Concept Similarity and Related Categories in SearchSleuth

Frithjof Dau, Jon Ducrou and Peter Eklund

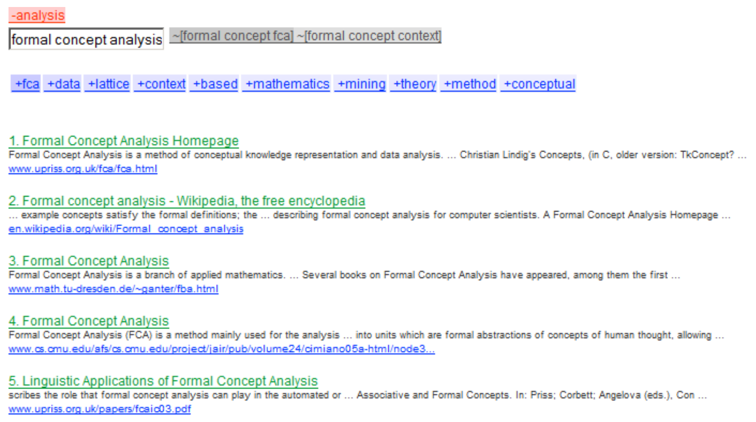
dau@dr-dau.net, jonducrou@gmail.com, peklund@uow.edu.au
School of Information Systems and Technology
University of Wollongong
Northfields Ave, Wollongong, NSW 2522, Australia.

Abstract. SearchSleuth is a program developed to experiment with the automated local analysis of Web search using formal concept analysis. SearchSleuth extends a standard search interface to include a conceptual neighborhood centered on a formal concept derived from the initial query. This neighborhood of the concept derived from the search terms is decorated with its upper and lower neighbors representing more general and specialized concepts respectively. In SearchSleuth, the notion of related categories – which are themselves formal concepts – is also introduced. This allows the retrieval focus to shift to a new formal concept called a sibling. This movement across the concept lattice needs to relate one formal concept to another in a principled way. This paper presents the issues concerning exploring and ordering the space of related categories.

1 Introduction

There are several Formal Concept Analysis-based Web search applications which provide automatic local analysis of search results for query refinement and labeled clustering [1–3]. These systems work via the creation of a conceptual space from polled search results which are displayed in various ways. The method is limited in that the systems fail to create a concept representing the query itself within the information space – meaning the space is representative of the results returned from the query terms, but not to the query terms themselves. SearchSleuth [4] overcomes this problem by creating a conceptual space as a neighborhood of the *search concept*: the formal concept derived from the search terms. The resulting neighborhood is comprised of generalisations (upper neighbors), specializations (lower neighbors) and related categories (called siblings). Fig. 1 shows the interface and these components.

By centering the conceptual space around the search concept, the resulting query refinement operations are more closely coupled to the search terms used in the creation of the space. SearchSleuth was first presented at the concept lattice applications conference in October 2007 [4], in that paper we discussed some of the preliminaries of search in the conceptual neighborhood of a query and go on to differentiate SearchSleuth work from other FCA-based web search tool such as *CREDO* [1] and FooCA[2, 3]. In this paper, we re-iterate some



-analysis

formal concept analysis ~[formal concept fca] ~[formal concept context]

+fca +data +lattice +context +based +mathematics +mining +theory +method +conceptual

1. Formal Concept Analysis Homepage
Formal Concept Analysis is a method of conceptual knowledge representation and data analysis. ... Christian Lindig's Concepts, (in C, older version: TkConcept? ...
www.upriss.org.uk/fca/fca.html

2. Formal concept analysis - Wikipedia, the free encyclopedia
... example concepts satisfy the formal definitions; the ... describing formal concept analysis for computer scientists. A Formal Concept Analysis Homepage ...
en.wikipedia.org/wiki/Formal_concept_analysis

3. Formal Concept Analysis
Formal Concept Analysis is a branch of applied mathematics. ... Several books on Formal Concept Analysis have appeared, among them the first ...
www.math.tu-dresden.de/~ganter/fba.html

4. Formal Concept Analysis
Formal Concept Analysis (FCA) is a method mainly used for the analysis ... into units which are formal abstractions of concepts of human thought, allowing ...
www.cs.cmu.edu/afs/cs.cmu.edu/project/jair/pub/volume24/cimiano05a-html/node3...

