Interactive Visualisation of Formal Concept Lattices

Tim Pattison

tim.pattison@defence.gov.au
Defence Science & Technology Organisation
West Ave, Edinburgh
South Australia 5111

Abstract. Formal Concept Analysis (FCA) is suitable for use within organisations at different levels of maturity in information management. It takes as input a bigraph, into which both structured and unstructured data can be readily transformed, and produces a multiple-inheritance type hierarchy suitable for formal knowledge representation. Accordingly, FCA has been widely applied in areas such as information retrieval, knowledge discovery and knowledge representation. The multiple-inheritance hierarchy produced by FCA is a complete lattice which can be represented as a labelled, directed, acyclic graph (DAG). We adopt a visual analytic approach to FCA by combining computational analysis with interactive visualisation. Scaling FCA to the interactive analysis of large data sets poses two fundamental challenges: the time required to enumerate the vertices, arcs and labels of the lattice DAG; and the difficulty of meaningful and responsive user interaction with a large digraph. This paper briefly describes three software prototypes which address aspects of these scalability challenges.

1 Introduction

Formal Concept Analysis (FCA) [1] derives a multiple-inheritance type hierarchy from a *formal context.* A formal context is a bigraph, consisting of a set of *object* vertices, a set of *attribute* vertices, and edges specified by a binary relation between these two sets. The "types" derived by FCA correspond to maximal bicliques in this bigraph, and are known as *formal concepts.* Each formal concept consists of a set of objects, called its *extent*, and a set of attributes, called its *intent*, which are fully interconnected and jointly maximal: neither objects nor attributes can be added while preserving full interconnection. The set of formal concepts, when partially ordered by set inclusion on their extents, forms a complete lattice. This lattice can be efficiently represented as a single-source, single-sink, labelled, directed acyclic graph (DAG), whose vertices are formal concepts, and whose adjacency relation is the transitive reduction [2] of the ordering relation.

The resultant multiple-inheritance hierarchy of formal concepts constitutes a useful generalisation of a hierarchy for applications such as the storage and retrieval of data objects using keywords or tags, the representation of a Description Logic subsumption hierarchy [3], or the partial ordering of closed frequent item sets in association mining. Accordingly, FCA has been widely applied in such disparate fields as information retrieval, knowledge discovery and knowledge representation [4].

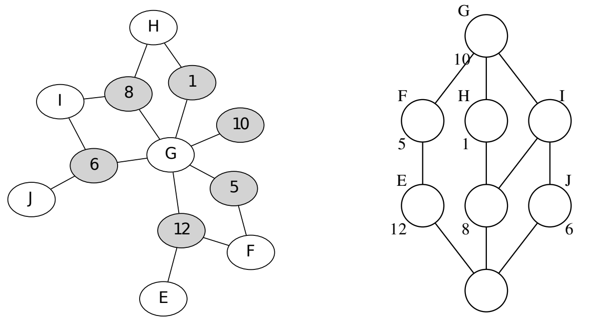
Formal Concept Analysis is an analytic technique suitable for organisations at different levels of maturity in information management. For those who primarily retrieve and read unstructured text, the context bigraph is equivalent to the "bag of words" representation common to a number of statistical techniques for natural language processing, such as Latent Semantic Analysis [5]. For those analysing user-tagged data, including various forms of social media, the tags are attributes associated with the media objects of interest. FCA constitutes a form of association mining for structured data such as the membership of people in organisations or communities of interest. For organisations aspiring to automated reasoning, the output of FCA is an empirically-derived subsumption hierarchy suitable for use in Description Logics.

