# Data Weeding Techniques Applied to Roget's Thesaurus

Uta Priss, L. John Old

Edinburgh Napier University, School of Computing,
www.upriss.org.uk, j.old@napier.ac.uk

**Abstract.** It can be difficult to automatically generate "nice" graphical representations for concept lattices from lexical databases, such as Roget's Thesaurus, because the data sources tend to be large and complex. This paper discusses a variety of "data weeding" techniques that can be applied in order to reduce the size of a concept lattice, first in general, and then with respect to Roget's Thesaurus. The aim is that resulting lattices should display neither too much, nor too little information, independently of which search terms have been entered by a user.

## 1 Introduction

Large and complex concept lattices provide a challenge for Formal Concept Analysis (FCA) because they can be difficult to display and navigate by users. Lattices that are automatically derived from lexical databases, such as Roget's Thesaurus, tend to be large and complex. What is needed is some purpose-driven manner of size reduction and selection of subsets of the data. Decisions about which data to select resemble a form of filtering analogous to "weeding" in a garden, because whether a plant is considered useful or a "weed" does not usually depend on structures of the plant itself but solely on whether it is desired in a location. Thus we define "data weeding" techniques as techniques that select data from a given set based on the specific needs and purpose of an application.

A variety of existing Formal Concept Analysis methods can be called data weeding techniques. These techniques, for example, select subsets of the data or reduce the visual complexity of the concept lattice. The choice of the technique usually relies on the type of application. This paper provides an overview of data weeding techniques in general, and then analyses their applicability to Roget's Thesaurus. The goal for this research is to automatically generate lattices from Roget's Thesaurus in an on-line interface[1] in a manner that adjusts the data weeding techniques for each request.

Data weeding techniques for FCA can be categorised into four types as shown in Table 1. The first type, visual reduction techniques, are techniques that change how the data is displayed without changing the mathematical structure of the underlying concept lattices. The second type, faceting and plain scaling, are techniques which lead to a division of the original concept lattices into smaller lattices without information loss. The division should be meaningful with respect to the content of the lattice. For example, if the attributes can be naturally subdivided into a set of partitions, then a separate lattice can be drawn for each partition of the data. In FCA, this is called "plain"

[1] http://www.roget.org

scaling (Ganter & Wille, 1999) for single-valued contexts. The third type, pruning and restricting, consists of techniques which reduce concept lattices by removal of objects, attributes or concepts usually based on statistics. The fourth type, decomposition and general scaling, decomposes a lattice and at the same time reduces complexity, for example, by aggregating objects or attributes. The third and fourth types usually cause some loss of information because of summarisation, abstraction or reduction.

**Table 1.** Four types of data weeding techniques

| Type of reduction | Effect on lattice | Loss of information |
|---|---|---|
| Visual reduction | Lattice structure unchanged | None |
| Faceting and plain scaling | Lattice is divided | None |
| Pruning and restricting | Some concepts are removed | Possible |
| Decomposition and general scaling | Lattice is divided | Possible |

The following four sections provide an overview of the four types of data weeding techniques. Section 6 discusses the applicability of these techniques to a lexical database of Roget's Thesaurus.

## 2 Visual reduction techniques

The first set of data weeding techniques are visual reduction techniques, which change how the data is displayed without modifying the underlying structure. Many FCA techniques for visual reduction have been described in the literature:

**Clarifying** and reducing (Ganter & Wille, 1999) result in omitting labels (and the corresponding objects and attributes) without changing the lattice structure. A clarified lattice has at most one object and at most one attribute attached to each concept in the lattice diagram. In the example at the top of Fig. 1, the concept furthest to the left has two objects ("daffodil" and "sunflower") and the concept furthest to the right has two attributes ("purple") and ("dark red"). In the clarified version, "sunflower" and "purple" are removed. If the clarification is achieved using FCA software, then usually the object/attribute that is listed first in the formal context is retained. This choice may not be optimal with respect to the application.

**Reducing** will further remove all objects that are not at join-irreducible concepts and all attributes that are not at meet-irreducible concepts. This means that in the lattice diagram objects are only attached to nodes that have exactly one edge coming from below and attributes are only at nodes that have exactly one edge from above (Fig. 1). Again, reduction may not be meaningful in all applications. In some applications where the data consists of exemplars and features, the remaining attributes, after reduction, can be considered characteristic, defining features and the remaining objects prototypical examples of the data because of their location at meet- or join-irreducible concepts.