5. Linguistic Applications of Formal Concept Analysis
scribes the role that formal concept analysis can play in the automated or ... Associative and Formal Concepts. In: Priss; Corbett; Angelova (eds.), Con ...
www.upriss.org.uk/papers/fcaic03.pdf

Fig. 1. SearchSleuth display, including top results, after a search for '`formal concept analysis`'. Generalization/specialization formal concepts shown above/below the search box resp. The related categories or siblings are to the right of search box.

of the fundamentals of SearchSleuth, so that the paper is self-contained, however our contribution is in terms of explorations of the category space: namely how alternative formal concepts in the neighborhood of the current query concept are derived. Our presentation includes the analysis of lattice-theoretic and set-theoretic notions of proximity and we conclude that these two ideas are orthogonal but complementary. The outcome is of the analysis is reflected in the design of SearchSleuth.

2 Navigation and Conceptual Neighborhoods

Kim and Compton [5, 6] presented a document navigation paradigm using FCA and a neighborhood display. Their program, *KANavigator* uses annotated documents that can be browsed by keyword and displays the direct neighborhood (in particular the lower neighbors) as its interface. Kim and Compton's system emphasised the use of textual labels as representations of single formal concepts as opposed to a line diagram of the concept lattice.

ImageSleuth [7] used a similar interface design to allow exploration of image collections. By showing upper and lower neighbors of the current concept and allowing navigations to these concepts, users could refine or generalise their position in the information space. This is aided by the use of pre-defined conceptual scales that could be combined to define the attribute set of the lattice which forms the information space (see Fig. 2 (left)).

ImageSleuth uses most of its interface (shown in Fig. 2) to show thumbnails of images in the extent of the chosen concept. As a result the user never sees the line diagram of a concept lattice. Instead, the lattice structure around the current concept is represented through the list of upper and lower neighbors that allow the user to move to super- or sub-concepts. For every upper neighbor (C, D) of the current concept (A, B) the user is offered to remove the set $B \setminus D$ of attributes from the current intent. Dually, for every lower neighbor (E, F) the user may include the set $F \setminus B$ of attributes which takes her to this lower neighbor. By offering the sets $B \setminus D$ and $F \setminus B$ dependencies between these attributes are shown. Moving to the next concept not having a chosen attribute in its intent may imply the removal of a whole set of attributes. ImageSleuth was usability tested and results indicated that the approach aided navigation in image collections [8, 9].

SearchSleuth follows from ImageSleuth and employs the same conceptual neighborhood paradigm for display purposes. Unlike ImageSleuth, SearchSleuth's context is not static, so the space is rebuilt with each navigation step. This is because computing the entire domain, the Internet, as a conceptual neighborhood would be computationally prohibitive.

3 Design Approach of SearchSleuth: Context Building

For the Web, result sets from search engines usually take the form of the lists of URLs, each with the document title, a short summary of the document (or snippet) and various details such as date last accessed. Formal Concept Analysis-based Web search tools use the text-based components of the result set to create a formal context of the results. This context is then the basis for the conceptual space to be navigated. One problem with the transformation from Web search results to formal context is that ranking information on the result set is lost. All results are treated equally, this issue is usually addressed by re-introducing the

