Galicia: an open platform for lattices

Petko Valtchev[1], David Grosser[1], Cyril Roume[2], Mohamed Rouane Hacene[1]

[1] DIRO, Université de Montréal, C.P. 6128, Succ. "Centre-Ville",
Montréal, Québec, Canada, H3C 3J7
[2] LIRMM, CNRS et Université Montpellier 2, 161 rue Ada
34392 Montpellier, Cedex 5, France

Abstract. Formal concept analysis (FCA) has proved helpful in the resolution of practical problems from fields such software engineering, knowledge engineering and data mining. Recently, a substantial push has been done toward the design of efficient procedures for lattice construction, with a variety of novel algorithms proposed in the literature. However, the FCA community has created only few effective tools for manipulating lattices so far and what is still missing is an integrated environment for constructing, visualizing, exploring and maintaining lattices. We present the Galicia project aimed at the construction of an open platform for lattice manipulation which follows the complete life-cycle of a lattice. More than just a lattice tool, the platform provides the necessary services for quick development and test of new lattice algorithms.

1 Introduction

Although Formal Concept Analysis (FCA) was initially intended as an alternative to the classical lattice theory [5], the discipline has largely outgrown this narrow frame. In particular, practitioners from various fields, e.g., social sciences, data analysis, software engineering, etc., have adopted the theoretical framework of FCA and made of the corresponding algorithmic methods an effective tool for the extraction of conceptual structure from raw data.

Nowadays, the FCA community has practical applications in, among others, data mining, re-engineering of software and knowledge acquisition. The number of practitioners that adhere to the concept lattice paradigm is steadily growing. Meanwhile, following the recent explosion in the volume of information that needs to be analyzed, the attention is increasingly drawn to the algorithmic aspects of the discipline and to their software realization, with requirements for scalability and flexibility. However, to our best knowledge, there is no software tool that supports the (formal) analysis process along its entire span. Thus, despite the large amount of source code produced and filed by the members of the FCA community, reliable and free software is probably yet to come[3], while available commercial tools cover only separate tasks, e.g., the visualization in Toscana [12].

[3] See the Tockit project at `http://www.tockit.org`.

For the above reasons, we claim that what is needed is an integrated environment for the manipulation of lattices. In addition, the environment should remain open source so that possible extensions could be easily designed. Yet another problem is the lack of software support for the development of new lattice-based tools. Our own experience has taught us that a system providing basic services such as lattice visualization and navigation, could greatly speed-up the process.

Motivated by the above observations, we initiated the GALICIA project whose main goal is to bridge the gap between theoretical advances in FCA and the level of tool availability. Our approach took the shape of an open platform integrating a collection of tools to support the entire life-cycle of a concept lattice (data preprocessing, construction, visualization, navigation, maintenance, etc.). The platform is intended for practitioners as it provides basic services necessary for practical applications of FCA. In addition, GALICIA offers to FCA researchers advanced tools for performance studies, as well as an open environment for the quick design of new lattice-related techniques.

The paper starts with a recall of classical FCA notions and of some newer results relevant to the project (Section 2) together with a critical view on the current availability of software tools in the domain. Next, the goals of the project are presented, followed by a description of the data processing and analysis services provided by the platform (Section 3). Particular emphasis is put on two facets of the platform which are the most thoroughly realized up till now, its visualization component (Section 4) and its open architecture (Section 5).

2 Background

In the following paragraphs, we recall key notions from FCA and summarize algorithmic developments up to date. Also, our previous work is put in this context.

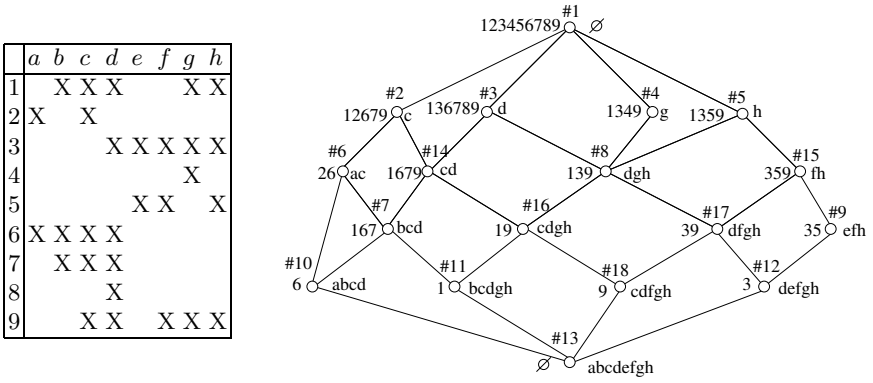
2.1 Formal concept analysis basics

Formal concept analysis [5] (FCA) studies the partially ordered structure, known under the names of *Galois lattice* [1] or *concept lattice*, which is induced by a binary relation over a pair of sets O (*objects*) and A (*attributes*).