The display of labels and nodes in the diagram can be modified by using **lists, counts and colours**. While it is possible in a manually drawn lattice diagram to carefully place each label in a position where it does not overlap with anything else, the

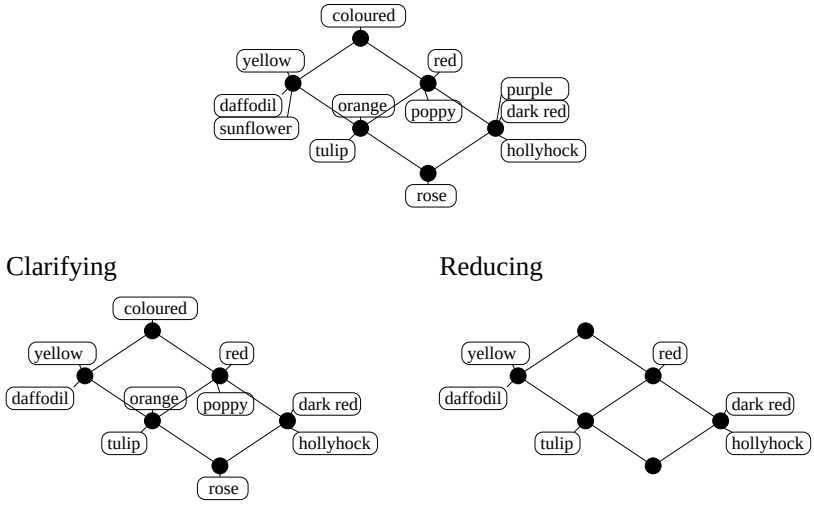


**Fig. 1.** Visual reduction techniques: clarifying and reducing

automatic placement of labels in a non-overlapping manner is a major challenge for FCA software. Thus, visual reduction strategies for the display of labels are beneficial, both for the algorithms used by the software and for the users who will be provided with diagrams that are easier to read. Different FCA software tools use different strategies for visual reduction. The FcaStone[2] software, for example, provides an option to draw the concept nodes as boxes which contain lists of objects and attributes truncated to 30 characters. In ToscanaJ[3], objects and attributes can be displayed as a count (showing the number of objects and attributes belonging to a concept instead of their names) or as a scrollable list. These methods reduce the physical space that is occupied by the labels in the display. There are different methods for how the objects and attributes of each concept are counted: either as absolute counts or as relative frequencies; either containing the full extents and intents; or only counting the objects/attributes that are directly attached to a concept in the diagram. The nodes in ToscanaJ can be coloured based on a count of the objects. In the ConExp software[4], a similar effect is achieved by varying the size of the nodes instead of using colours as in ToscanaJ. ConExp uses colours also to show whether a node has objects or attributes directly attached. This feature is especially useful if the display of the labels is turned off.

Some FCA applications do not use lattice diagrams at all, but instead use the lattice structure only for internal algorithms or use **textual displays**. An example is the Credo[5] software. Credo is a meta search engine that calculates a concept lattice for the results of a Yahoo query. The lattice is not displayed graphically but instead as an expandable

[2] `http://fcastone.sourceforge.net`
[3] `http://toscanaj.sourceforge.net`
[4] `http://conexp.sourceforge.net`
[5] `http://credo.fub.it`

tree hierarchy. Any lattice can be displayed as a tree by creating multiple copies of any node that has more than one upper neighbour as shown in Fig 2. The first display in Credo consists of the top node of the lattice and its immediate lower neighbours, which are displayed as a bulleted list that is slightly indented compared to the top node. If a user clicks on any of the nodes, the immediate neighbours underneath this node are displayed as a further indented list (but only up to three levels deep). Clicking on the top node will collapse the expanded sublist. Credo's display is similar to tree displays used for directory listings of folders and files on a computer. Most users will be familiar with such displays. The drawback for these approaches is that the full lattice structure is not immediately visible. Only the relationship between a node and its immediate neighbours is presented.

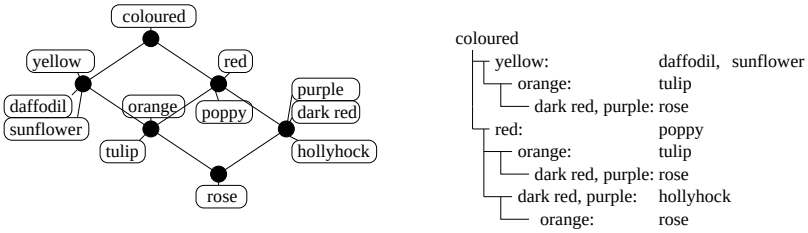


**Fig. 2.** Representing a lattice as a tree hierarchy

**Fish-eye** and zoom are graph visualisation techniques that are used in many graphical displays. Fish-eye displays (Furnas, 1981) enlarge a focal point of a display while gradually reducing the rest of the display (as in Fig. 3). The idea is that users can move the enlarged focal area across a display similar to using a magnifying glass. The focal area can be read in detail while the non-focal area provides structure without detail. The use of fish-eye views for concept lattices was first suggested by Godin et al. (1989) and implemented by Carpineto & Romano (1995). The notion of **zooming** is used with different meanings with respect to concept lattices. In FCA software, "zooming" usually means to enlarge the display. If the lattice is larger than the display window, only parts of the lattice will be visible after zooming in. Thus, this is similar to fish-eye displays except that the non-focal parts are omitted instead of reduced. The notion of "zooming" is also used in a different meaning for the navigation between scales as discussed in the next section.