"Visual analytics is the science of analytical reasoning facilitated by interactive visual interfaces," which, inter alia, "seeks to marry techniques from information visualisation with techniques from computational transformation and analysis of data" [6]. We adopt a visual analytic approach to FCA by combining computational analysis with interactive visualisation. Scalability is a key challenge for visual analytics. Algorithms must scale to large data sets, visualisations must make efficient and intelligible use of screen real-estate, and both must be responsive for interactive use. The number of formal concepts derived from a formal context is bounded above by an exponential function of the number of objects and attributes in that context. Consequently, two fundamental challenges confront those who wish to scale FCA to the interactive analysis of large data sets: the time required to enumerate the vertices, arcs and labels of the lattice DAG; and the difficulty of meaningful and responsive user interaction with a large lattice digraph.

This paper is organised as follows. Section 2 provides a graph-theoretic introduction to Formal Concept Analysis. Section 3 undertakes a brief survey of techniques aimed at improving the scalability of FCA for more responsive visualisation and interaction. Section 4 briefly presents three techniques and associated software prototypes through which the Defence Science and Technology Organisation has addressed selected aspects of FCA scalability.

2 Graph-theoretic introduction to FCA

A formal context (G, M, I) is a bipartite graph, or bigraph, with object vertex set G, attribute vertex set M, and undirected edge set $I \subseteq G \times M$. Each edge is adjacent to one object and one attribute vertex. Each object and attribute vertex has a unique label which derives from the domain of application. For an information retrieval domain, for example, the object labels may be document titles and the attribute labels keywords. Figure 1a shows an example formal context, in which the object and attribute vertices have numerical and alphabetic labels respectively.



(a) Bigraph (b) Hasse diagram

Fig. 1: Bigraph and Hasse diagram (v.i.) for example formal context.

A sub-context (G', M', I') of the formal context (G, M, I) is a bigraph consisting of a subset $G' \subseteq G$ of its objects, a subset $M' \subseteq M$ of its attributes, and the subset $I' = I \cap (G' \times M')$ of its edges adjacent to those object and attribute vertices.

2.1 Formal concepts

A formal concept consists of a set of objects, called the extent, and a set of attributes, called the intent, which form a maximal biclique in the context bigraph. For example, $(\{\mathtt{5}, \mathtt{12}\}, \{\mathtt{F}, \mathtt{G}\})$ is a formal concept in the formal context of Figure 1a, since both attributes in its intent $\{\mathtt{F}, \mathtt{G}\}$ are connected to both objects in its extent $\{\mathtt{5}, \mathtt{12}\}$, and no other objects or attributes can be added while preserving full inter-connection. Two concepts are said to be comparable iff the extent of one is a subset of the extent of the other.

2.2 Concept lattice DAG

FCA produces a set of formal concepts which are, or can be, partially-ordered by extent set inclusion. This partially-ordered set forms a complete lattice [7], which includes *inter alia* a unique maximum element, called the supremum, and a unique minimum element, called the infimum.

This complete lattice can be represented as a DAG, in which each vertex represents a formal concept, and each arc connects a lower neighbour in the partial order to its upper neighbour. This DAG has a single source vertex, corresponding to the infimum of the lattice, and a single sink vertex, corresponding to the supremum. Two concepts are comparable iff there is a directed path between their corresponding vertices in the DAG.

2.3 Hasse diagram

A Hasse diagram is a layered drawing of this DAG in which the vertical component of each arc is upwards on the page. This convention aids the interpretation of the partial ordering and obviates the need to explicitly indicate the direction of each arc. The source [sink][1] vertex appears at the bottom [top] of the diagram, and all other vertices are assigned to intervening layers. Figure 1b shows the Hasse diagram resulting from FCA of the formal context shown in Figure 1a.

Each concept bears an attribute label in the Hasse diagram, and is said to be an attribute concept, iff its extent is the set of objects adjacent to that attribute in the context bigraph. Similarly, each concept bears an object label, and is said to be an object concept, iff its intent is the set of attributes adjacent to that object. For example, the top vertex in Figure 1b is an attribute concept for attribute `G` and an object concept for object `10`. Attribute [object] labels are placed above [below] the labelled concept.