Definition 1 *A formal context is a triple $\mathcal{K} = (O, A, I)$ where O and A are sets and I is a binary (incidence) relation, i.e., $I \subseteq O \times A$.*

Within a context (see Figure 1 on the left), objects are denoted by numbers and attributes by small letters. The way a lattice arises on top of a context may be summarized as follows (see [5] for details). Two closure operators are associated to a context (further denoted by $''$ as in X''), one for each dimension of the table, leading to two families of closed sets. Moreover, each family of closed sets is organized into a complete lattice by the set-theoretic inclusion,

whereas both lattices are anti-isomorphic, i.e., the underlying isomorphism is an antitonic mapping. The pairs of closed sets in the isomorphism are called *(formal) concepts* (where the object set is referred to as the *extent* and the attribute set as the *intent*). The concepts correspond to the maximal rectangles of the table which are made up of strictly positive entries (e.g., $(167, bcd)$ is a concept[4], while $(13, dh)$ is not). Finally, the concept or Galois lattice is the overlapping structure made up of both lattices in which the inclusion on extents prevails, i.e., a concept is greater than (is a super-concept of) another one iff its extent is larger.



	a	b	c	d	e	f	g	h
1		X	X	X			X	X
2	X		X					
3				X	X	X	X	X
4							X	
5					X	X		X
6	X	X	X	X				
7		X	X	X				
8				X				
9			X	X		X	X	X

Fig. 1. Left: Binary table $\mathcal{K} = (O = \{1, 2, ..., 9\}, A = \{a, b, ..., h\}, R)$. **Right:** The Hasse diagram of the lattice of $\mathcal{K}$.

The Hasse diagram of the lattice $\mathcal{L}$ drawn from $K = (\{1, 2, ..., 8\}, A, R)$ is shown on the right-hand side of Figure 1 where intents and extents are drawn on both sides of a node representing a concept. It is noteworthy that contexts and their lattices constitute alternative representations of the same reality [5].

Relevant FCA constructs include composition/decomposition operations for contexts and the corresponding operations on lattices. Among those, the splits of a context on its object/attribute set and the reverse "join" operations called *subposition/apposition*, respectively, are worth mentioning. The equivalent lattice operations, the semi-products, underly an important visualization technique for lattices (see below) and may even have significant computational benefits.

Besides the complete lattice, two other structures, derived from the lattice, have acquired a large popularity among FCA practitioners: the iceberg lattices and the Galois sub-hierarchy (GSH). Iceberg lattices are maximal upper sets (or order filters) of a lattice, which means they are generated by maximal anti-chains of the lattice. Intuitively, an iceberg arises through a complete horizontal

[4] A separator-free form for sets is used, e.g., 127 stands for $\{1, 2, 7\}$

[5] The third representation mode, the one made up of attribute implications [6] will not be discussed here.

cut of the lattice into two parts, lower one and upper one. The upper one, called the *iceberg* lattice, includes general concepts while excluding the overtly specific ones. Extent cardinality restrictions account for the most popular cut criteria for icebergs, although other factors could also match. For example, the iceberg of the lattice in Figure 1, obtained by a cut factor $|Extent(c)| \geq 3$ is shown in Figure 2, on the left.

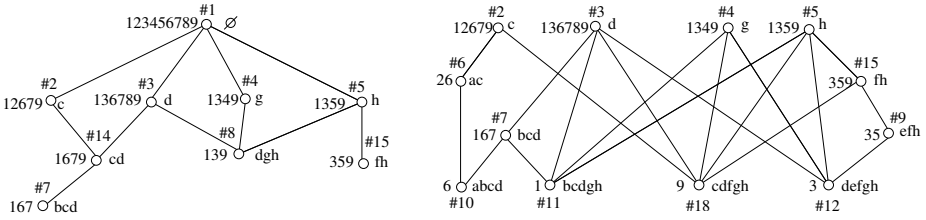


Fig. 2. Left: The iceberg of the lattice in Figure 1, obtained by a cut factor $|Extent(c)| \geq 3$. **Right:** The GSH of the context from Figure 1.