**Moving displays** are the final visual reduction technique mentioned in this section. Moving the nodes in a lattice diagram can significantly change the visual complexity of the diagram. From some "angles" a lattice may have a much more complicated display (i.e., more edge crossings, less symmetry) than from other angles. Many FCA tools let users move individual nodes of the lattice. Some FCA software allows for lattices to be rotated. Other FCA software lets users move parts of a lattice (ideals or filters) at the same time, which helps in the detection of symmetries and other structures in the

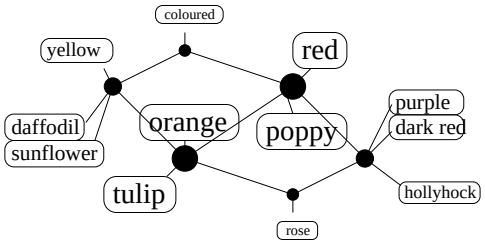


**Fig. 3.** Fish-eye visualisation

lattice. (More details and comparisons of FCA software can be found in Tilley (2004) and Priss (2008a)).

## 3 Faceting and plain scaling

Apart from visual reduction techniques that modify the display of the lattices, but not their internal structures, there are data weeding techniques that do change the lattice structures. **Faceting and plain scaling** are techniques which subdivide the set of attributes into smaller sets. The lattice of such a smaller set is traditionally called a "scale" (Ganter & Wille, 1999). Different scales can be combined in nested lattice diagrams and can be interactively explored in Toscana systems (Vogt & Wille, 1995). In this section, only scaling of "single-valued contexts" is considered. Priss (2008b) argues that the idea of "plain scaling" as described in FCA occurs in several other disciplines under other names. In particular, plain scaling is very similar to the idea of "faceting" in library and information science. For both scales and facets it is usually required that the subdivisions are carefully selected (usually manually) and meaningful. Meaningful subdivisions will group attributes based on their types, shared features, and so on. For example, one scale could contain size attributes while another contains colour attributes. Each scale then arranges the objects based on its particular aspect or viewpoint.

As mentioned above, the notion of "zooming" is used to describe the navigation from an "outer scale" into an "inner scale" in a nested lattice diagram in Toscana systems (Vogt & Wille, 1995). Roth et al. (2008) modify this notion of "zoom" by applying local criteria to the calculation of inner scales instead of a global algorithm. In Stumme's (1996) approach to "local scaling" some concepts are expanded with inner scales, while others are not.

Historically, faceted systems in library science have been less popular with end users because it can be difficult to use such systems in a paper-based environment. To some degree facets or scales are only really useful if they are presented via a software interface which lets users explore the relationships interactively. This is because it is impossible to display all of the facets simultaneously for all but the smallest data sets. As a rule of thumb, three levels of nesting (or three scales) is the maximum that can be shown simultaneously. Thus, the information contained in a formal context and its

scales is best explored interactively by starting with a few scales, deciding how to nest them, and then navigating from one scale to the other.

Besides the Toscana systems, as described by Vogt & Wille (1995), there have been recent software developments in the library and information science community which are potentially interesting for FCA because of the similarity between scales and facets. This has been discussed in detail by Priss (2008b). Such software often presents different scales side by side instead of nested. The effect of user selections on one scale is instantly applied to the other scales by the software. It would be interesting to conduct a user study comparing different techniques of representing scales. This could lead to new developments in FCA software.

## 4 Pruning and restriction

Data weeding techniques of **pruning** reduce a lattice in a manner that is usually based on statistical measures. The effect is a removal of concepts usually from the bottom of the lattice. In many cases the lattice structure is changed significantly. Some forms of pruning result in ordered sets which are not lattices.

Kuznetsov (2007) defines a notion of "stability" of a formal concept based on his earlier work on stability in similarity operations (Kuznetsov, 1990). Roughly, the stability index of a concept $C$ is based on counting the number of those subsets of the extent of $C$ whose intent equals the intent of $C$, divided by some number related to the size of the extent. The idea behind this is to determine how many objects in the extent are necessary and sufficient to creating the concept. This is because a concept can also have many objects that are shared with many other concepts and are not defining for this particular concept. The ConExp software (mentioned above) implements a similar but slightly different notion of "stability" which, according to ConExp's user guide, calculates for each concept the minimal number of objects needed to be removed so that the intent of this concept disappears from the concept lattice.

Pruning is then the process of removal of "less stable" concepts based on any of the stability notions. All concepts that are less stable than a user-defined threshold are removed. Roth et al. (2008) use a slight variant of Kuznetsov's definition of stability for their pruning, in combination with their notions of nesting and zooming. As mentioned above, Roth et al.'s notion of "zooming" is slightly different from Vogt & Wille's (1995). They divide the set of attributes based on preferences and use attributes that are considered more important in outer scales. The inner scales are then "pruned" individually, using local instead of global criteria. Belohlavek & Sklenar (2005) propose a different method of pruning based on attribute dependency formulas, which uses an expert-specified hierarchy on the set of attributes.

