An application of relation algebra to lexical databases*

Uta Priss, L. John Old

Napier University, School of Computing,
u.priss@napier.ac.uk, j.old@napier.ac.uk

Abstract. This paper presents an application of relation algebra to lexical databases. The semantics of knowledge representation formalisms and query languages can be provided either via a set-theoretic semantics or via an algebraic structure. With respect to formalisms based on n-ary relations (such as relational databases or power context families), a variety of algebras is applicable. In standard relational databases and in formal concept analysis (FCA) research, the algebra of choice is usually some form of Cylindric Set Algebra (CSA) or Peircean Algebraic Logic (PAL). A completely different choice of algebra is a binary Relation Algebra (RA). In this paper, it is shown how RA can be used for modelling FCA applications with respect to lexical databases.

1 Introduction

Formal Concept Analysis (FCA) is a method for data analysis and knowledge representation that provides visualisations in the form of mathematical lattice diagrams for data stored in "formal contexts". A formal context consists of a binary relation between what are named "formal objects" and "formal attributes" (Ganter & Wille, 1999). FCA can, in principle, be applied to any relational database. Lexical databases, which are electronic versions of dictionaries, thesauri and other large collections of words provide a challenge for FCA software because the formal contexts of lexical databases tend to be fairly large consisting of 100,000s of objects and attributes. An overview of FCA applications with respect to lexical databases can be found in Priss & Old (2004). That paper also provides pointers to future research, for example, with respect to specific implementations of a construct called a "neighbourhood lattice". In this paper, we are continuing the thread of that research and provide a further mathematical analysis supported by empirical data from Roget's Thesaurus.

For the development of a mathematical analysis of structures in lexical databases it is beneficial to make use of existing relational methods. In particular, this paper elaborates on how methods from relation algebra (RA) can be applied. RA is a perfect companion for FCA because, while FCA facilitates visualisations of binary relations, RA defines operations on binary relations. Thus RA seems a natural candidate for a context-relation algebra as defined in Sect. 3. Section 2 compares RA to other algebras of relations and explains why RA is more suitable for FCA applications than the other methods. Section 4 focuses on the applications of RA/FCA in the area of lexical databases using examples from Roget's Thesaurus.

* This is a preprint of a paper published in Schärfe; Hitzler; Ohrstrom (eds.), ICCS 2006, Springer Verlag, LNAI 4068, 2006, p. 388-400.

2 Algebras as a representation of logical structures

Logicians usually employ set-theoretic models for the formal semantics of formal languages. But it is also possible to use algebraic structures instead of set-theoretic models because a formal language can be first interpreted as an algebra, which then further can be interpreted as a set-theoretic structure. The advantage of this approach is that additional structures are represented in the algebra. Historically, the development of formal logic has been closely tied to the development of algebras of relations (cf. Maddux (1991)) as is evident in Peirce's work in both fields. The modern field of Algebraic Logic studies the relationship between different types of logics and different types of algebras (cf. Andreka et al., 1997). Surprisingly many structures from logic can be translated into algebraic structures and vice versa. The translations are beneficial because theorems and methods from either field become available in the other.

With respect to relational databases and other structures based on n-ary relations (such as power context families in FCA (Wille, 2002)), traditionally the most popular algebras use n-ary relations as basic units. Codd's (1970) modelling of relational databases with relational algebra (RLA) is well known. With respect to power context families, the algebra of choice is usually Peirce Algebraic Logic (PAL), such as in the modelling suggested by Eklund et al. (2000). Both RLA and PAL and a third algebra called Krasner algebra (cf. Hereth Correia & Pöschel, 2004) are very similar in nature to Henkin & Tarski's (1961) Cylindric Set Algebras (CSA). CSA, RLA, PAL and Krasner algebra all have the expressive power of first order logic (FOL) with equality (cf. Van den Bussche (2001) for CSA and RLA and Hereth Correia & Pöschel (2004) for PAL and Krasner). It is quite possible that they are equivalent or even isomorphic to each other in some sense (for CSA and RLA this is investigated by Imielinski & Lipski (1984); for PAL and Krasner, by Hereth Correia & Pöschel (2004)).

In addition to CSA, Tarski (1941) also studied an algebra which is quite different and goes back to Peirce, De Morgan and Schroeder (cf. van den Bussche (2001) for an overview). This algebra is called relation algebra (RA) – the similarity in name to Codd's relational algebra (RLA) is unfortunate, but these names are established in the literature. The difference between RA and CSA is that RA uses exclusively binary relations, whereas CSA and the other algebras use n-ary relations. At first sight, this appears to be a limitation of RA. But in fact, RA is quite powerful and has the expressive power of FOL with up to three variables. If one adds a form of projection operation to RA, one can obtain what is called a Fork algebra (FRA)[1] which has the expressive power of full FOL, but still only uses binary relations (Frias et al. 2004). In FRA, n-ary relations are encoded as a part-whole structure among binary relations. For example, a ternary relation consists of a binary relation, whose left or right element contains two parts which can be retrieved using the projection operation. While this may sound complicated, the operations of RA are overall much simpler (and easier to implement) than

[1] For the purposes of modelling FCA contexts with RA (Priss, 2006), only two elements that contain information about projections are required from FRA but none of the other operations. Thus in the remainder of this paper, the abbreviation RA is used to mean "RA including FRA elements if needed".

the operations of CSA; and RA has many interesting properties that can be calculated on an abstract level (cf. Pratt (1992) and (1993) for an overview).