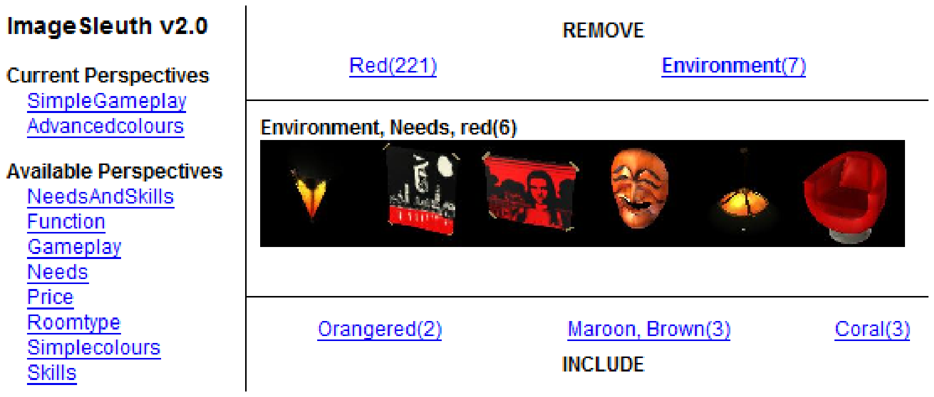


Fig. 2. ImageSleuth: the interface presents only the extent of the current concept as thumbnails and generalizations/specializations by removal/addition of attributes to reach the upper and lower neighbors (shown to the top/bottom of the thumbnails). Pre-defined scales (perspectives) are displayed on the left.

rank ordering from the search engine on any result set that is realized from the concept lattice.

Another difficulty experienced with Web search using FCA is that ranking methods use techniques such as link structure, page popularity and analysis of referring pages. As such, we cannot assume that all results of a multiple term query will contain all the queried terms used. Even a single query term may yield a page that does not contain the search term entered. This seems counter intuitive, but if there are enough Web pages linked to the result page that *do* contain the search term, that page's rank may be inflated enough to feature in the result set.

SearchSleuth uses the 'result has term' representation to build a formal context. The formal context for SearchSleuth is created on demand for each query; this suits the dynamic nature of the Internet. The formal objects are the individual results, and the formal attributes are the terms contained in the title and summary of each result. Terms are extracted from the title and summary after stemming and stop-word filtering has been performed. Stemming reduces words to their lexical root (e.g. `jump`, `jumping` and `jumps` are all reduced to `jump`). Stop-word filtering removes words without individual semantic value, for example `a`, `the` and `another`. Removing these words reduces the complexity of the context without noticeable reduction in semantic quality.

The context is then reduced by removing attributes with low support. Every attribute that has less than 5% of the objects in the incidence relation is removed. This decreases the computational overhead of involved in computing the concept lattice. Experience shows that this reduction rarely effects the computed conceptual neighborhood as the terms removed are scarce within the information

Once the formal context is constructed, the search concept is created. This is done by taking the provided query terms as attributes and deriving the formal concept. The upper neighbors of this formal concept are then derived and used to expand the context. This is done by querying the search engine with the attributes of each upper neighbor and inserting the results into the context. Results for these ancillary searches are limited to fewer results.

This process of building the context increases the number of terms in the information space based on a single level of generalisation. It makes the information space larger and richer.

4 Building the Information Space

Once the context is expanded, the search concept is recomputed as it may have been invalidated by this process. The upper and lower neighbors are computed next. A concept A is said to be the upper neighbor (or cover) of a iff we have $A > B$, and/but there is no concept C with $A > C > B$. A concept A is said to be the lower neighbor of (or covered by) a concept B iff we have $A < B$, and/but there is no concept C with $A < C < B$. The DownSet (DS) and UpSet (US) are defined as follows;

$$DS(X) := \{y \mid y \leqslant x \text{ for an } x \in X\} \qquad US(X) := \{y \mid y \geqslant x \text{ for an } x \in X\}$$

Upper and lower neighbors of a concept C are written as $UN(C)$ and $LN(C)$ respectively. Consider now the set of concepts X, $UN(X)$ is defined as the union of all upper neighbors of the concepts in X. Dually, consider the set of concepts X, $LN(X)$ is defined as the union of all lower neighbors of the concepts in X.

$$UN(X) := \bigcup\{UN(C) \mid C \in X\} \qquad LN(X) := \bigcup\{LN(C) \mid C \in X\}$$