A concept inherits the attributes [objects] appearing as labels on comparable concepts above [below] it in the Hasse diagram. The vertex having attribute label set $\{\mathtt{F}\}$ and object label set $\{\mathtt{5}\}$ in Figure 1b corresponds to the concept having extent $\{\mathtt{5}, \mathtt{12}\}$ and intent $\{\mathtt{F}, \mathtt{G}\}$. In addition to its own object and attribute labels, it inherits attribute `G` from its upper neighbour and object `12` from its lower neighbour.

3 Layout, visualisation and interaction

This section provides a brief survey of techniques aimed at improving the scalability of FCA for more responsive visualisation and interaction.

3.1 Reducing DAG size

The most obvious approach to improving Hasse diagram layout, visualisation and interaction, is to reduce the number of formal concepts. Querying and filtering the input context to remove objects and attributes which are not of interest will expedite concept enumeration and reduce the number of labels on the Hasse diagram, but is not guaranteed to reduce the number of concepts [8]. Another means of achieving this objective is to impose a threshold on extent set cardinality, so that screen real-estate and user attention are reserved for formal concepts which represent suitably large subsets of the objects in the formal context. The partial order amongst these frequent closed item sets is referred to as an *iceberg* lattice. Algorithms exist [9] which exploit the monotonicity of the constraint on extent cardinality to expedite enumeration of the formal concepts.

[1] Square brackets are used throughout this paper to indicate that a sentence is true both when read without the bracketed terms and when read with each bracketed term substituted for the term which precedes it.

3.2 Layout of Hasse diagram

Standard algorithms [10, 11] and genetic variants (see e.g. [12]) exist for assigning the vertices of a DAG to layers and ordering them within each layer to improve aesthetic criteria such as edge crossings. In the present case of a lattice digraph, layer assignment is constrained by the maximum path length of a vertex from the source, and to the sink, vertex. The assignment of vertices to layers in the Hasse diagram is typically under-constrained by the partial order, so that both layer assignment and horizontal order within a layer can be permuted when adjusting the graph layout to optimise aesthetic criteria. This graph layout problem has combinatorial complexity [11].

3.3 Interactive visualisation

Usability testing of FCA applied to the management of email has demonstrated that users can successfully interpret Hasse diagrams [13]. However, the combinatorial explosion of concepts with increasing size of the formal context poses challenges for the layout and visualisation of, as well as interaction with, the lattice digraph. On-demand construction and layout of the entire lattice digraph cannot be achieved in interactive timescales for large lattices, so that either prior or user-guided construction and layout is required to support responsive interaction[2]. For contexts of even moderate size, the potentially large number of resultant vertices and arcs compete for limited screen real estate and challenge user comprehension.

To help the user manage this problem of scale, interactive exploration, as opposed to static presentation, of the Hasse diagram is essential. Information visualisation techniques such as pan and zoom, focus-plus-context, details-on-demand and structural navigation [14] can support user interaction with, and comprehension of, large graphs [15]. For example, geometric zooming or distortion of the Hasse diagram [16, 17] can help allocate more screen real-estate to an area of interest. Alternatively, structural navigation of the lattice digraph can be facilitated by presentation of the immediate graph neighbourhood of the current vertex [16, 18, 19], possibly combined with an overview showing where that vertex resides in the full lattice. A third option is an interactive version of nested Hasse diagrams [1, 16], in which each vertex serves as a container within which to display the Hasse diagram for the same object set and (some of) the remaining attributes. In many applications, however, it is not clear *a priori* how best to group the attributes, or how to order the groups for nesting.

3.4 Discovering or imposing tree structure

A range of mature visualisation and interaction techniques exist for tree, as opposed to lattice, data structures [20–22]. Operating system interfaces for the structural navigation of directory hierarchies are ubiquitous, and user intuition is

[2] User-guided construction of the DAG is addressed in section 4.3.

accordingly well established [16]. This intuition can be exploited for visualisation of the concept lattice digraph, provided that a tree structure can be discovered in, or imposed on, the graph.