Last but not least, an iceberg lattice (Stumme et al., 2002) is an order filter consisting of the top most concepts of a concept lattice. Only those concepts are included whose concepts have a "support" that is higher than a threshold. These concepts are called "frequent". The support is calculated as the number of objects of the concept divided by the total number of objects in the formal context. The notion of iceberg lattices was developed in the framework of data mining and is based on what is called "frequent itemsets" in data mining terminology. A variation of this approach is to also include the

immediate lower neighbours of the frequent concepts, which may result in an ordered set that is not an order filter.

Old (2003) describes the notion of **restricting** concept lattices, with respect to neighbourhood lattices. Neighbourhood lattices were first described by Rudolf Wille in an unpublished manuscript and first published by Sedelow & Sedelow (1993) in the context of lexical databases. The operation which underlies the selection of elements in a neighbourhood has been called the "plus operator" (Priss & Old, 2004) as opposed to the "prime operator" used in concept formation. This is because the prime operator applied to a set of objects selects all attributes which are shared among *all* objects in the set, whereas the plus operator selects all attributes that belong to *at least one* of the objects in the set. For a large formal context $A$, the plus operator can be used to derive a smaller "neighbourhood context" $B$ as follows: a "$n$-$m$-neighbourhood" starts with an object from $A$ and has the plus operator applied $(2n-2)$-times to obtain the set of objects of $B$ and $(2m-1)$-times to obtain the set of attributes of $B$. This context $B$ represents the neighbourhood of the object that was used at the start of the operation. Because the plus operator is not a closure operator, a few iterations of the operator can result in a context $B$ whose size is fast approaching the size of $A$. Therefore Old (2003) experiments with "restricted" neighbourhood lattices which modify the plus operator by selecting objects (attributes) which have at least two (three, etc) attributes (objects) instead of at least one.

## 5 Decomposition and general scaling

Ganter & Wille (1999) describe numerous methods for deriving parts and **decompositions** of concept lattices. The methods that are used in actual applications and that are implemented in FCA software are usually only the simplest examples of such constructions. For example, a horizontal decomposition of a lattice is a decomposition into components whose horizontal sum (Ganter & Wille, 1999) is the original lattice. A horizontal decomposition of a lattice refers to the components that a lattice falls into after removing the top and bottom concept. If a plus operator is applied until the sets do not change any further, it yields a horizontal decomposition of the original lattice (Priss & Old, 2006). Horizontal decompositions have been used in software analysis (Snelting, 2005) and other applications. A form of decomposition is also the Semantic Mirrors method. This method was invented by Dyvik (2004) and translated into FCA terminology by Priss & Old (2005). The Semantic Mirrors method is similar to a form of repeated applications of decomposition.

In contrast to plain scaling, in **general scaling** the scales are combined using a "composition operator". This appears to be mostly of theoretical interest. We are not aware of any software implementations of scaling techniques other than plain scaling. Plain scaling is often applied to many-valued contexts, which are transformed into single-valued contexts via the scales. This does imply loss of information.

## 6 Data weeding techniques for Roget's Thesaurus

Before discussing which of the data weeding techniques described above are relevant for Roget's Thesaurus, it shall be explained why Roget's Thesaurus is of interest. Roget's Thesaurus (1911, 1962) is a semantic dictionary that is organised by concepts, rather than words, into a classification tree. The explicit structure of the book consists of three main parts: the top level of the hierarchy represented by the tabular Synopsis of Categories; the Sense Index which continues the hierarchy down to the lowest level; and the Word Index which lists the words in alphabetic order along with their senses ordered by part-of-speech. The senses are represented in the Word Index as references to locations in the Sense Index.

The Sense Index lists the 1,000 or so categories representing the notions found at the most detailed level of the Synopsis. Categories generally occur in pairs as opposed notions, or antonyms. Each category is subdivided into paragraphs which contain groups of words at the lowest level. The words in each group at the lowest level are considered "synonyms" with respect to the thesaurus structure. Each group of synonyms denotes a "sense". A particular occurrence of a word is also called an "entry" of the thesaurus. The notation for senses used in this paper (e.g. 227:1:1) consists of the category number (227), followed by the paragraph number (1) and the synonym group number (1).

Roget's Thesaurus has been studied or used for the automatic classification of text, automatic indexing, natural language processing, word sense disambiguation, semantic classification, computer-based reasoning, content analysis, discourse analysis, automatic translation, and a range of other applications by many different researchers (cf. Old (2003 and 2004) and Priss & Old (2009) for more details). The reason why the Thesaurus has been used in such applications is that it contains an implicit conceptual structure based on the polysemy and synonymy relationships between the word entries and their senses. However, although the explicit structure of Roget's Thesaurus is evident to any reader, the implicit, hidden, or "inner structure" (Sedelow, 1988) is not. FCA can be a useful tool in exploring this inner structure because the relationship between words and senses for the Thesaurus is a very large formal context. Each "entry" corresponds to a cross in this context. Since the formal context of Roget's Thesaurus has about 113,000 objects, 71,000 attributes and 200,000 crosses, some data weeding technique is required in order to extract smaller-sized lattices.