Works on FCA applications to software engineering (SE) problems such as hierarchy optimization in object-oriented systems [6, 4] have underlined the importance of another sub-structure of the complete lattice, GSH. The latter represents the restriction of the lattice order to the set of all object concepts and attribute concepts, i.e., concepts that are minimal (maximal) among those whose extent (intent) includes a given object (attribute). As an illustration, Figure 2, on the right, shows the GSH of the context from Figure 1.

Recent work within the SE domain, namely on the re-engineering of analysis-level models of software described as UML class diagrams, has brought to the forth the need for a richer description formalism to accommodate the links that may exist among objects in a context (e.g., the associations among classes in UML). As a straightforward approach to the problem, the notion of *Relational context family* (RCF) has been proposed (see [7] for formal definitions and first results). A RCF is a family of contexts to which a set of higher-order binary relations has been added. The binary relations describe links between objects from two (possibly identical) contexts of the family. For example, a relation r binding the above context K to itself could be $r = \{(1,6),(2,3),(2,8),(4,7),(5,6)\}$. The links behind the relations are invariably present in a realistic domain model, e.g., kinships, spatial relations, "part-of" relations, etc. Often, they convey valuable information that may determine the way objects are grouped into (domain) concepts.

2.2 Construction of lattices and substructures

Lattice construction from contexts has been a challenge since the very early days of FCA. The problem is a hard one since in the worst-case, there can be

exponentially many concepts. However, in practical cases, only a small number of concepts do occur, so it makes sense to look for methods that discover and, if necessary, hierarchically organize them, in an efficient manner. There is nowadays a large variety of algorithms dedicated to the computation of either the set of all concepts or the entire lattice, i.e., concepts plus order[6].

A major distinction among these algorithms lays in the way they acquire input data. According to a classical dichotomy, batch algorithms consider all the data to be completely known beforehand. In contrast, on-line algorithms allow small changes in the data to be propagated to the final result, i.e., the concept lattice, without starting from scratch. Thus, they can be used to simulate batch lattice construction by a sequence of object/attribute additions to an initially void context (also called incremental construction). Recently, we have suggested a novel paradigm for lattice construction that generalizes the increments to sets of attributes/objects [11]. Based on apposition/subposition and semi-products, we defined a binary operation on complete lattices, called ASSEMBLY and further extended it to a first-class lattice construction procedure that implements a "divide and conquer" strategy.

Besides complete lattices, the lattice substructures mentioned in the previous paragraph have also been the target of algorithmic work. Thus, specific techniques have been designed to cope with these structures directly and therefore at lower cost. Concerning GSH, a set of efficient algorithm has been proposed in the literature, both batch and incremental, starting with the work of Godin [6]. Icebergs remain closer to lattices and therefore few algorithms have been explicitly designed to construct them. Dedicated methods for icebergs include the recent work on the TITANIC [9] method.

2.3 Lattice visualization

An assistance to the examination of complex lattices, that may even arise from small-size context, has been traditionally provided through visualization. As lattices are usually represented through their Hasse diagrams which are basically directed acyclic graphs, their drawing meets the same difficulties and suffers on the same limitations as general graph drawing. Thus, it is quite a challenge to represent a lattice of more than, say, 50 concepts in a readable way at a normally-sized computer screen.

To deal with the "curse" of the lattice size, the research in FCA has yielded an original method for visualizing complex concept lattices (see [5]). Thus, the lattice corresponding to a context is drawn as embedded in a nested structure made up of two or more smaller lattices. The method relies on two dual operators for assembling contexts called *subposition* and *apposition* respectively. Both operations may be thought as the reverse of a horizontal, respectively vertical, split of a context into two fragment contexts. Just as the initial context can be retrieved by joining both fragments, the lattice of the global context can be obtained by assembling the lattices of the fragments. Prior to our work on lattice

[6] An algorithmic is beyond the scope here, hence the reader is referred to [8].

assembly [11], context fragmentation has only supported lattice visualization: the direct product of the fragment lattices is used as a multi-level framework within which the nodes belonging to the global lattice are highlighted. The resulting method, called nested line diagrams (NLD), as each level may be drawn as embedded in the upper levels, not only enables the visualization and the interactive exploration of complex lattices, but also provides for a multi-viewpoint presentation of the analysis results.