We believe that RA is currently under-valued among computer scientists, although there has recently been an increased interest in relational methods in computer science as evidenced by the recent creation of a new journal in this field[2]. Part of the reason for RA's lack of popularity may be the fact that any university student who learns some mathematics is likely to learn some linear algebra. But not even every full time mathematics student learns about universal algebra, algebraic logic and lattice theory. With respect to physical space, linear algebra and vector spaces are useful models, but we would argue that, with respect to information spaces, perhaps other algebras than linear algebra can be more useful[3]. This claim is supported by the results of algebraic logic, which demonstrate the close connection between logic and algebra. An elaboration of this claim with respect to the use of non-linear algebra-based methods in information retrieval can be found in Priss (2000).

Apart from the availability of simpler operations in RA, another advantage of RA is the availability of visualisations. There are several types of visualisations for different algebras: Venn Diagrams for set-based Boolean algebras, n-dimensional coordinate systems for vector spaces and CSA-style operations (cf. Andreka et al. 1997), Peirce's Existential Graphs for PAL-style operations. But all of these have the limitation that they become difficult or impossible to draw as the complexity increases. They tend to be suitable for instances of relations (the relationship between a few points in space or between a few Existential Graphs), but not for an overview of a larger system of relations. On the other hand, because RA uses solely binary relations, the visualisation methods of FCA are instantly available (i.e. concept lattices). Although FCA visualisations also have a limit with respect to how much complexity can be represented, that limit is much higher than for n-dimensional vector spaces. For example, it is already difficult to represent 3-dimensional vector spaces on paper, but a concept lattice can easily represent a (Boolean) lattice with up to 5 independent co-atoms (corresponding to 5 dimensions), and many more if they are not completely independent. Methods of zooming and nesting are available for larger systems using the software TOSCANA (Eklund et al., 2000). In that manner, FCA visualisations enable users to obtain an overview of larger sets of data and to explore hidden structures among the data.

Of course, FCA visualisations have always been applied to power context families and many-valued contexts. The strategy that is normally used is to first apply PAL operations to formal contexts and then to extract a binary relation from the n-ary relations, by selecting two columns with or without scaling and applying combination operations to columns before selecting them. But that implies that it is also conceivable to reverse these two steps and to construct a binary encoding of the n-ary relations right away and then to use RA to operate on these formal contexts.

---

[2] `www.jormics.org`

[3] An anonymous reviewer of this paper remarked that a similar dichotomy can be found in other areas of mathematics: "in topology, Hausdorff spaces are convincing models of real geometries, but T0 spaces are more interesting from a logical viewpoint. Likewise, classical metric spaces are nice generalizations of Euclidian distances, while ultrametrics are often more appropriate to measure the 'distance' between pieces of information."

3 Context-RAs and Context Algebraic Structures

This section provides a brief introduction into how relation algebra can be applied to FCA. A more detailed explanation of this topic can be found in Priss (2006), from which the definitions below are taken. The cross table of a formal context can be considered a Boolean (or binary) matrix in the sense of Kim (1982), for which matrix operations are defined as follows: with $(i,j)_I$ denoting the element in row i, column j in matrix I and $\vee$, $\wedge$ and $\neg$ denoting Boolean OR, AND and NOT: $(i,j)_{I\cup J} := (i,j)_I \vee (i,j)_J$; $(i,j)_{\overline{I}} := \neg(i,j)_I$; $(i,j)_{I\circ J} := 1$ iff $\exists_k : (i,k)_I \wedge (k,j)_J$; $(i,j)_{I^d} := (j,i)_I$. The operations $\cap$ and $\subseteq$ are as usual: $I \cap J := \overline{\overline{I} \cup \overline{J}}$ and $I \subseteq J :\Longleftrightarrow I \cap J = I$. A matrix containing all 0's is denoted by nul; a matrix with all 1's is one, a matrix with 1's on the diagonal and 0's otherwise is dia. A matrix is symmetric if $I = I^d$, reflexive if $dia \subseteq I$, transitive if $I \circ I \subseteq I$.

To distinguish between operations on sets and on matrices, we use typewriter font (e.g. A, B) for sets and italics (e.g. A, B) for matrices. A formal context normally consists of two sets and a matrix: $(\mathtt{G}, \mathtt{M}, I)$. In this paper, all formal contexts of an application are assumed to be defined with respect to a finite, linearly ordered set ACT called an *active domain*. That means that for all sets of objects and attributes: $\mathtt{G}, \mathtt{M} \subseteq \mathtt{ACT}$. A set $\mathcal{A}$ is defined as the set of all Boolean $|\mathtt{ACT}| \times |\mathtt{ACT}|$ matrices together with an interpretation that ensures that, semantically, for $I \in \mathcal{A}$ and $1 \leq n \leq |\mathtt{ACT}|$, the nth row and column in I corresponds to the nth element in ACT. It is then said that I *is based on* $\mathcal{A}$ denoted by $I_\mathcal{A}$ (although the subscript can be omitted if it is clear which active domain is meant).

Definition 1. *A matrix-RA based on $\mathcal{A}$ is an algebra $(R, \cup, ^-, one, \circ, ^d, dia())$ where $one \in R$ is a reflexive, symmetric and transitive matrix; $R := \{I_\mathcal{A} | I_\mathcal{A} \subseteq one\}$ is a set of Boolean matrices; $\cup, ^-, \circ, ^d$ are the usual Boolean matrix operations; and for any set $\mathtt{S} \subseteq \mathtt{ACT}$ and $\mathtt{a(n)}$ denoting the nth element in ACT, $dia(\mathtt{S})$ is defined by $(i,j)_{dia(\mathtt{S})} = 1$ iff $i = j$ and $\mathtt{a(i)} \in \mathtt{S}$ (but only if $dia(\mathtt{S}) \subseteq one$).*

It can be shown that a matrix-RA is an RA and fulfills all the axioms of an RA, such as $(R, \cup, \cap, ^-, nul, one)$ is a Boolean algebra; $\circ$ is associative and distributive with $\cup$; dia is a neutral element for $\circ$ (but unique inverse elements need not exist); $(I^d)^d = I$; $(I \cup J)^d = I^d \cup J^d$; $(I \circ J)^d = J^d \circ I^d$; and so on (see Priss (2006)).