Any spanning tree of the lattice digraph, which is rooted at the source or sink vertex, would arguably serve this purpose. Melo et al. [23] investigate various criteria by which a single parent can be chosen for each concept. Whereas any given vertex will typically lie on multiple directed paths from the source [to the sink] of the lattice digraph, the corresponding path in a spanning tree is unique. To make it easier to purposefully locate a vertex of interest, or more likely that such a vertex might be encountered during less goal-directed user exploration, each vertex, along with the sub-lattice of which it is the supremum, can be replicated on demand under each of its parent vertices [16, 24].

Another approach to imposing tree structure on a graph to facilitate user interaction is hierarchical clustering or partitioning of its vertex set [15]. Hierarchical clustering involves the recursive application of a graph clustering algorithm to the clusters (sub-graphs) it identifies. Graph clustering involves optimising some measure of cluster quality, such as modularity, which takes into account factors such as the number, or total weight, of intra- versus inter-cluster links. Whilst the global optimisation of modularity is NP complete, sub-optimal solutions can be computed for large graphs in responsive timeframes [25].

A range of techniques and tools exist for browsing hierarchically clustered graphs. Amongst these are structural zooming on inclusion layouts [22], and the GrouseFlocks environment [26] which supports the use and modification of multiple hierarchical clusterings on the same graph.

3.5 Demand for enhanced tool support

There is a clear trend in operating system interfaces towards tagging and querying rather than navigation of a hierarchical file system. Users typically require assistance in recalling or constructing a set of tags with which to retrieve a suitably small set of objects which contains the object(s) of interest. This trend will drive a demand for well-designed user interfaces through which, like trees before them, multiple-inheritance hierarchies become intuitive with use. The conceptually simple generalisation of a tree to allow a vertex to have multiple parents poses significant challenges for user navigation. More generally, scalable visualisation and interaction of multiple-inheritance hierarchies, and in particular of the DAG produced by FCA, remains an open challenge.

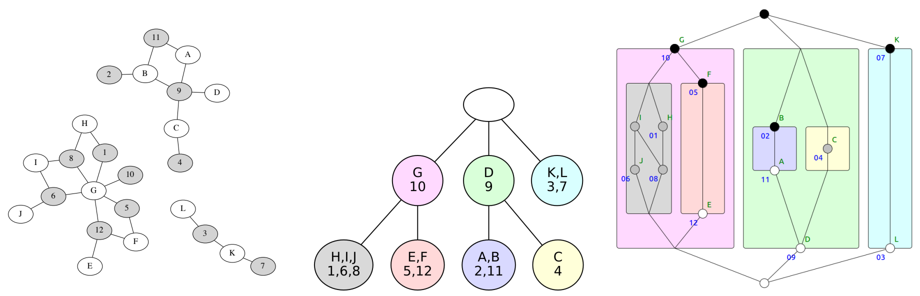
4 Three FCA prototypes

This section briefly presents three software prototypes which the Defence Science and Technology Organisation has developed to address aspects of this scalability challenge.

4.1 Hierarchical parallel decomposition

Carve [27] supports interactive analysis of large formal contexts by discovering and exploiting hierarchical structure which is present in some contexts. That hierarchical structure is used to expedite and enhance both the layout of, and user interaction with, the concept lattice. The Carve algorithm [28] discovers a hierarchical decomposition of amenable contexts and of the corresponding lattice DAG. It produces a tree, representing a partial parallel decomposition of the DAG [11], along with the DAG itself. The decomposition tree for the example context in Figure 2a is shown in Figure 2b. Carve uses this tree both to divide and conquer the computational problem of laying out the DAG as a Hasse diagram, and as a coordinated view to facilitate user interaction with the DAG.