Neighbourhood lattices, as described in the previous section, have been used for the exploration of Roget's Thesaurus since Sedelow & Sedelow (1993). Fig. 4 shows a 2-1 neighbourhood for the word "think" (i.e., the plus operator has been applied twice to retrieve the objects and once to retrieve the attributes). Neighbourhood lattices can be of very different sizes. We have calculated the sizes of all 2-1 and 2-2 neighbourhoods of common words in Roget's Thesaurus (using Roget (1962)). The neighbourhoods range from single-concept lattices (for words such as "amoeba") to a lattice with 118 concepts (713 concepts) for the word "cut" in the case of 2-1 neighbourhoods (2-2 neighbourhoods, respectively). Lattices with less than 5 concepts are not very interesting whereas lattices with more than 35-40 concepts tend to be too large to be visualised. Therefore, different words require different types of neighbourhoods in Roget's Thesaurus. About 100 words (mostly verbs of Anglo-Saxon origin) have 2-1 neighbourhood lattices with more than 35 concepts. These could be benefit from using data weeding techniques.

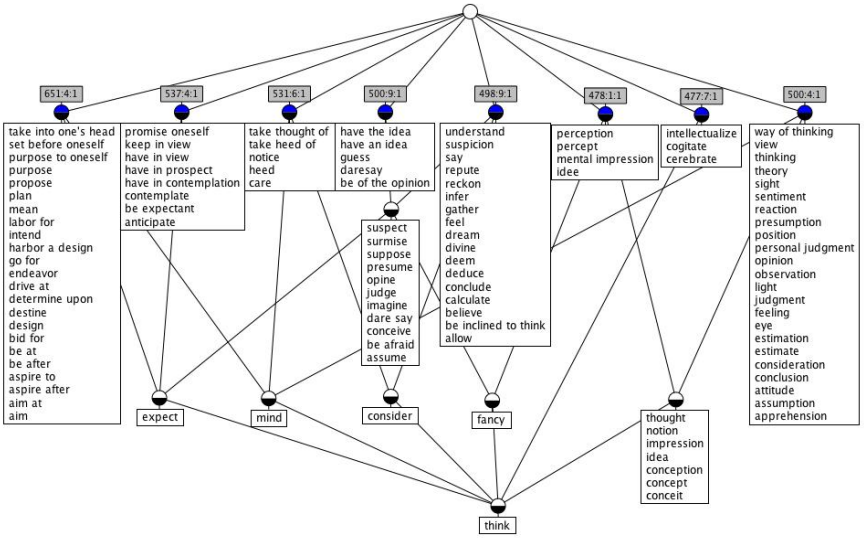


**Fig. 4.** A neighbourhood lattice from Roget's Thesaurus for the word "think"

The on-line interface at `www.roget.org` lets a user explore Roget's Thesaurus by entering a word and then viewing a neighbourhood lattice of that word. The lattices are generated in run-time. The remainder of this section investigates which data weeding techniques are appropriate for automatically adjusting the size of the lattices. Visual reduction techniques depend on the display software that is used because they do not affect the structure of the lattices. The on-line interface uses the "concept as boxes" display feature of FcaStone[6] in order to avoid overlapping labels. Scaling and faceting often depend on manual subdivision of the set of attributes, which is not applicable in this case. Thus, the techniques that are of interest are pruning, restriction and decomposition.

Figs. 5 and 6 show a pruned lattice and an iceberg lattice, respectively, for the example from Fig. 4. Unfortunately, pruning and iceberg lattices appear not to be appropriate data weeding techniques for Roget's Thesaurus. This is because neighbourhood lattices are always of a particular structure. Neighbourhood lattices of type 2-1 always have the original word attached to the bottom node. They normally contain many more objects than attributes (because the plus operator has been applied twice to retrieve the objects). Most of the objects that were added in the last step in a 2-1 neighbourhood lattice are attached to the nodes adjacent to the top node because their distinguishing attributes are not yet part of the neighbourhood context. Dually, most of the attributes in a 2-2 neighbourhood lattice will be attached to nodes that are adjacent to the bottom node. Clearly, the most interesting objects in Fig. 4 are "expect", "mind", "consider",

[6] `http://fcastone.sourceforge.net`

"fancy", and "thought, notion, ..." because they share more than one sense with "think". But these objects are exactly the ones that are pruned away in Figs. 5 and 6. Because of the particular structure of neighbourhood lattices, pruning and iceberg lattices are not appropriate. Pruning and iceberg lattices are based on the relative sizes of the extents, but for neighbourhood lattices these sizes are distorted by the algorithm.