With $\mathtt{G}, \mathtt{M} \subseteq \mathtt{ACT}$, a formal context can be represented "based on $\mathcal{A}$": the sets can be represented as $dia(\mathtt{G})$ and $dia(\mathtt{M})$ or combined as $sqr(\mathtt{G}, \mathtt{M}) := dia(\mathtt{G}) \circ one \circ dia(\mathtt{M})$. A formal context can then be represented using three matrices: $(dia(\mathtt{G}), dia(\mathtt{M}), I_\mathcal{A})$ or using two matrices $(sqr(\mathtt{G}, \mathtt{M}), I_\mathcal{A})$, in both cases with $I_\mathcal{A} \subseteq sqr(\mathtt{G}, \mathtt{M})$. A formal context without empty rows or columns can be represented by a single matrix: $I_\mathcal{A}$. A natural RA can be defined for any set of formal contexts:

Definition 2. *A context-RA based on $\mathcal{A}$ for a set of formal contexts is the smallest matrix-RA based on $\mathcal{A}$ that contains these contexts.*

Instead of representing every formal context using $|\mathtt{ACT}| \times |\mathtt{ACT}|$ matrices, it is also possible to represent formal contexts in the usual way, but in that case, the RA operations need to be modified (see the next definition). The sets $\mathtt{G}, \mathtt{M} \subseteq \mathtt{ACT}$ are assumed

to be linearly ordered according to $\mathtt{ACT}$. This linear order ensures that operations can be defined among formal contexts which have different sets of objects and attributes. In this representation, a formal context $(\mathtt{G}, \mathtt{M}, I)$ *based on* ACT contains a Boolean matrix I of size $|\mathtt{G}| \times |\mathtt{M}|$ where the ith row corresponds to the ith element in $\mathtt{G}$ and the jth column corresponds to the jth element in $\mathtt{M}$. This can be denoted as $I_{\mathtt{G},\mathtt{M}}$. But in the remainder of this paper the subscript of I is omitted, if the formal context is denoted using sets of objects and attributes (not matrices). In these cases, it is always assumed that I's dimensions correspond to its sets of objects and attributes.

Definition 3. *For formal contexts* $\mathcal{K}_1 := (\mathtt{G}_1, \mathtt{M}_1, I)$ *and* $\mathcal{K}_2 := (\mathtt{G}_2, \mathtt{M}_2, J)$ *based on* ACT *the following context operations are defined:*
$\mathcal{K}_1 \sqcup \mathcal{K}_2 := (\mathtt{G}_1 \cup \mathtt{G}_2, \mathtt{M}_1 \cup \mathtt{M}_2, I \sqcup J)$ *with* $\mathtt{g}I \sqcup J\mathtt{m} :\Longleftrightarrow \mathtt{g}I\mathtt{m}$ *or* $\mathtt{g}J\mathtt{m}$
$\mathcal{K}_1 \sqcap \mathcal{K}_2 := (\mathtt{G}_1 \cup \mathtt{G}_2, \mathtt{M}_1 \cup \mathtt{M}_2, I \sqcap J)$ *with* $\mathtt{g}I \sqcap J\mathtt{m} :\Longleftrightarrow \mathtt{g}I\mathtt{m}$ *and* $\mathtt{g}J\mathtt{m}$
$\mathcal{K}_1 \diamond \mathcal{K}_2 := (\mathtt{G}_1, \mathtt{M}_2, I \diamond J)$ *with* $\mathtt{g}I \diamond J\mathtt{m} :\Longleftrightarrow \exists_{\mathtt{n} \in (\mathtt{M}_1 \cap \mathtt{G}_2)} : \mathtt{g}I\mathtt{n}$ *and* $\mathtt{n}J\mathtt{m}$
$\overline{\mathcal{K}_1} := (\mathtt{G}_1, \mathtt{M}_1, \overline{I})$; $\quad \mathcal{K}_1^d := (\mathtt{M}_1, \mathtt{G}_1, I^d)$.

Definition 4. *A context algebraic structure (CAS) based on* ACT *is a three sorted algebra* $(R_1, R_2, R_3, \sqcup, ^-, \diamond, ^d, dia(), \mathtt{set}(), (\cdot,\cdot,\cdot))$ *where* R_2 *is a set of subsets of* $\mathtt{ACT}$*,* R_3 *is a set of Boolean matrices,* R_1 *is a set of formal contexts based on* $\mathtt{ACT}$ *and constructed using the partial function* $(\cdot,\cdot,\cdot) : R_2^2 \times R_3 \to R_1$*;* $\sqcup, ^-, \diamond, ^d$ *are according to Definition 3;* $\mathtt{set}_\mathtt{G}(I) := \{\mathtt{g} \in \mathtt{G} \mid \exists_{\mathtt{m} \in \mathtt{M}} : \mathtt{g}I\mathtt{m}\}$*;* $\mathtt{set}_\mathtt{M}(I) := \{\mathtt{m} \in \mathtt{M} \mid \exists_{\mathtt{g} \in \mathtt{G}} : \mathtt{g}I\mathtt{m}\}$*; and* $dia_\mathtt{G}(\mathtt{S}_\rightarrow)$ *is defined by* $(i,j)_{dia_\mathtt{G}(\mathtt{S}_\rightarrow)} = 1$ *iff* $i = j$ *and for the ith element in* $\mathtt{G}$*:* $\mathtt{g}(\mathtt{i}) \in \mathtt{S}$*;* $dia_\mathtt{M}(\mathtt{S}_\uparrow)$ *is defined analogously.*