Each vertex of the tree returned by the Carve algorithm corresponds to the lattice digraph for a sub-context identified during hierarchical decomposition of the formal context. This tree can be drawn using an inclusion layout, in which each vertex of the tree is represented as a container within which the containers representing descendant tree vertices are nested. In Figure 2c, these nested containers are shown as coloured boxes whose colour is that of the corresponding vertex in the decomposition tree in Figure 2b. Each leaf vertex of this tree serves as a container for the Hasse diagram of the corresponding trivial or otherwise indivisible sub-lattice digraph.



(a) Context bigraph (b) Decomposition tree (c) Hasse diagram

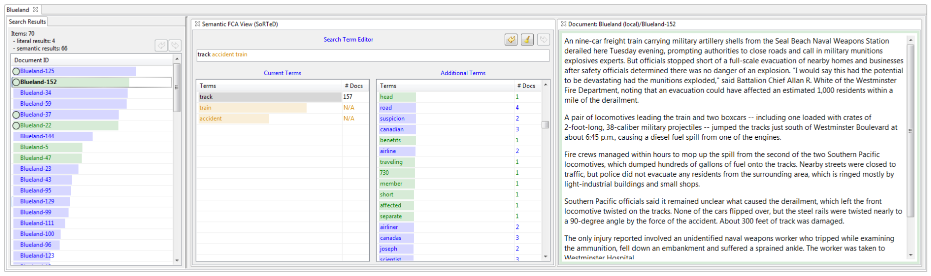
Fig. 2: Example context bigraph, the corresponding decomposition tree, and the modified Hasse diagram with discovered sub-lattices within nested containers.

The sink [source] vertex of the sub-lattice digraph corresponds to a concept in the global context (G, M, I) iff it has an attribute [object] label. Sink [source] vertices which are concepts are shown in Figure 2c as circles with black [white] fill, while those which are not are represented as point junctions of the arcs from their lower [to their upper] neighbours. Such junctions can be seen at the top and bottom of the grey container in Figure 2c. Each sink [source] vertex is connected by an arc to its counterpart in the parent container. In the case where the former vertex is not a concept, it serves as a collection [distribution] point for a "trunk" arc to [from] its counterpart. These trunk arcs reduce clutter by condensing multiple arcs into a single line.

4.2 Structural navigation

The SORTeD prototype supports the retrieval of documents (objects) from a corpus based on queries over the terms (attributes) they contain. User queries are constrained to term combinations which occur in the corpus, and are generalised by removing, or specialised by adding, terms to navigate to comparable concepts. Unlike previous interfaces for structural navigation of the DAG [16, 18, 19], those comparable concepts are not constrained to be neighbours of the current concept. Depicted in Figure 3, the user interface offers valid terms to add or remove from the query. The computational challenge is to compute the set of all concepts comparable to the query concept to facilitate interactive use. Despite structurally navigating the DAG through a keyhole view, the user may be unaware of the DAG's existence, relying instead on intuition established through long-term use of conventional information retrieval interfaces.

In SORTeD, FCA provides a mechanism for literal search over the corpus, with the user interface assisting the construction and refinement of conjunctive Boolean queries. The search results are ranked using Latent Semantic Analysis (LSA) according to their cosine similarity to the search terms [5], and the search terms are ranked according to their cosine similarity to the result set. The latter ranking is less conventional, indicating the comparative relevance of the search terms to the result set, from which the user may judge whether the result set is likely to satisfy their requirements. In addition to offering "valid" search terms which co-occur with the existing search terms to assist the user to refine the query, the interface also offers, ranks and visually distinguishes terms which are only semantically related to the result set. Selecting one of these initiates a literal query in which the selected term is substituted for the existing set of query terms.