The TOSCANA [12] system has been designed to support exploration of complex lattice through navigation of NLD. The system offers valuable services such as single and multi-scale diagram drawing, navigation, etc.

2.4 Actual needs in the domain

The last decade has seen a significant increase in the number of the FCA community members (about 1 200 hits for the query "FCA + lattice" via Google). On the one hand, more practitioners get interested in applying the FCA theoretical achievements, which supposes the appropriate software tools are available. On the other hand, more researchers get involved in studies of the theoretical and/or algorithmic aspects of FCA, in which tool development is often required. As a result, a large quantity of source code dedicated to a particular FCA-related task has been or is being currently developed or maintained. Due to the specific features of the discipline, mainly because of its young age, most of the time the produced code is neither reused nor reusable. This leads to a situation in which newcomers in the community desperately look for some available source code to help them build their own specific tools.

Under these circumstances, we see an urgent need for a generic software solution for the following frequently reoccurring problems: lattice construction, visualization, maintenance and navigation. Such a solution could save effort for a large part of the community, on the one hand, and enforce a certain degree of cohesion among the researchers by facilitating data and results exchange, on the other hand. In addition, the development of new applications could largely benefit from the availability of adaptable code, which in turn will increase the external visibility of the entire FCA paradigm.

To fulfill the above general requirements, a system should exhibit specific software qualities, in particular high *adaptability* and *extensibility*. Adaptability is necessary in order for the system to fit various situations and problem settings. Extensibility follows adaptability as it takes further effort on the appropriate design and implementation of the system, so that it could easily integrate new features. Extensibility is highly desirable for a tool which is intended to support research on a young and dynamically developing discipline such as FCA.

It is noteworthy that the first answers to the above questions are getting on track. On the one hand, the Tockit project concentrates the work on a large-scale software framework for FCA-related tasks. On the other hand, first multi-functional tools for basic FCA are getting available[7].

[7] See for example the CONEXP tool at `http://www.sf.net/projects/conexp/`.

As neither of the above approaches fits our own goals and schedule, we have launched our own software project, called Galicia. The project is aimed at the development of a fully integrated platform, i.e., a complete set of tools for manipulating contexts and lattices, built on top of a common kernel providing basic services (I/O, low-level data processing, etc.), and sharing a user-friendly graphic interface. In addition, a key feature of the target platform is the open architecture that supports adaptive maintenance and future extensions.

3 Galicia

Galicia is intended as an integrated software platform including components for the key operations on lattices that might be required in practical applications or in more theoretically-oriented studies. Thus, the basic configuration of the platform performs major functions such as context input, lattice construction and visualization.

3.1 Goals and scope of Galicia project

The platform started as an ordinary lattice tool to support our research on applications of FCA to data mining and software engineering. Initially, the volume of the data and its complex structure were the major stakes, which put the emphasis on expressive power and efficiency of the tool whose configuration was limited to lattice computation and exploration. In addition, to support the exploratory analysis mode, the specifications were extended to cover major aspects of lattice maintenance upon data evolution (e.g., add/remove of elements of the context). Later on, as computational efficiency and intelligibility brought some coarse-grain lattice operations to the forth, algorithms to carry out those operations have been added as well.

The quick evolution of the tool requirements persuaded us to move to a more open architecture, with an emphasis on extensibility both on data representation and on lattice manipulation levels. Moreover, the absence of equivalent tools in the FCA community became a major motivation for us to make our platform available for the entire community, a fact that further emphasized the need for generic implementations that adapt easily to various circumstances and analysis goals. To sum up, Galicia is designed to cover the whole range of basic tasks that make up the complete life-cycle of a lattice as we understand it (see Figure 3).

The intended impact of the platform is two-fold since it should support both applications of FCA and development of new lattice-based techniques. As a FCA-tool, Galicia offers an open architecture and generic implementations which ease its adaptation to a particular application domain and problem settings. For example, a wide range of data formats (context types) are allowed in the platform. Beside classical binary contexts, multi-valued contexts are admitted in the design as well as more complex data descriptions (e.g., the previously mentioned relational context families). In addition, a rich set of algorithmic

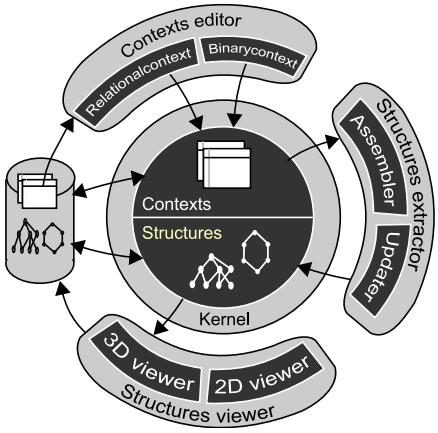


Fig. 3. Life-cycle of a lattice: contexts are either loaded or created by means of context editor; lattices are constructed and visualized; rearrangements of the context are performed to clarify the lattice structure; the resulting lattice is reduced to a suborder or decomposed into smaller lattices.

methods for lattice construction and maintenance is included in the system's architecture.