If $\mathtt{G}_1 = \mathtt{G}_2$, $\mathtt{M}_1 = \mathtt{M}_2$, then $\cup$ and $\cap$ can be used instead of $\sqcup$ and $\sqcap$. If $\mathtt{M}_1 = \mathtt{G}_2$, then $\circ$ can be used instead of $\diamond$. Using these definitions, the normal FCA operations can be defined, such as the prime operator (cf. Priss (2006)), using either context-RAs or CAS. For many-valued contexts and power context families, a fork algebraic extension is required (Priss, 2006), but that is not relevant for this paper.

For implementation purposes, it should be noted that it is not suggested that by using matrix-RAs for modelling contexts that these actually have to be represented as giant matrices in a computer. In FCA applications, giant formal contexts are usually sparse matrices or contain many duplicate rows or columns. When dealing with sparse matrices, non-RA-based algorithms might be faster, but RA can be used as an underlying theory for proving theorems. This is shown using examples from lexical databases in the next sections. On the other hand, with respect to small contexts (i.e. less than 100 objects and attributes), RA operations can be directly implemented. RA software already exists[4] and could possibly be combined with existing FCA software. RA operations could then be used as an additional query interface.

4 Lexical Databases

A lexical database is defined here as an organised collection of words in electronic form. That includes dictionaries such as Webster's or the Oxford English Dictionary,

[4] RelView: `www.informatik.uni-kiel.de/~progsys/relview.shtml`

with formal definitions and glosses; semantic lexicons such as Roget's Thesaurus (Roget, 1962) and WordNet (Miller et al. (1990)), organised by conceptual categories and lexical relations; and bilingual dictionaries, where the primary structure is a conceptual mapping between languages. Previously, Priss & Old (2004) developed methods for suitable representations for lexical databases – a set of guidelines for visualising lexical relations using FCA.

Roget's Thesaurus (ROGET), in particular in the form of a lexical database[5], is the source of examples used in this paper. ROGET consists of a conceptual hierarchy, at the bottom of which are words grouped by shared meaning. These groups are referred to as senses, and members of a particular group are commonly referred to as synonyms. The relationship between words and senses can be represented as a formal context where instances (actual entries of particular words in particular senses) are represented by a cross in the context (e.g. Fig. 1). In this paper, the term "entry" is reserved for this relationship between words and senses. A word has usually more than one sense (several entries in ROGET); and each sense is usually represented by (and contains) more than one synonym. The word "over", for example, has 22 senses in ROGET, represented by 22 entries in the thesaurus; and the number of entries in those 22 senses (synonyms sharing a sense with over) ranges from 3 to 37. The question which arises is: which words and senses should be included in a formal context for a given word? A possible answer to this question is provided by neighbourhood contexts and lattices as discussed in the next section.

WORDS ⟶ SYNONYMY

SENSES ↓

	over	above	beyond	across	past	in excess of	...
9:12:1	x						
206:24:2	x	x					
206:27:4	x	x	x			x	
227:40:1	x	x					
...	x	x	x	x	x		

POLYSEMY

Fig. 1. Words, senses and entries in Roget's Thesaurus

4.1 A formalisation of neighbourhood lattices using RA

A mathematical definition of "neighbourhood lattices" of Roget's Thesaurus was originally defined in an unpublished manuscript by Rudolf Wille and can be found in Priss

[5] The relational database used in this paper is based on Roget (1962), which was converted to electronic format by Dr. W. A. Sedelow and Dr. S. Yeates Sedelow and edited and enhanced by L. J. Old.

(1998). But, although neighbourhood lattices have been produced for numerous examples over the years, so far the underlying theory has not been further investigated. This section shows how RA can be used to analyse and model the structures of neighbourhood lattices. A neighbourhood lattice starts with a word $\mathtt{w}$, collects all the senses that $\mathtt{w}$ has, then collects all other words that also have these senses and so on. Or, alternatively, the process can be started with a sense, collecting all its words and so on. The operation of collecting all objects that an attribute has (or vice versa) is called a "plus operator" (Priss & Old, 2004). Under different names, the idea of the plus operator can be found in many linguistic applications, such as starting with a word in one language, then retrieving all the translations in another language, and so on (e.g. Wunderlich (1980) or Dyvik (1998)). Using RA, the plus operator is defined as follows:

Definition 5. *For a context* $(\mathtt{G}, \mathtt{M}, I)$*,* $\mathtt{H} \subseteq \mathtt{G}$*,* $\mathtt{N} \subseteq \mathtt{M}$*, a column matrix* H *that has a 1 in the position of each element of* $\mathtt{H}$ *(i.e.,* $H := dia_{\mathtt{G}}(\mathtt{H}_{\rightarrow}) \circ one_{\mathtt{G},\{\}}$*), and* N *an analogous row matrix, a plus operator is defined as* $H^+ := H^d \circ I$ *and* $N^+ := I \circ N^d$*.*