Items: 70
- literal results: 4
- semantic results: 66

Document ID
Blueland-125
Blueland-152
Blueland-34
Blueland-59
Blueland-37
Blueland-22
Blueland-144
Blueland-5
Blueland-47
Blueland-23
Blueland-43
Blueland-95
Blueland-129
Blueland-99
Blueland-111
Blueland-100
Blueland-96
Blueland-123

track accident train

Terms	# Docs
track	157
train	N/A
accident	N/A

Terms	# Docs
head	1
road	4
suspicion	2
canadian	3
benefits	1
airline	2
traveling	1
730	1
member	1
short	1
affected	1
separate	1
airliner	2
canadas	3
joseph	2
scientist	3

An nine-car freight train carrying military artillery shells from the Seal Beach Naval Weapons Station derailed here Tuesday evening, prompting authorities to close roads and call in military munitions explosives experts. But officials stopped short of a full-scale evacuation of nearby homes and businesses after safety officials determined there was no danger of an explosion. "I would say this had the potential to be devastating had the munitions exploded," said Battalion Chief Allan R. White of the Westminster Fire Department, noting that an evacuation could have affected an estimated 1,000 residents within a mile of the derailment.

A pair of locomotives leading the train and two boxcars -- including one loaded with crates of 2-foot-long, 38-caliber military projectiles -- jumped the tracks just south of Westminster Boulevard at about 6:45 p.m., causing a diesel fuel spill from one of the engines.

Fire crews managed within hours to mop up the spill from the second of the two Southern Pacific locomotives, which dumped hundreds of gallons of fuel onto the tracks. Nearby streets were closed to traffic, but police did not evacuate any residents from the surrounding area, which is ringed mostly by light-industrial buildings and small shops.

Southern Pacific officials said it remained unclear what caused the derailment, which left the front locomotive twisted on the tracks. None of the cars flipped over, but the steel rails were twisted nearly to a 90-degree angle by the force of the accident. About 300 feet of track was damaged.

The only injury reported involved an unidentified naval weapons worker who tripped while examining the ammunition, fell down an embankment and suffered a sprained ankle. The worker was taken to Westminster Hospital

Fig. 3: SORTeD interface for information retrieval combining FCA and LSA.

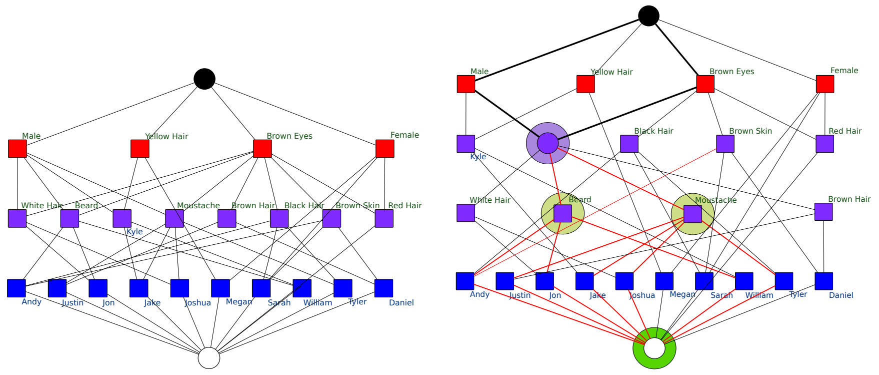
4.3 User-guided FCA

The DAnCe prototype improves the scalability of FCA for interactive use by allowing the user to steer the analysis towards areas of interest, and to halt construction of the DAG as soon as their analytic objectives are satisfied. The resultant DAG will be more task-focused, and, depending on the application, may be considerably smaller, than the lattice DAG for the original context.

We have developed the DAnCe prototype to explore this possibility, modifying a top-down algorithm for concept enumeration to respond to user control and guidance. Instead of autonomous enumeration of all concepts followed by batch-mode construction of the DAG and corresponding Hasse diagram, DAnCe allows the user interactive control over the process of concept enumeration and provides dynamic, incremental update of the Hasse diagram. In addition to being able to start, stop, restart and step through the process of concept enumeration, the user can: select a concept of interest and prioritise the enumeration of comparable concepts which are below it in the lattice; or select multiple concepts and prioritise the generation of the concepts which correspond to the set intersections of their intents and extents. Each vertex and arc is displayed either as soon as it is discovered, or in a batch-mode update of the Hasse diagram after a specified number of steps of the enumeration algorithm.