The next step is to compute the related categories or sibling concepts. Sibling concepts are then calculated by finding all of the lower neighbors of upper neighbors which are upper neighbors of lower neighbors. Put another way, *siblings* constitute formal concepts created by the removal of an attribute (or attributes) that define an upper neighbor (UN), and the inclusion of an attribute (or attributes) that defines a lower neighbor (LN). Child Siblings (CS) and Parent siblings (PS) defined as: are defined:

$$CS(C) := UN(LN(C))\backslash\{C\} \qquad PS(C) := LN(UN(C))\backslash\{C\}$$

Exact Siblings (ES or Type I siblings) are those which are both Parent and Child siblings, Since they represent a stricter version of the notion of siblings they are referred to as Type I siblings and PS and CS are termed Type II siblings:

$$ES(C) := [LN(UN(C)) \cap UN(LN(C))]\backslash\{C\}$$

General Siblings (GS – Type III) define an even broader set of sibling concepts and are defined:

$$GS(C) := [DS(UN(C)) \cap US(LN(C))]\backslash(\{C\} \cup UN(C) \cup LN(C))$$

namely, anything strictly between some lower and some upper neighbor.

Child Siblings (CS), Parent Siblings (PS), and Exact Siblings (ES) form anti-chains, but General Siblings (GS) do not.

An example is shown in Fig. 3; concepts with a grey backing are Exact Siblings (ES) of the concept marked C.

Using the same labeling scheme as ImageSleuth for upper and lower neighbors and using the full intent as labels of sibling concepts, a display is rendered for the user (shown in Fig. 1).

Upper neighbors are shown above this text entry box, displayed as text labels (shown in Fig. 1). The labels are the attributes which would be removed to navigate to that upper neighbor. These labels are preceeded by a minus symbol (-) to reinforce the notion of *removal*.

Lower neighbors are similarly displayed (also indicated with arrows in Fig. 1), but placed below the text entry box. These labels are the attributes which would be added to navigate to that lower neighbor. Like upper neighbor labels, these labels are preceeded by a symbol to reinforce the labels meaning, namely the plus symbol (+) and the notion of *include*.

The display order of the upper and lower neighbors is defined by extent size, larger extents displayed first (left-most). Extent is representative of the

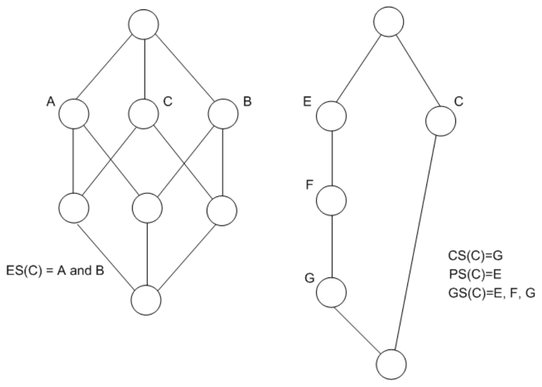


Fig. 3. Diagram demonstrating the *Parent Sibling (PS), Child Siblings (CS), Exact Sibling (ES)* and *General Siblings (GS)* concepts of the concept labeled with a C in two lattices.

importance or prominence within the current information space. Extent is also used to aid in the coloring of the labels background. The higher the extent on a lower neighbor, the deeper the blue block shade behind that concepts label. Upper neighbors are displayed with the same principle but with red block shade.

One method for dealing with the return of empty-extents from term-based searching is to provide users with a list of the terms entered so that they can incrementally remove terms to unconstrain the search. SearchSleuth explores an approach based on variations on defined distance [10] and similarity [11] metrics in the FCA literature in order to find similar relevant concepts.