Finally, the platform offers several visualization mechanisms including 2D and 3D graph drawing modes and should soon provide a nested-line-diagram mode.

3.2 Context manipulation

Galicia admits both classical, i.e., binary, and multi-valued contexts to be processed. In addition, relational context families (RCF) can be manipulated within the platform. Basic context manipulations include input/output to a disk file, interactive editing, split into fragments (subcontexts), etc. A context can be modified by changing the composition of its object/attribute sets or the set of pairs in the incidence relation ('X' in the binary table). The edition of a RCF admits modifications in the relation r_i as well (see Figure 4). Several data formats are available in Galicia such as IBM formats for transaction databases, proprietary human-readable formats for contexts and lattices, etc. Other formats are currently examined in order to insure the inter-operability with existing tools (e.g., in application domains).

3.3 Lattice construction and maintenance

The lattice manipulation functions of Galicia cover many of the classical algorithms for lattice construction as well as a large set of recent lattice-related techniques which have been designed by members of our team.

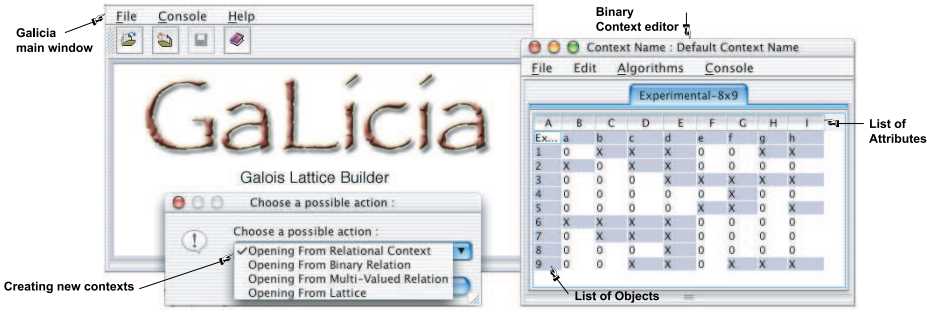


Fig. 4. Loading data dialog box with context family editor.

The platform features a set of lattice construction algorithms. In fact, as different algorithm performances may vary depending on type of the context, we consider that it is crucial for applications where efficiency is critical, to offer the tool user a choice among the most powerful techniques. In addition, the availability of several competing algorithms eases the development and test of new ones, in particular, the comparison of practical performances. A special emphasis has been put on incremental construction with GALICIA featuring both the initial algorithm of Godin *et al.* and our recent generalization [10]. Moreover, a lattice construction following a divide-and-conquer approach (based on lattice assembly) is included in the platform design.

The platform offers a set of iceberg construction techniques as well, including an implementation of TITANIC. Incremental construction is performed by our MAGALICE algorithm. Similarly, several modes and techniques for GSH construction are included in the platform design. In the batch mode, the available construction services rely on an implementation of the algorithm CERES, whereas the incremental mode features ARES and the method of Godin and Mili [6].

Being able to provide efficient support for data evolutions is of high importance for FCA tools intended for exploratory tasks. As an approach to the problem, we specified a complete set of context modifications (add and remove of lines/columns/X's in the table) together with the corresponding operations that propagate efficiently these modifications to the lattice. Implementations of these operations support the interactive lattice construction mode of GALICIA in which the user gradually refines his/her view on which parts of the data are worth analyzing. Moreover, the same operations applies to iceberg lattices and GSH.

The platform design provides support for higher-level operations on lattices such as *assembly* and *split* of lattices. These can be seen as the add/remove of several lines or columns at a time and therefore generalize the classical on-line algorithms. Instead of a one-element-at-a-time strategy, assembly and split consider the initial and the modified contexts, as well as the context made up of the elements in the difference, together with their corresponding lattices. In the

case of assembly, the final lattice is directly obtained as the semi-product of the initial lattice and the lattice drawn from the difference [11].