Visualisation challenges faced by DAnCe include: ensuring intelligible layout of the partially-constructed diagram and maintaining the user's mental model while vertices and arcs are added; and modifying the labelling scheme described in Section 2.3 to allow consistent interpretation when some DAG vertices and edges have yet to be discovered. DAnCe maintains the complete Hasse diagram for the partial order amongst the concepts generated to date. Figure 4 shows mock-ups of this Hasse diagram for an example formal context consisting of people and their physical attributes. Figure 4a depicts the state of the Hasse diagram first presented to the user. By this stage, all of the attribute and object concepts have been generated by the concept enumeration algorithm, labelled, and laid out to establish the framework for insertion of subsequent concepts.

Figure 4b shows the result of the user selecting in this diagram the attribute concepts for "Beard" and "Moustache", which are highlighted in response with small grey halos. This multiple selection triggers the calculation of the extent and intent intersections for the selected concepts. The former corresponds to the infimum, which is accordingly highlighted with a green halo; the latter corresponds to a new concept, which is consequently inserted into the Hasse diagram and highlighted with a purple halo. Since this extent intersection has path length 2 from the supremum, a new row has been inserted to accommodate concepts with path length 3, and the selected concepts demoted to it. The extent intersection is inserted into row 2 at ordinal position 2 of 5; this position is based on the horizontal barycentre of its associated layer 1 ancestors (atoms) and layer 5 descendants (co-atoms), which are predominantly to the left of the centreline.



(a) Initial diagram (b) Multiple selection

Fig. 4: The initial Hasse diagram and the result of multiple selection.

5 Summary

In this paper I have described, in graph-theoretic terms, the analytical technique of FCA, which is applicable to a range of clients of the Defence Science and Technology Organisation (DSTO). The scalability challenges of construction and interactive visualisation of the resultant DAG have been described, along with three prototype tools developed by DSTO to address aspects of these challenges.

Acknowledgement The author gratefully acknowledges the contributions of Aaron Ceglar and Derek Weber to the development of the prototypes and screenshots described in this paper. Graphviz was used in the preparation of most graph drawings.

References

1. Wille, R.: Restructuring lattice theory: An approach based on hierarchies of concepts. In Rival, I., ed.: Ordered sets. Volume 23. Reidel Publishing, Dordrecht–Boston (1982) 445–470
2. Aho, A., Garey, M., Ullman, J.: The transitive reduction of a directed graph. SIAM Journal on Computing **1**(2) (1972) 131–137
3. Russell, S., Norvig, P., eds.: Artificial Intelligence: A Modern Approach. Second edn. Prentice Hall Series in Artificial Intelligence. Prentice Hall (2003)
4. Priss, U.: Formal Concept Analysis in Information Science. Annual Review of Information Science and Technology **40** (2006) 521–543
5. Landauer, T., McNamara, D., Dennis, S., Kintsch, W.: Handbook of Latent Semantic Analysis. Taylor & Francis (2013)
6. Thomas, J.J., Cook, K.A., eds.: Illuminating the Path: The Research and Development Agenda for Visual Analytics. IEEE Press (2005)