Exact Siblings (ES) are shown to the right of the text entry box (indicated with arrows in Fig. 1) and are indicative of related concepts. The complete intent of these concepts is displayed within square brackets preceded by a tilde (`~[...]`). This helps group the concept intents and aids distinguishing between related concepts. Unlike upper and lower neighbors, Exact Siblings are ordered by *similarity*. The similarity metric is based on work by Lengnink [10] and was initially adapted for ImageSleuth. It uses the size of the common objects and attributes of the concepts. For two concepts (A, B) and (C, D), we set:

$$s((A,B),(C,D)) := \frac{1}{2}\left(\frac{|A \cap C|}{|A \cup C|} + \frac{|B \cap D|}{|B \cup D|}\right). \tag{1}$$

The similarity metric is used to order the exact sibling concepts, while highlighting remains based on extent size. Coloring on sibling labels is based on grey block shades.

By clicking any of the possible concept labels, the query is set to the intent of the selected concept and the query process is restarted. This is an important restructuring step as a change in the query will change the result set, and in order for the information to be valid it needs to be recomputed.

Looking back to Fig. 1, we see the search concept shown is based on the query `formal concept analysis`. It shows a single upper neighbor `analysis` which interestingly shows an implication that `formal` and `concept` are implied by `analysis`. The first of the lower neighbors is the acronym `fca`. This is followed by terms such as `lattice`, `mathematics` and `theory`. These terms are good examples of specialisation from the concept of Formal Concept Analysis. This neighborhood is based on 115 formal objects. The initial number of formal attributes for this example was 623, after reducing the context this was lowered to 40. This offers a tremendous reduction in context complexity, and therefore computation time but these numbers also reflect the need to search a subset of conceptual neighborhood.

A main question in the design of SearchSleuth is whether the definition of Exact Siblings provides sufficient space for proximity search of neighboring categories. The remainder of the paper addresses this issue in detail.

5 Distance, Similarity and Siblings

We have two measures to consider the proximity of formal concepts. In addition to similarity (s) defined in Eqn. (1) we also have for two formal concepts (A, B), (C, D),

$$d((A,B),(C,D)) := \frac{1}{2}\left(\frac{|A \backslash C| + |C \backslash A|}{|G|} + \frac{|B \backslash D| + |D \backslash B|}{|M|}\right)$$

where d the *distance* of the concepts (A, B), (C, D) [10]. To ease comparison between the two measures, let

$$s'((A,B),(C,D)) := 1 - s((A,B),(C,D))$$

Let us first note that s' and d are metrics in the mathematical understanding. That is, d satisfies for arbitrary concepts x, y, z: $d(x, y) \geqslant 0$ and $d(x, y) = 0 \Leftrightarrow x = y$ (non-negativity and identity of indiscernibles), $d(x, y) = d(y, x)$ (symmetry), and $d(x, z) \leqslant d(x, y) + d(y, z)$ (triangle inequality). The triangle inequalities can easily be shown by straight-forward computations, and the remaining properties are easily to be seen.

Next, note that we have

$$s'((A,B),(C,D)) = \frac{1}{2}\left(\frac{|A \cup C| - |A \cap C|}{|A \cup C|} + \frac{|B \cup D| - |B \cap D|}{|B \cup D|}\right) \tag{2}$$

$$d((A,B),(C,D)) = \frac{1}{2}\left(\frac{|A \cup C| - |A \cap C|}{|G|} + \frac{|B \cup D| - |B \cap D|}{|M|}\right) \tag{3}$$

Comparing Eqns. (2) and (3), we see that they differ in that in (2), we divide through $|A \cup C|$ and $|B \cup D|$, whereas in (3), we divide through $|G|$ and $|M|$. Therefore s' is a local distance, focusing on the shared attributes and objects of the two formal concepts being compared, and d is a global distance, using all the attributes and objects in the context. The choice of measurement to use therefore depends on the sensitivity of the proximity measure required. The preferred approach for SearchSleuth is proximity in the conceptual neighborhood to the current formal concept. Therefore the local measure is considered most suitable.