It follows that H^+ is a row matrix and N^+ is a column matrix. Applying the plus operator twice yields $H^{++} = I \circ (H^d \circ I)^d = I \circ I^d \circ H$ and $N^{++} = N \circ I^d \circ I$; three times: $H^{+++} = H^d \circ I \circ I^d \circ I$ and $N^{+++} = I \circ I^d \circ I \circ N^d$, and so on. A neighbourhood context can utilise the plus operator any number of times and can be started with sets of objects or attributes. A typical neighbourhood context is $(\mathtt{set}_{\mathtt{G}}(H^{++}), \mathtt{set}_{\mathtt{M}}(H^{+++}), I)$. The RA representation of the plus operator implies that the operator is essentially formed by repeated composition of I and I^d. For finite matrices repeated composition leads to a "transitive closure", normally defined as $I^{trs} := I \cup (I \circ I) \cup (I \circ I \circ I) \cup ...$. It should be noted that the calculation of a transitive closure is not an FOL operation, but an additional operation that cannot axiomatically be derived from the other RA operations.

If a matrix I is reflexive in all rows that are not empty, then $I^{trs} = I \circ I \circ I \circ ...$ because $\forall_{\mathtt{x},\mathtt{y}} : (\mathtt{x}I\mathtt{y} \Rightarrow \mathtt{x}I\mathtt{x}, \mathtt{x}I\mathtt{y}) \Rightarrow \forall_{\mathtt{x},\mathtt{y}} \exists_{\mathtt{z}} : (\mathtt{x}I\mathtt{y} \Rightarrow \mathtt{x}I\mathtt{z}, \mathtt{z}I\mathtt{y}) \Longleftrightarrow I \subseteq I \circ I$. The matrix $(I \circ I^d)$ which is used in neighbourhood contexts is reflexive for non-empty rows, i.e. $\mathtt{x}I\mathtt{y} \Rightarrow \mathtt{x}(I \circ I^d)\mathtt{x}$, because $\mathtt{x}(I \circ I^d)\mathtt{y} \Longleftrightarrow \exists_{\mathtt{z}} : \mathtt{x}I\mathtt{z}, \mathtt{y}I\mathtt{z}$. Thus $(I \circ I^d)^{trs} = (I \circ I^d) \circ (I \circ I^d) \circ ...$. The matrix is also symmetric because $I \circ I^d = (I \circ I^d)^d$ according to the rules for ^{d}. This implies the following lemma and leads to the definition of a context which uses this matrix. Figure 2 provides an illustration.

Lemma 1. *If* I *is a matrix of a formal context without empty rows and without empty columns, then* $(I \circ I^d)^{trs}$ *is an equivalence relation on objects and* $(I^d \circ I)^{trs}$ *is an equivalence relation on attributes.*

Definition 6. *With* $I^{dec} := I \circ (I^d \circ I)^{trs}$*, the neighbourhood closure context of a set* $\mathtt{H}$ *of objects is defined as* $(\mathtt{set}_{\mathtt{G}}((I \circ I^d)^{trs} \circ H), \mathtt{set}_{\mathtt{M}}(H^d \circ I^{dec}), I)$ *and of a set* $\mathtt{N}$ *of attributes as* $(\mathtt{set}_{\mathtt{G}}(I^{dec} \circ N^d), \mathtt{set}_{\mathtt{M}}(N \circ (I^d \circ I)^{trs}), I)$*. Its corresponding lattice is called the neighbourhood closure lattice.*

I^{dec} (where "dec" stands for "decomposition") has some interesting properties. For example, it does not matter whether one starts the calculation with objects or attributes. The properties are summarised in the next lemma. A horizontal decomposition of a lattice is a decomposition into components whose horizontal sum (Ganter & Wille,

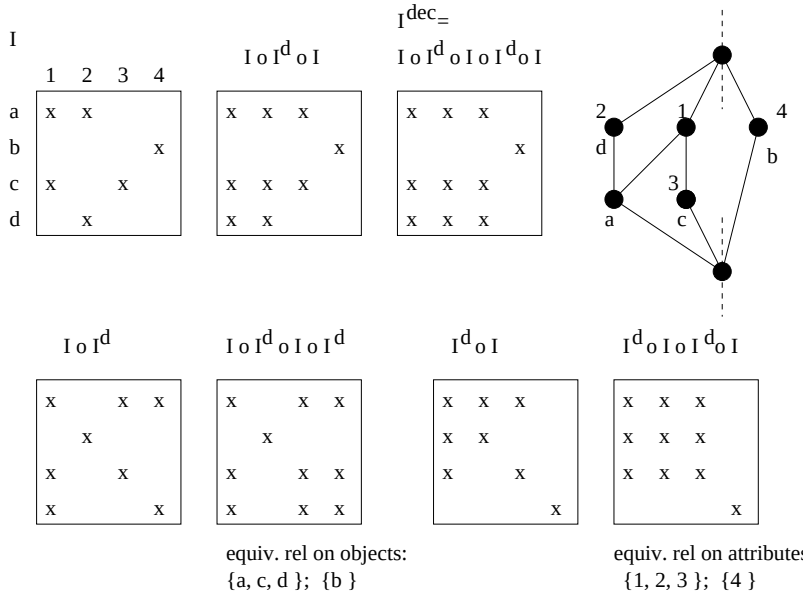

Fig. 2. An example illustrating Lemma 1 and Lemma 2

1999) is the original lattice. If one removes the top and bottom nodes from a lattice, then the remaining connected graphs are the components of the horizontal decomposition.