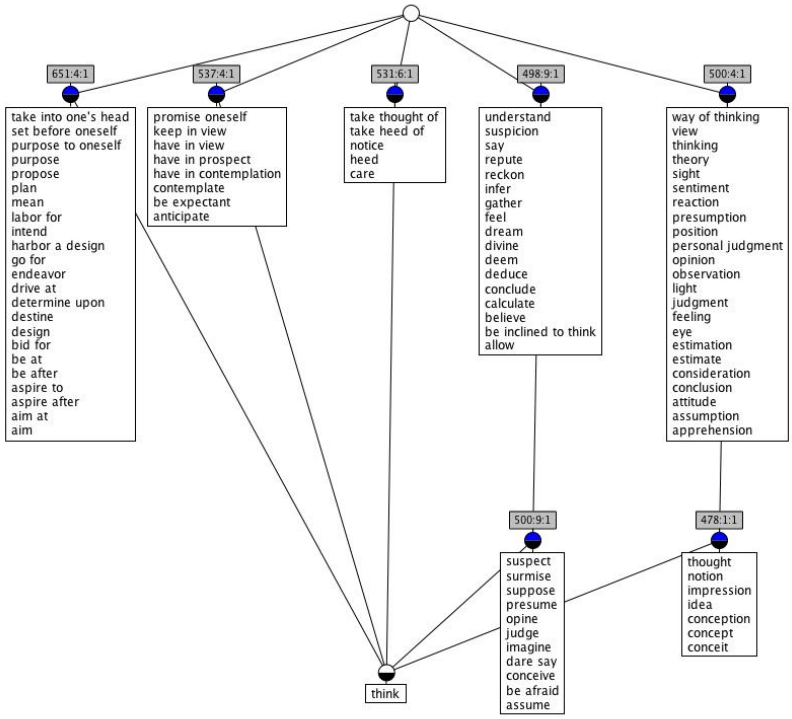


**Fig. 5.** A pruned lattice showing he most stable concepts (using ConExp) of Fig. 4

In a similar fashion, the Semantic Mirrors method (see above) is not of interest for these kinds of lattices because this method essentially looks for symmetries between the objects and attributes. This works well for neighbourhood lattices that are derived from bilingual data (the objects are words from one language, while the attributes are words from a second language). But it does not yield interesting results for Roget's Thesaurus because the structures among words are quite different from the structures among senses. The Semantic Mirrors method applied to the lattice in Fig. 4 would result in a decomposition of the lattice into single-concept lattices. On the other hand, other forms of decomposition might be of interest. A horizontal decomposition separates the

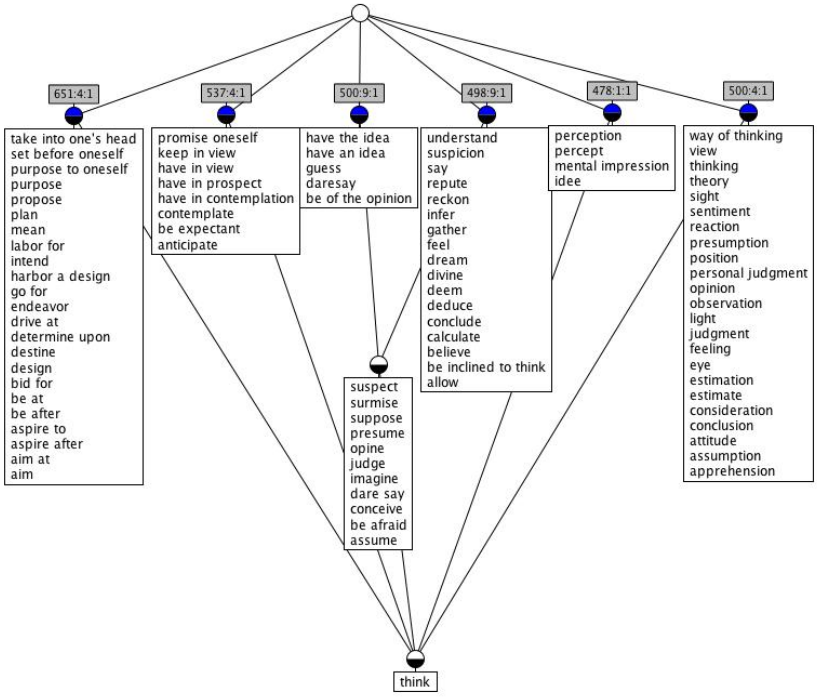


**Fig. 6.** An iceberg lattice for the lattice in Fig. 4

concept with the sense 477:7:1 in Fig. 4 from the rest of the lattice. If horizontal decomposition results in more than one component, this can (but does not have to) indicate that a word is a homograph (Old, 2006), i.e. that the word has two completely unrelated senses. In this case, sense 477:7:1 is a polysemous sense of "think", but not a homograph.

The most promising data weeding technique for neighbourhood lattices of Roget's Thesaurus appears to be restriction (Old, 2003). Fig. 7 shows the restricted lattice of Fig. 4. This lattice maintains the main structures from Fig. 4, but all objects that do not share at least two senses with "think" have been removed. Different degrees of restriction are possible: only keeping words (or senses) that have at least 3 senses (or words), and so on. Incidentally, in retrospect Priss & Old (2009) discovered that the restriction algorithm is very similar to techniques developed for Roget's Thesaurus by the Cambridge Language Research Unit in the 1950s.