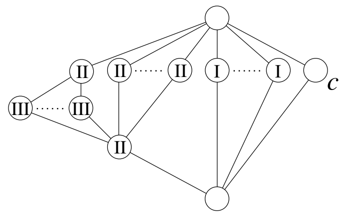
One can however easily combine the two measures. Let $l \in [0, 1]$, measuring the desire of a local point of view ($l = 0$ means the user wants a purely global point of view, and $l = 1$ means the user wants a purely local point of view). Then the corresponding distance (*dist*) measure is,

$$dist((A, B), (C, D)) := l \cdot s'((A, B), (C, D)) + (1 - l) \cdot d((A, B), (C, D)).$$

6 Relationship between Metric and Sibling Explored

The basic approach of SearchSleuth is to explore the 'conceptual neighborhood' of a given concept. To grasp this 'conceptual neighborhood', SearchSleuth takes advantage of two fundamentally different notions of neighboorhood. On the one hand, we use the lattice-theoretic notions of siblings, which do do not take the sizes of the concept-extents or intents into account. On the other hand, we use the notions of similarity and distance metrics are set-theoretic notions (they do not take the lattice-order into account). In the next two pararaphs, we first investigate the different types of siblings, and then the two kinds of similarity metrics.

The notions of siblings is more fine-grained divided into exact siblings (Type I), parent- and child siblings (Type II), and general siblings (Type III). Obviously, this is a hierarchy of types: Each Type I sibling is a Type II sibling, and each Type II sibling is a Type III sibling. Besides this inclusions, we cannot provide any general estimations on the number of the different types of siblings. To be more precise: If $n_{\mathrm{I}}, n_{\mathrm{II}}, n_{\mathrm{III}} \in \mathbb{N}_0$ are three numbers with $n_{\mathrm{I}} \leqslant n_{\mathrm{II}} \leqslant n_{\mathrm{III}}$ and $n_{\mathrm{II}} \neq 1$, then there exists a lattice with an element c which has n_{I} Type I siblings, n_{II} Type I siblings, and n_{III} Type I siblings. An example for such a lattice is given below. In the diagram, for each sibling of c, the most special type the sibling belongs to is inscribed into its node. That is in the diagram, n_{I} nodes are labelled with 'I', $n_{\mathrm{II}} - n_{\mathrm{I}}$ nodes are labelled with 'II', and $n_{\mathrm{III}} - n_{\mathrm{II}} - n_{\mathrm{I}}$ nodes are labelled with 'III'.



For the notions of local (s') and global (d) distance, a somewhat similar consideration applies. Due to Eqns. (2) and (3), in each lattice, for any concepts

x, y, we have

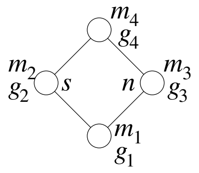
$$s'(x,y) \geqslant d(x,y) \quad .$$

On the other hand, there are examples (one is given in the following subsection) of lattices where there are two concepts c, n which are arbitrary close with respect to the local, but arbitrary distant with respect to the global distance.

The question remains whether there are dependencies between the lattice-theoretic (i.e., siblings) and the set-theoretic (i.e., metrics) notions of conceptual neighborhood. We will investigate some examples in the following sections. As these examples will show, the two notions are somewhat orthogonal but complementary in determining the most appropriate related categories in SearchSleuth.

6.1 Exact Siblings (*ES*) and Proximity Metrics

We first consider an example where we have an exact sibling n of a concept c, and we investigate whether we can draw some conclusions about the local or global distance between c and n. The example we consider is the following concept-lattice:



In this diagram, g_1, g_2, g_3, g_4 resp. m_1, m_2, m_3, m_4 do *not* denote objects or attributes, but the *numbers* of objects resp. attributes which generate the concept. For example, for $c = (G_2, M_2)$, we have $g_2 = |G_2| - |G_1|$. That is, the g_i and m_i are the numbers of objects and attributes in the common diagrams of concept lattices. Two concepts are given names, namely c and n. We have:

$$s(c,n) = \frac{1}{2}\left(\frac{g_1}{g_1+g_2+g_3} + \frac{m_4}{m_2+m_3+m_4}\right)$$
$$d(c,n) = \frac{1}{2}\left(\frac{g_2+g_3}{g_1+g_2+g_3+g_4} + \frac{m_2+m_3}{m_1+m_2+m_3+m_4}\right)$$