Lemma 2. *For finite matrices I:*

1. $I^{dec} = (I \circ I^d)^{trs} \circ I = I \circ (I^d \circ I)^{trs}$.
2. $I \subseteq I^{dec}$.
3. I^{dec} *implies a horizontal decomposition of the concept lattice of* I.

Proof: 1) If, for example, $(I^d \circ I)^{trs} = (I^d \circ I)$ and $(I \circ I^d)^{trs} = (I \circ I^d) \circ (I \circ I^d)$ but $(I \circ I^d) \circ (I \circ I^d) \circ I \neq I \circ I^d \circ I$ then this would be a contradiction to $I^d \circ I \circ I^d \circ I = I^d \circ I$ because of $(I^d \circ I)^{trs}$.
2) $\mathtt{x} I \mathtt{y} \Rightarrow \exists_{\mathtt{z}_1,\ldots,\mathtt{z}_n} : \mathtt{x} I \mathtt{z}_1, \mathtt{z}_2 I \mathtt{z}_1, \mathtt{z}_2 I \mathtt{z}_3, \ldots, \mathtt{z}_n I \mathtt{y} \Longleftrightarrow \mathtt{x}(I \circ I^d \circ \ldots \circ I^d \circ I)\mathtt{y}$ with $\mathtt{x} = \mathtt{z}_\mathtt{i}$ for even i and $\mathtt{y} = \mathtt{z}_\mathtt{i}$ for odd i.
3) Because of Lemma 1 and Lemma 2.2.

Thus the neighbourhood closure lattice of an object or an attribute is the component of the horizontal decomposition of the original lattice that contains the object or attribute. This is, of course, what was to be expected given the definition of the plus operator – but the RA modelling provides an interesting representation of how the neighbourhood closure context is computed.

For small formal contexts, it is trivial to calculate neighbourhood closure contexts because one simply needs to decompose the lattice. But for large formal contexts, it is not efficient to first calculate the whole lattice if one is only interested in the neighbourhood of some objects or attributes. In that case, it is much more feasible to calculate

only neighbourhood closure lattices because these contain the complete information about an object or attribute. This is especially the case if a lattice contains a few large components and many small components and one is interested in objects or attributes that belong to the small components (see the next section). In some cases, even neighbourhood closure lattices may be too big, in which case simple neighbourhood lattices may be a sufficient approximation. Future research should be conducted to develop heuristics to determine which neighbourhood lattice is best for which type and size of context. It might also be of interest to obtain heuristics for estimating how many iterations are needed to reach the transitive closure.

4.2 Empirical results for a neighbourhood closure context of ROGET

Using the RA formalisation as a guideline, we have calculated the neighbourhood closure context for ROGET. Considering that the full ROGET context consisting of all words and senses has about 113,000 objects, 71,000 attributes and 200,000 crosses, calculating a transitive closure is not trivial. Calculating the neighbourhood closure context for the full ROGET database results in 26,314 equivalence classes, or components, ranging from one largest component of 138,919 entries (belonging to 38,621 senses), to 22,206 single-entry components. This means that the majority of entries in ROGET (about 70%) are connected, either by words with shared senses (synonymy) or by senses having shared words (polysemy). The majority of the single-entry components derive from ROGET lists, a classification type where words representing such objects as ship parts, species of animal, or capital cities, each occupy a single sense and have no synonyms. The components with 10 to 28 senses typically contain words that are fairly specialised, but still somewhat polysemous, such as belief systems (“freethinker”, “nihilist”), occupations (“moneylender”), musical forms (“serenade”), temporal adjectives (“dayly”) and countries and capital cities, which happen to occur in more than one list.

The distribution of component sizes is shown in Fig. 3. To display the number of components (y-axis), a log2 function was used. A distribution consisting of one large component, a large number of tiny components and a smaller number of exponentially distributed components in between has been called a “power law distribution” by Strogatz (2001). Power law distributions are a phenomenon found among such diverse areas as word or letter frequency in text, strength of earthquakes, connectivity in the brain and HIV epidemics (Barabasi, 2002), the distribution of forest fires, species group size in biology, and web sites on the Internet. There has been some debate as to the actual significance of power distributions but Barabasi (p. 77) makes the claim that: “Nature normally hates power laws. In ordinary systems all quantities follow bell curves, and correlations decay rapidly, obeying exponential laws. But all that changes if the system is forced to undergo a phase transition. Then power laws emerge – nature’s unmistakable sign that chaos is departing in favor of order.” Thus it is of potential significance that components in ROGET follow this particular distribution.

4.3 Antonymy

A second example is the treatment of antonymy. Antonyms are generally known as words with opposite meanings. Antonymy is interesting for two reasons. First, antonymy