One of the most original features of GALICIA is the support for multiple lattice construction on top of a RCF. To that end, we have designed the MULTI-FCA procedure [7]. Two variants of MULTI-FCA have been devised: one for complete lattices and another one for GSH. The former is intended as a tool for our theoretical investigations whereas the latter supports a concrete application in the software re-engineering area.

4 Visualization

Drawings of concept lattices provide the most common mechanism for the communication of FCA results [3]. Layered diagrams constitute a common graph drawing approach to the layout of a partially ordered sets. Within such a diagram, each vertex is assigned to a horizontal layer while the vertices of a layer are ordered to reduce edge crossings. Ordering relies on various heuristics since the general crossing minimization problem is known to be NP-complete [2].

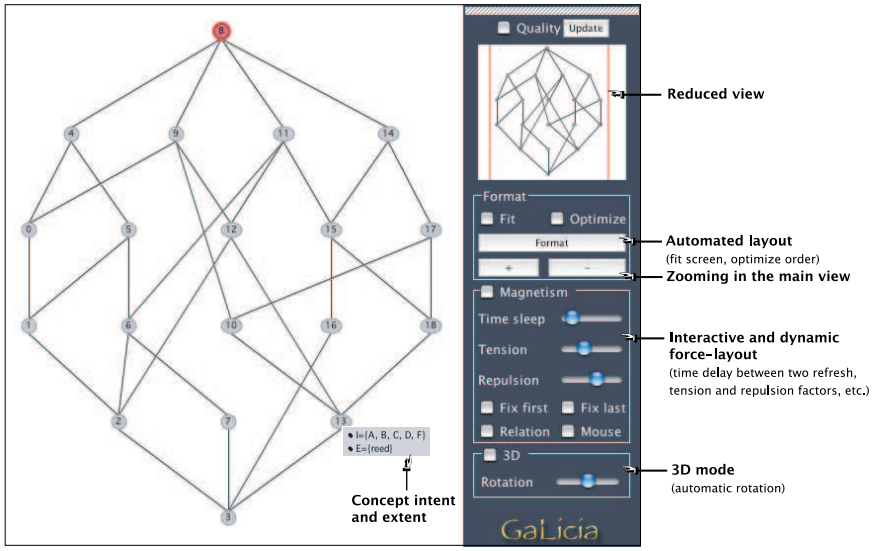


Fig. 5. Layered diagram of a concept lattice and layout parameters.

GALICIA provides two layout methods, an automated one and an interactive one. The former method relies on local optimization heuristics: the edge crossings are minimized layer-wise, with a top-down traversal of the layer set. The second one follows a strong analogy with the magnetism phenomena since vertices and

even edges are assigned attraction and repulsion forces[8]. Thus, layout elements sustain each other's impact which tends to push each of them to a specific part of the drawing. The key idea is to reach a global layout as a convergence point of minimal energy for the entire set of elements.

The platform provides a user-friendly interface for concept lattice layout. The layout console shown in Figure 5 contains the main graph drawing tools. Thus, zooming in/out and reduced view of the entire diagram enable the navigation through complex lattices. Moreover, the parameters that tune the layout methods are included: switches for second-order heuristics, various force degrees, rotation speed and direction for the 3D mode, etc.

Diagrams can be displayed in 2D or 3D modes (see Figure 6) regardless of the layout method. In the 3D mode, the lattice model is a mapping of each vertex on a 3D sphere.

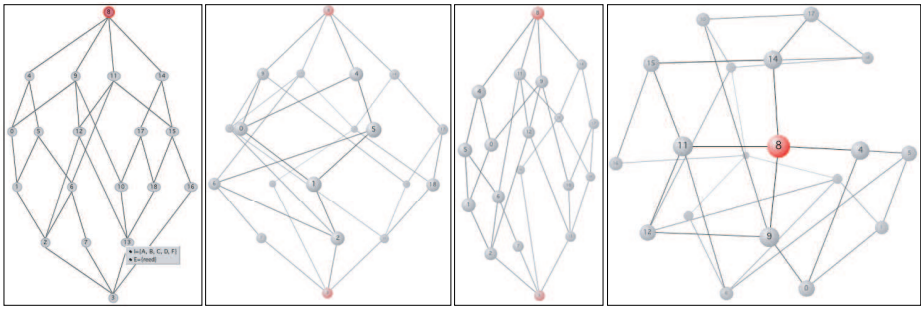


Fig. 6. a) 2D-force-layered. b) 3D-spherical layout. c) 3D-force-layout. d) 3D-spherical layout, top view.

