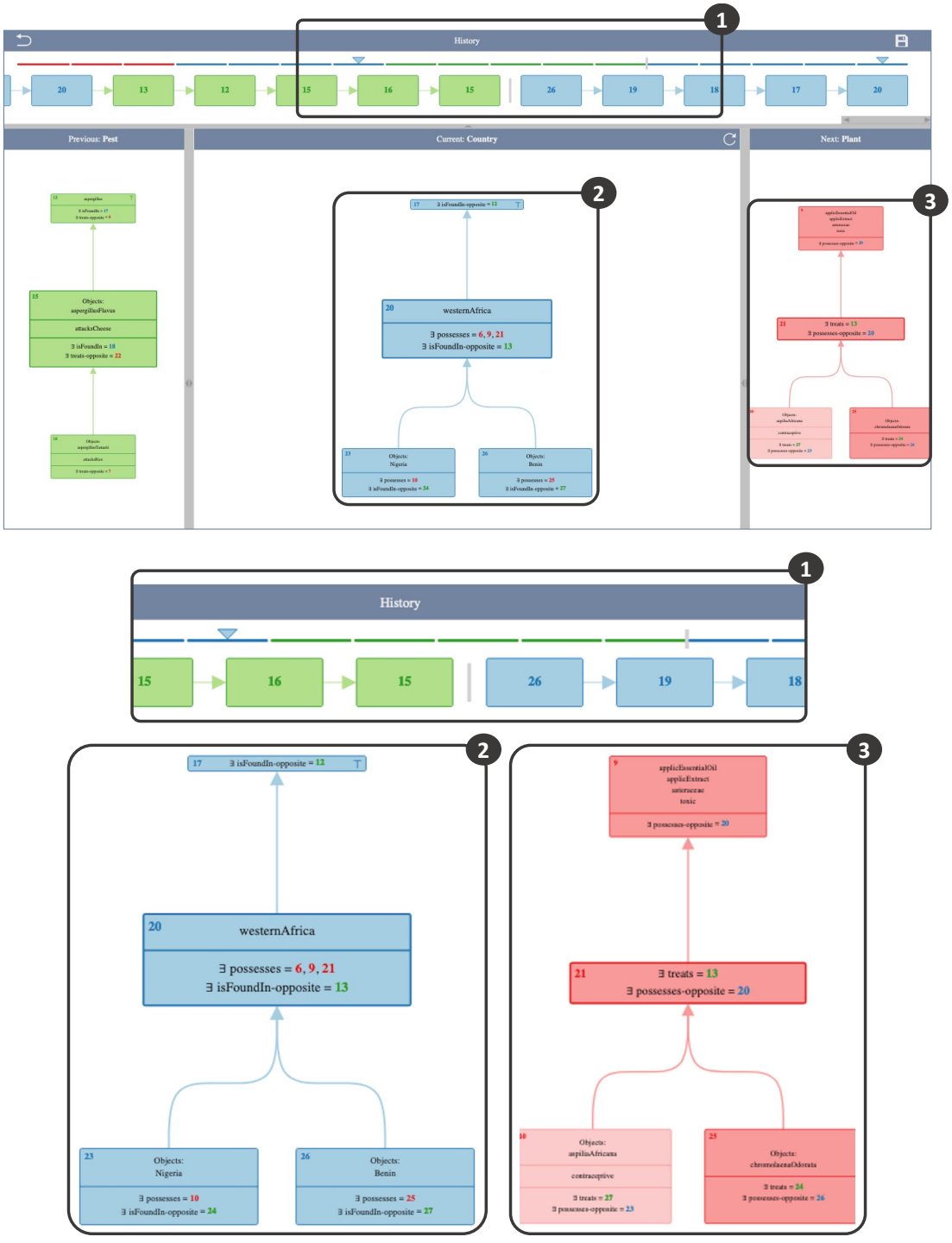

Figure 3: The *History*. The top panel shows the *History* containing the list of the previous explored concepts. The *Explorer*. On the center panel, the view shows a concept and its neighborhood. The user can navigate through the concepts by clicking on them. The left panel shows the previous DAG displayed. When the user hovers over a link of another DAG, the last is displayed on the right panel.