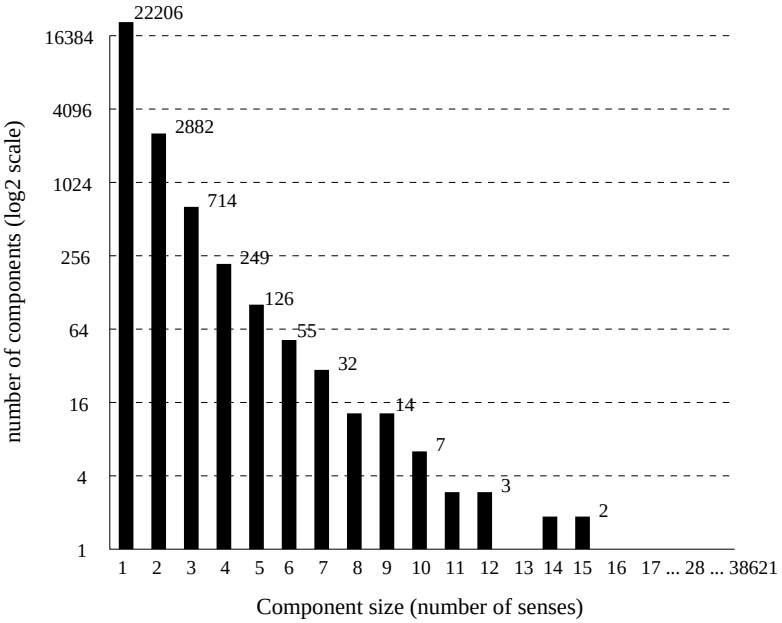

Fig. 3. Exponential distribution of components

is what is called a “lexical relation” in WordNet (Miller et al., 1990). That means that it is a relationship between senses of words and not between words. For example, “big” and “small” can be considered antonyms. But synonyms of “big” (such as “huge”) need not be antonyms of “small” and its synonyms. To model this using RA and the ROGET context from the previous section, it would be necessary to use a Fork relational (FRA) matrix and projections. This is because a binary relation among antonyms would correspond to a relation among word/sense pairs which would need to be encoded using FRA. For the purposes of this paper, we are treating antonymy as a relation among words (similar to synonymy), thus avoiding the need for FRA.

Second, even though antonyms express some form of contrast, negation or duality, they are also usually quite close in meaning. For example, “hot” and “cold” both describe temperature; “up” and “down” both describe direction. Word association studies (where a person is told a word and has to name the first word that comes to mind) show that antonyms are as closely associated in the mental lexicon as are synonyms (Miller et al., 1990).

ROGET does not identify antonyms on a sense-level. The first edition of the book version of Roget’s Thesaurus identified antonymous categories by arranging them opposite to each other in the table of contents. A similar structure on the category-level has been included in ROGET. Furthermore, we have included in ROGET some data that is available from word association tests (Nelson et al., 1998) in the form of an antonymy relation among words.

The goal of this section is to detect and analyse this second feature of antonymy (contrast versus shared synonyms) in ROGET. For example (cf. Fig. 4), on a category level, “293 progression” (containing “progress”) and “294 regression” (containing “retreat”) are antonyms in ROGET. They share the higher-level class of “Motion With Reference To Direction”. They also share a synonym “go” in category “289 direction”. Table 1 shows a (manually compiled) list of antonyms from word association data, which share a synonym in ROGET. It should be noted that in this case, synonymy is not only evaluated on the sense-level but also on the paragraph level. That means that all words in a paragraph of ROGET are considered synonyms for this purpose.

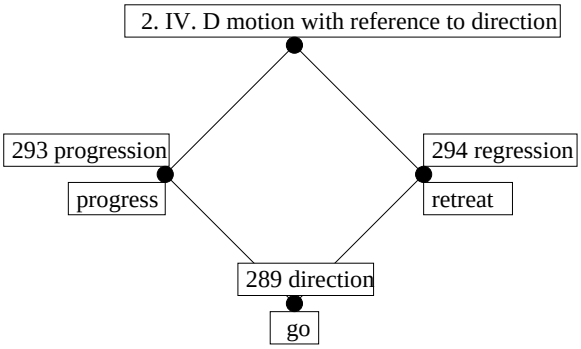

Fig. 4. Antonymous categories share meaning

Table 1. Antonyms that share synonyms

antonym 1	shared synonym	antonym 2
give	yield	accept
add	compute	subtract
descend	slope	ascend
editor	reviewer	author
after	then	before
white	bleak	black
sweet	brisk	bitter
suck	snuffle	blow
sharp	abrupt	blunt
effect	sequence	result
major	elective	minor
future	sometime	past

One approach to studying shared meanings among antonyms is to identify those neighbourhood contexts of antonyms where the sets of attributes are not disjoint. In this

case, again it is useful to consider paragraphs as attributes instead of senses because otherwise there would not be much overlap. But because the sets of synonyms at that level are large, the plus operator is used only once. Thus, if H denotes the column matrix which contains exactly one 1 in the position of a word $\mathtt{w}$ and J denotes the column matrix which contains one 1 in the position of the antonym of $\mathtt{w}$ and if $H^{++} \cap J^{++} \neq \emptyset$ then the neighbourhood context $(\mathtt{set_G}(H \cup J \cup (H^{++} \cap J^{++})), \mathtt{set_M}(H^{+} \cup J^{+} \cup (H^{++} \cap J^{++})^{+}), I)$ is formed. Figure 5 shows the example of "hot" and "cold", which share "keen" as a synonym if synonymy is evaluated at the paragraph level. In this case the shared synonym refers to metaphoric senses of the original words. It would be of interest to conduct a more detailed investigation of all such cases in ROGET in future research.