5 Design and implementation of the platform

Galicia has been designed as an open platform of independent components.

5.1 Architecture

System architecture reflects the push towards adaptability, extensibility and reusability. In fact, a traditional layered architecture has been used, with the lower most and upper most level composed by the Java environment and user interaction services, respectively (see Figure 7). The remaining layers represent the platform kernel and the tool set. The kernel is in charge of all low-level services related to the representation and manipulation of contexts and ordered structures. These services are exported toward the tool layer where the implementations of high-level FCA methods lay.

[8] See also Freeze's LatDraw (`www.math.hawaii.edu/~ralph/LatDraw`).

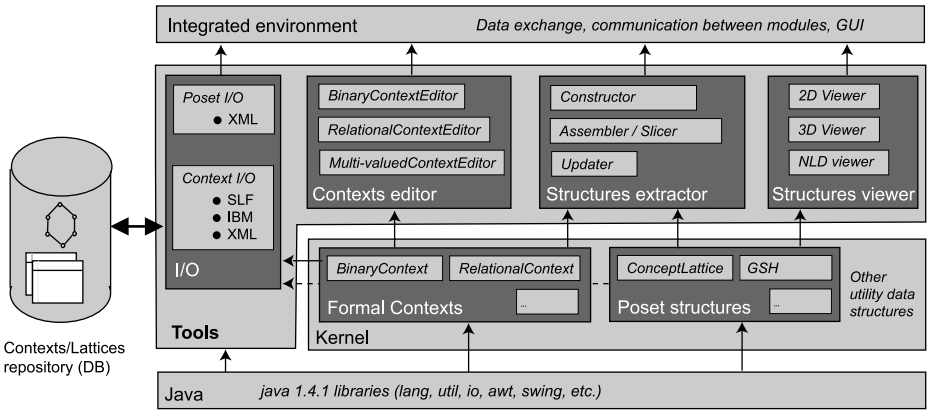


Fig. 7. Architectural layers of the GALICIA platform and dependencies between components.

The architecture of the second highest layer of GALICIA (see Figure 7) follows the partition of the platform functions into life-cycle phases. Thus, the *Import/Export engine (I/O)* insures the input/output flow of data and results whereas the *Context editor* offers the necessary environment for interactive creation and/or modification of contexts and RCF. The *Structure extractor* component is responsible for the construction and/or maintenance of lattices and posets. It contains a set of concrete methods which are further organized into subcomponents (see Figure 8) with respect to the kind of structure manipulation they perform. For example, the *Constructor* subcomponent provides services of pure construction from data, either batch or incremental. Methods for assembly

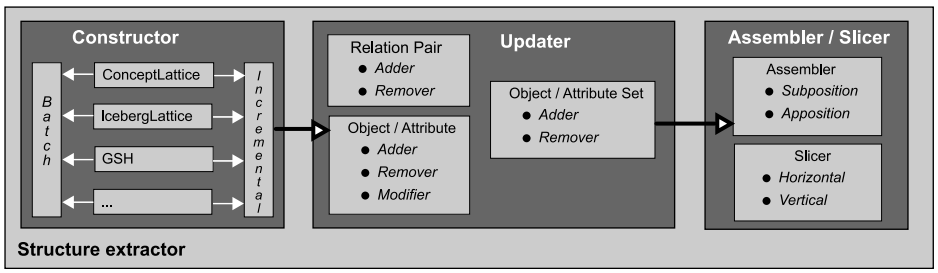


Fig. 8. The architecture of the structure extracting component of GALICIA.

and decomposition of lattices are located in *Assembler/Slicer* while the `Updater` is responsible for maintaining incrementally posets when attributes, objects or (object, attribute) pairs are added/removed. Finally, the *Lattice viewer* features

poset visualization and browsing. Two modes for lattice representation are available, 2D and 3D, while for complex lattices, a nested line diagram engine (still in progress) enables gradual exploration through navigation.

To sum up, the architecture of Galicia favors the integration among tools since these tools use the same kernel services while interacting with the user via a uniform interface.

5.2 Detailed design and implementation

The modular architecture of Galicia facilitates modifications, both in the set of available algorithmic methods, i.e., in the tool layer, and in the specific data structures used in the kernel. First, extensive use of abstract data types (for posets, contexts, context elements, etc.) results in generic services whose implementation can be easily modified. Consequently, dependency of client tools on implementation decisions is reduced while easing the integration of alternative tools for the same task (e.g., favoring different aspects such as efficiency versus low memory consumption).