The goal for the on-line interface at `www.roget.org` is to generate neighbourhood lattices that are of appropriate sizes and calculated sufficiently fast. A simple strategy could be to use the number of crosses in the context as a rule of thumb for pre-

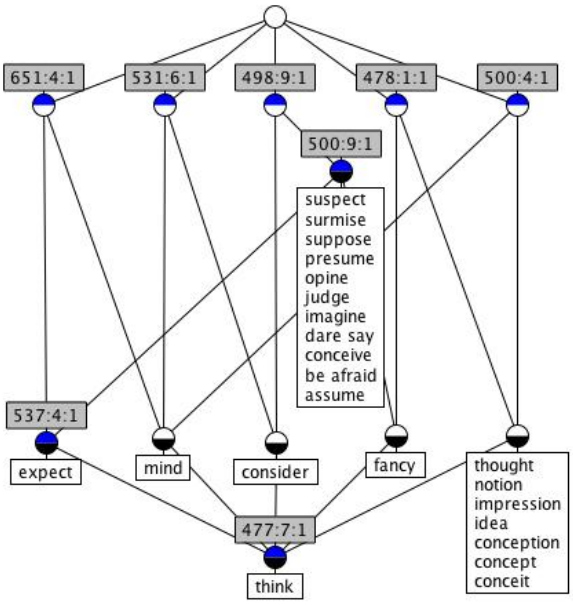


**Fig. 7.** A restricted neighbourhood lattice

dicting the size of the neighbourhood lattices (for example, if the context has more than 100 crosses, then use restriction). Another simple strategy would be to apply restriction if the smaller of the sets of objects and attributes is larger than 20 and the larger one is larger than 80. These strategies have been tried and found to yield reasonably good results for Roget's Thesaurus.

A third, more precise strategy is to calculate the sizes of all neighbourhood lattices (2-1, 2-2 neighbourhoods and restriction) in an off-line mode and to store the results in a database table. Calculating the number of concepts can be slow for large lattices, but because they need to be calculated only once that is not a problem. When a user enters a word into the interface, the size of its neighbourhood lattice can be looked up and be used to select a type of neighbourhood with an appropriate size. Only about 6000 words in Roget (1962) have 2-2 neighbourhood lattices with more than 20 concepts. For about 500 of these words, the 2-1 neighbourhood lattice also has more than 20 concepts. When applying restriction, only about 60 words are left which have more than 30 concepts in their neighbourhood lattices. For the word "cut" which has the largest concept neighbourhood of all words in the Thesaurus, a 2-1 neighbourhood needs to be restricted to objects and attributes that have at least 3 crosses in the context in order to reduce the size of the lattice to 38 concepts. This is a size that can still be graphically

represented. At the other extreme, considering lattices that might be too small, according to Priss & Old (2006) there are about 20,000 words for which the lattice never has more than one concept for any type of neighbourhood. But these words include archaic, foreign and specialist words and phrases, which are unlikely to be entered by users in the on-line interface.

## 7 Conclusion

In summary, data weeding techniques select data from a given set based on the specific needs and purpose of an application. Visual reduction techniques do not change the lattice structure and are mostly a feature of the software interface that is used to draw and display the lattices. Scales and facets rely on meaningful subdivisions of the set of attributes that are often manually derived and not ideally suited for automatically generated lattices. Although pruning is very useful in other applications (for example for clustering), it appears to prune away the wrong concepts with respect to neighbourhood lattices. Some forms of decompositions might be applicable to neighbourhood lattices, but this needs to be investigated further. Experimentation has shown that restriction is useful for neighbourhood lattices because it maintains most of the structure of the lattices while removing objects and attributes.

In general, data weeding techniques appear to be very much dependent on the type of application. There may not be a single type of technique that is suitable for all kinds of lattices. An interesting topic for further research would be to determine whether it might be possible to develop some guidelines that predict which data weeding techniques are appropriate for which types of lattices.

## References


1. Belohlavek, Radim; Sklenar, Vladimir (2005). *Formal Concept Analysis Constrained by Attribute-Dependency Formulas.* In: Ganter & Godin (Eds.), Formal Concept Analysis: Third International Conference, ICFCA 2005, Springer Verlag, LNCS 3403, p. 176-191.
2. Carpineto, Claudio; Romano, Giovanni (1995). *Ulysses: a lattice-based multiple interaction strategy retrieval interface.* In B. Blumenthal, J. Gornostaev & C. Unger (Eds.), Human-Computer Interaction, 5th International Conference, EWHCI'95, Moscow, Russia, Springer Verlag, LNCS 1015, p. 91-104.
3. Dyvik, Helge (2004). *Translations as semantic mirrors: from parallel corpus to wordnet.* Language and Computers, 49, 1, Rodopi, p. 311-326.
4. Furnas, George W. (1981). *The FISHEYE View: A New Look at Structured Files.* Bell Laboratories Technical Memorandum.
5. Ganter, Bernhard; Wille, Rudolf (1999). *Formal Concept Analysis.* Mathematical Foundations. Springer Verlag.
6. Godin, Robert; Gecsei, Jan; Pichet, Claude (1989). *Design of browsing interface for information retrieval.* In N. J. Belkin, & C. J. van Rijsbergen (Eds.), Proc. SIGIR '89, p. 32-39.
7. Kuznetsov, Sergej O. (1990). *Stability as an estimate of hypotheses based on similarity operation* (in Russian). Nauchno-Tekhnicheskaya Informatsiya (NTI), 2, N12, p. 21-29.
8. Kuznetsov, Sergej O. (2007). *On Stability of a Formal Concept.* Annals of Mathematics and Artificial Intelligence, 49, p. 101-115.