7. Davey, B.A., Priestley, H.A.: Introduction to Lattices and Order. second edn. Cambridge University Press, England (2002)
8. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer-Verlag New York, Inc. (1997)
9. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L.: Computing Iceberg concept lattices with Titanic. Data and Knowledge Engineering **42**(2) (2002) 189–222
10. Sugiyama, K., Tagawa, S., Toda., M.: Methods for visual understanding of hierarchical system structures. IEEE Transactions on Systems, Man, and Cybernetics **11**(2) (1981) 109–125
11. Di Battista, G., Eades, P., Tamassia, R., Tollis, I.: Graph drawing: Algorithms for the visualization of graphs. Prentice Hall PTR, Upper Saddle River, NJ, USA (1998)
12. Owais, S., Gajdoš, P., Snášel, V.: Usage of genetic algorithm for lattice drawing. In Bělohlávek, R., Snášel, V., eds.: Concept Lattices and Applications 2005. (2005) 82–91
13. Eklund, P.W., Ducrou, J., Brawn, P.: Information visualization using concept lattices: Can novices read line diagrams? In Eklund, P., ed.: Proc. of the 2nd Int. Conference on Formal Concept Analysis, Springer-Verlag (2004) 57–72
14. Card, S., Mackinlay, J., Shneiderman, B.: Readings in information visualization: Using vision to think. Morgan Kaufmann (1999)
15. Herman, I., Melançon, G., Marshall, M.S.: Graph visualization and navigation in information visualization: a survey. IEEE Transactions on Visualization and Computer Graphics **6** (2000) 24–43
16. Carpineto, C., Romano, G.: Exploiting the potential of concept lattices for information retrieval with CREDO. Journal of Universal Computing **10**(8) (2004) 985–1013
17. Melo, C., Bezerianos, A., Le-Grand, B., Aufaure, M.A.: Cubix: A visual analytics tool for Formal Concept Analysis. In: 23ième Conférence Francophone Sur l'IHM (IHM 2011), Demo. Sophia-Antipolis, France (2011)
18. Ducrou, J., Eklund, P.: Browsing and searching MPEG-7 images using Formal Concept Analysis. In: Proceedings of the 24th IASTED International Conference on Artificial Intelligence and Applications (AIA'06), Innsbruck, Austria, ACTA Press (2006) 317–322
19. Wray, T., Eklund, P.: Exploring the information space of cultural collections. In Valtchev, P., Jäschke, R., eds.: Formal Concept Analysis: 9th International Conference, ICFCA 2011. Springer Verlag, Nicosia, Cyprus (May 2011)
20. Shneiderman, B.: Tree visualization with treemaps: a 2-D space-filling approach. ACM Transactions on Graphics **11**(1) (Jan 1992) 92–99
21. Lamping, J., Rao, R., Pirolli, P.: A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '95, New York, NY, USA, ACM Press/Addison-Wesley Publishing Co. (1995) 401–408
22. Pulo, K., Eades, P., Takatsuko., M.: Smooth structural zooming of h-v inclusion tree layouts. In: Proceedings of International Conference on Coordinated & Multiple Views in Exploratory Visualization. (2003)
23. Melo, C., Le-Grand, B., Marie-Aude, A., Bezerianos, A.: Extracting and visualising tree-like structures from concept lattices. In: Proceedings of the 15th International Conference on Information Visualisation. (2011) 261–266

24. Nauer, E., Toussaint, Y.: CreChainDo: An iterative and interactive web information retrieval system based on lattices. International Journal of General Systems **38**(4) (May 2009) 363–378
25. Clémençon, S., Arazoza, H.D., Rossi, F., Tran, V.C.: Hierarchical clustering for graph visualization. In: Proceedings of XIXth European Symposium on Artificial Neural Networks (ESANN 2011), Bruges, Belgium (April 2011) 227–232
26. Archambault, D., Munzner, T., Auber, D.: Grouseflocks: Steerable exploration of graph hierarchy space. IEEE Transactions on Visualization and Computer Graphics **14**(4) (July/August 2008) 900–913
27. Pattison, T., Weber, D., Ceglar, A.: Enhancing layout and interaction in Formal Concept Analysis. In: 2014 IEEE Pacific Visualization Symposium (PacificVis). (March 2014) 248–252
28. Pattison, T., Ceglar, A., Weber, D.: Efficient, interactive Formal Concept Analysis through recursive context partitioning. Journal of Universal Computer Science (2014) To be submitted.