In its current version[9], the platform provides full support for the minimal set of FCA tasks, i.e., input data processing (import/export and editing), lattice construction (two incremental algorithms) and visualization (2D and 3D modes). Moreover, the Multi-FCA method is already available in its GSH-based variant. Implementation of further construction algorithms (both batch and incremental) as well as more complex manipulations such as NLD visualization and assembly/split of lattices, are still under design. More research-oriented features, e.g., a test generator for comparative performance studies, are also being examined.

6 Discussion

Galicia is an open platform that provides support for the entire life-cycle of a lattice. It offers basic services (I/O for various data types, representation and manipulation of contexts and structures, etc.), a collection of various algorithms for the construction of lattices and related order structures, and a powerful diagram visualization tool that works both in 2D and in 3D modes.

Besides being a tool for FCA practitioners, Galicia fits perfectly to scientific studies in which both theoretical and algorithmic aspects are investigated. First, a minimal framework which is necessary for routine performance studies on competing techniques is already available within the platform. The framework can be easily extended to suit more sophisticated problem settings. Moreover, Galicia's modular architecture and generic code allows new and complex data types to be integrated to the platform with a reasonable effort while reusing most of the low-level services. As an illustration of this feature, we integrated into Galicia the processing of *relational context families* which represent relations in the data in a way inspired by UML.

[9] Latest public releases of Galicia available at `www.iro.umontreal.ca/~valtchev/galicia/`.

Acknowledgements

The authors want to thank to anonymous referees who helped improve the paper, as well as to the senior members of the GALICIA team, especially to Robert Godin and Rokia Missaoui.

References

[1] M. Barbut and B. Monjardet. *Ordre et Classification: Algèbre et combinatoire*. Hachette, 1970.

[2] G. Battista, P. Eades, R. Tamassia, and I. Tollis. *Graph dawing: Algorithms for the visualisation of graphs*. Prentice Hall, 1999.

[3] R. Cole. Automated layout of concept lattices using layered diagrams and additive diagrams. In *Australasian Computer Science Conference (ASC'01)*, pages 47–60, Queensland, Australia, 2001.

[4] H. Dicky, C. Dony, M. Huchard, and T. Libourel. On automatic class insertion with overloading. In *Proceedings of OOPSLA'96, San Jose (CA), USA*, special issue of ACM SIGPLAN Notices, 31(10), pages 251–267, 1996.

[5] B. Ganter and R. Wille. *Formal Concept Analysis, Mathematical Foundations*. Springer-Verlag, 1999.

[6] R. Godin and H. Mili. Building and maintaining analysis-level class hierarchies using Galois lattices. In *Proceedings of OOPSLA'93, Washington (DC), USA*, special issue of ACM SIGPLAN Notices, 28(10), pages 394–410, 1993.

[7] M. Huchard, C. Roume, and P. Valtchev. When concepts point at other concepts: the case of uml diagram reconstruction. In *Proceedings of the 2nd Workshop on Advances in Formal Concept Analysis for Knowledge Discovery in Databases (FCAKDD)*, pages 32–43, 2002.

[8] S. Kuznetsov and S. Ob'edkov. Algorithms for the Construction of the Set of All Concept and Their Line Diagram. preprint MATH-AL-05-2000, Technische Universität, Dresden, June 2000.

[9] G. Stumme, R. Taouil, Y. Bastide, N. Pasquier, and L. Lakhal. Computing Iceberg Concept Lattices with Titanic. *Data and Knowledge Engineering*, 42(2):189–222, 2002.

[10] P. Valtchev and R. Missaoui. Building concept (Galois) lattices from parts: generalizing the incremental methods. In H. Delugach and G. Stumme, editors, *Proceedings, ICCS-01*, volume 2120 of *Lecture Notes in Computer Science*, pages 290–303, Stanford (CA), USA, 2001. Springer-Verlag.

[11] P. Valtchev, R. Missaoui, and P. Lebrun. A partition-based approach towards building Galois (concept) lattices. *Discrete Mathematics*, 256(3):801–829, 2002.

[12] F. Vogt and R. Wille. TOSCANA – a graphical tool for analyzing and exploring data. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing*, volume 894 of *Lecture Notes in Computer Science*, pages 226–233. Springer-Verlag, 1994.