9. Old, L. John (2003). *The Semantic Structure of Roget's.* A Whole-Language Thesaurus. PhD Dissertation. Indiana University.
10. Old, L. John (2004). *Unlocking the Semantics of Roget's Thesaurus.* In P. Eklund, (Ed.), Proceedings, Second International Conference on Formal Concept Analysis, Springer Verlag, LNCS 2961, p. 236-243.
11. Old, L. John, (2006). *Homograph Disambiguation using Formal Concept Analysis.* In: R. Missaoui & J. Schmidt (Eds.), 4th International Conference on Formal Concept Analysis, Springer, LNCS 3874, p. 221-232.
12. Priss, Uta; Old, L. John (2004). *Modelling Lexical Databases with Formal Concept Analysis.* Journal of Universal Computer Science, 10, 8, p. 967-984.
13. Priss, Uta; Old, L. John (2005). *Conceptual Exploration of Semantic Mirrors.* In: Ganter; Godin (Eds.), Formal Concept Analysis: Third International Conference, ICFCA 2005, Springer Verlag, LNCS 3403, p. 21-32.
14. Priss, Uta; Old, L. John (2006). *An application of relation algebra to lexical databases.* In: Schaerfe, Hitzler, Ohrstrom (Eds.), Conceptual Structures: Inspiration and Application, Proceedings of the 14th International Conference on Conceptual Structures, ICCS'06, Springer Verlag, LNAI 4068, p. 388-400.
15. Priss, Uta (2008a). *FCA Software Interoperability.* In: Belohlavek; Kuznetsov (Eds.), Proceedings of the Sixth International Conference on Concept Lattices and Their Applications (CLA'08), p. 133-144.
16. Priss, Uta (2008b). *Facet-like Structures in Computer Science.* Axiomathes, 14, Springer Verlag, p. 243-255.
17. Priss, Uta; Old, L. John (2009). *Revisiting the Potentialities of a Mechanical Thesaurus.* In: Ferre; Rudolph (Eds.), Proceedings of the 7th International Conference on Formal Concept Analysis, ICFCA'09, Springer Verlag, LNAI 5548, p. 284-298.
18. Roget, Peter Mark (1962). *Roget's International Thesaurus.* 3rd Edition Thomas Crowell, New York.
19. Roget, Peter Mark (1911). *Roget's Thesaurus.* Available from the Project Gutenberg `http://promo.net/pg`.
20. Roth, Camille; Obiedkov, Sergei; Kourie, Derrick G. (2008). *On succinct representation of knowledge community taxonomies with formal concept analysis.* International Journal of Foundations of Computer Science (IJFCS), 19, 2, p. 383-404.
21. Sedelow, Walter A., Jr. (1988). *Computer-based planning technology: an overview of inner structure analysis.* Sixth Annual Conference on New Technology and Higher Education: Acquisition, Integration, and Utilization.
22. Sedelow, Sally; Sedelow, Walter (1993). *The Concept "concept".* Proceedings of the Fifth International Conference on Computing and Information, Sudbury, Ontario, Canada, p. 339-343.
23. Snelting, Gregor (2005). *Concept Lattices in Software Analysis.* In: Ganter, Stumme, Wille (Eds.): Formal Concept Analysis, Foundations and Applications. Springer Verlag, LNCS 3626, p. 272-287.
24. Stumme, Gerd (1996). *Local Scaling in Conceptual Data Systems.* In: Proceedings of the 4th International Conference on Conceptual Structures: Knowledge Representation as Interlingua, ICCS'96, Springer Verlag, LNCS 1115, p. 308-320.
25. Stumme, G.; Taouil, R.; Bastide, Y.; Pasquier, N.; Lakhal. L. (2002). *Computing Iceberg Concept Lattices with Titanic.* J. on Knowledge and Data Engineering, 42, 2, p. 189-222.
26. Tilley, Thomas (2004). *Tool Support for FCA*. In: Eklund (Ed.), Concept Lattices: Second International Conference on Formal Concept Analysis, Springer Verlag, LNCS 2961, p. 104-111.
27. Vogt, Frank; Wille, Rudolf (1995). *TOSCANA - a graphical tool for analyzing and exploring data.* In: Tamassia; Tollis (Eds.). Graph Drawing. Springer Verlag, LNCS 894, p. 226-233.