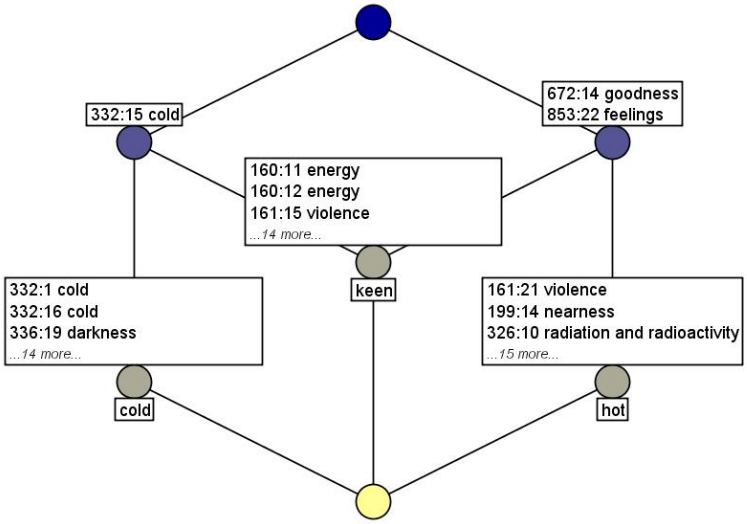

Fig. 5. Overlapping neighbourhoods for antonyms

5 Conclusion

This paper demonstrates the applicability of RA formalisations for modelling lexical databases with FCA by using "neighbourhood contexts and lattices". Specific examples are neighbourhood closure contexts of the complete ROGET context and neighbourhood lattices for antonymous words. The examples show that different types of neighbourhood contexts are relevant for different aspects of lexical databases. But the research presented in this paper is not restricted to linguistic applications. Similar structures could be investigated in concept lattices in other application areas.

References

1. Andreka, H.; Nemeti, I.; Sain, I. (1997). *Algebraic Logic.* In: Gabbay (ed.), Handbook of Philosophical Logic, Kluwer.
2. Barabasi, A. L. (2002). *Linked: The New Science of Networks.* Perseus Publishing.
3. Codd, E. (1970). *A relational model for large shared data banks.* Communications of the ACM, 13:6.
4. Dyvik, H. (1998). *A Translational Basis for Semantics.* In: Johansson and Oksefjell (eds.): Corpora and Crosslinguistic Research: Theory, Method and Case Studies, Rodopi, 51-86.
5. Eklund, P,; Groh, B.; Stumme, G.; Wille, R. (2000). *A contextual-logic extension of TOSCANA.* In: Ganter & Mineau (eds.), Conceptual Structures: Logical, Linguistics, and Computational Issues, Springer LNAI 1867, p. 453-667.
6. Frias, M.; Veloso, P.; Baum, G. (2004). *Fork Algebras: Past, Present and Future*. Journal on Relational Methods in Computer Science, 1, p. 181-216.
7. Ganter, B.; Wille, R. (1999). *Formal Concept Analysis.* Mathematical Foundations. Berlin-Heidelberg-New York: Springer, Berlin-Heidelberg.
8. Henkin, L.; Tarski, A. (1961). *Cylindric algebras.* In: Dilworth, R. P., Lattice Theory, Proceedings of Symposia in Pure Mathematics, AMS, p. 83-113.
9. Hereth Correia, J.; Pöschel, R. (2004). *The Power of Peircean Algebraic Logic (PAL).* In: Eklund (Ed.) Concept Lattices. LNCS 2961, Springer Verlag. p. 337-351.
10. Imielinski, T.; Lipski, W. (1984). *The relational model of data and cylindric algebras.* Journal of Computer and Systems Sciences, 28, p. 80-102.
11. Kim, K. H. (1982). *Boolean Matrix Theory and Applications.* Marcel Dekker Inc.
12. Maddux R. (1991). *The origin of relation algebras in the development and axiomatization of the calculus of relations.* Studia Logica, 5, 3-4, p. 421-456.
13. Nelson, D. L., McEvoy, C. L., & Schreiber, T. A. (1998). The University of South Florida word association, rhyme, and word fragment norms. Available at `http://www.usf.edu/FreeAssociation/`
14. Miller, G.; Beckwith, R.; Fellbaum, C.; Gross, D.; Miller, K. J. (1990). *Introduction to WordNet: an on-line lexical database.* International Journal of Lexicography, 3, 4, p. 235-244.
15. Pratt, V.R. (1992). *Origins of the Calculus of Binary Relations.* Proc. IEEE Symp. on Logic in Computer Science, p. 248-254.
16. Pratt, V.R. (1993). *The Second Calculus of Binary Relations.* Proc. 18th International Symposium on Mathematical Foundations of Computer Science, Gdansk, Poland, Springer-Verlag, p. 142-155.
17. Priss, U. (1998). *Relational Concept Analysis: Semantic Structures in Dictionaries and Lexical Databases.* (PhD Thesis) Verlag Shaker, Aachen 1998.
18. Priss, U. (2000). *Lattice-based Information Retrieval.* Knowledge Organization, Vol. 27, 3, 2000, p. 132-142.
19. Priss, U.; Old, L. J. (2004). *Modelling Lexical Databases with Formal Concept Analysis.* Journal of Universal Computer Science, Vol 10, 8, 2004, p. 967-984.
20. Priss, U. (2006). *An FCA interpretation of Relation Algebra.* ICFCA'06. Springer Verlag, LNCS (to appear).
21. Roget, Peter Mark (1962). *Roget's International Thesaurus.* 3rd Edition Thomas Crowell, New York.
22. Strogatz, H. (2001). Exploring complex networks. Nature, 410, 268-276.
23. Tarski, A. (1941). *On the calculus of relations.* Journal of Symbolic Logic, 6, p. 73-89.
24. Van den Bussche, J. (2001). *Applications of Alfred Tarski's Ideas in Database Theory.* Proceedings of the 15th International Workshop on Computer Science Logic. LNCS 2142, p. 20-37.

25. Wille, R. (2002). *Existential Concept Graphs of Power Context Families.* In: Priss; Corbett; Angelova (Eds.) Conceptual Structures: Integration and Interfaces. LNCS 2393, Springer Verlag, p. 382-395.
26. Wunderlich, D. (1980). *Arbeitsbuch Semantik*. Königstein/Ts: Athenaeum.