For fixed $g_2, g_3, g_4, m_1, m_2, m_3$ (e.g., $g_2 = g_3 = g_4 = m_1 = m_2 = m_3 = 1$), we have

$$\lim_{\substack{g_1\to\infty\\ m_4\to\infty}} s'(c,n) = 1 - \frac{1}{2}(1+1) = 0 \qquad \text{and} \qquad \lim_{\substack{g_1\to\infty\\ m_4\to\infty}} d(c,n) = \frac{1}{2}(0+0) = 0$$

i.e., c and n can be arbitrarily similar with respect to both s' and d.

On the other hand, for fixed g_1, g_3, m_3, m_4, we have

$$\lim_{\substack{g_2\to\infty\\ m_2\to\infty}} s'(c,n) = 1 - \frac{1}{2}(0+0) = 1 \qquad \text{and} \qquad \lim_{\substack{g_2\to\infty\\ m_2\to\infty}} d(c,n) = \frac{1}{2}(1+1) = 1$$

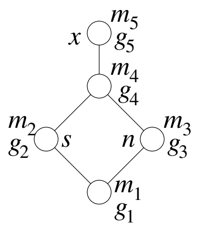
(similar for g_2, m_3, and g_3, m_2, and g_3, m_3). That is, c and n can be arbitrarily different (again with respect to s and to d).

Now let $\varepsilon_1, \varepsilon_2 > 0$. Let g_3, g_4, m_1, m_3 be fixed. By *first* choosing g_2 and m_2 sufficiently large, we can achieve $s(x,n) < \varepsilon_1$, and by *then* choosing g_1, m_4 sufficiently large (which does not affect $s(c,n)$), we can achieve $d(c,n) < \varepsilon_2$. That is, we can achieve that in a local understanding (i.e., w.r.t. s'), the concepts c and n are very similar, whereas in a global understanding ((i.e., w.r.t. d), the concepts c and n are very distant.

To summarize this example: even for the most special case of being an exact sibling n of a given concept c, we cannot draw any conclusion about the local or global distance between c and n.

6.2 The Proximity of Type I Siblings versus Non Siblings

A concept n which is a sibling for a given concept c belongs, from a lattice-theoretic point of view, to the conceptual neighborhood of c; a concept x which is not a sibling of c does not belong to the conceptual neighborhood. Is this property reflected by the distances s' and d? We consider again an example where n even is an exact sibling of c.



In terms of similarity we have:

$$s_n := s(c,n) = \frac{1}{2}\left(\frac{g_1}{g_1+g_2+g_3} + \frac{m_4+m_5}{m_2+m_3+m_4+m_5}\right)$$
$$s_x := s(c,x) = \frac{1}{2}\left(\frac{g_1+g_2}{g_1+g_2+g_4+g_5} + \frac{m_5}{m_2+m_4+m_5}\right)$$

Note that we can have $g_1 = 0$, $m_1 = 0$, $g_4 = 0$, and $m_5 = 0$, but all other numbers must be $\geqslant 1$. Now, n could be more similar to c than x, equally similar, or less similar, as the following examples show.

g_1	g_2	g_3	g_4	g_5	m_1	m_2	m_3	m_4	m_5	$2 \cdot s_n = 2 \cdot s(c,n)$	$2 \cdot s_x = 2 \cdot s(c,x)$
1	1	1	0	1	1	1	1	1	1	$1/3 + 2/4 = 5/6$	$2/3 + 1/3 = 1$
1	1	1	1	1	0	1	1	1	1	$1/3 + 2/4 = 5/6$	$2/4 + 1/3 = 5/6$
1	1	1	1	1	0	1	1	1	0	$1/3 + 1/3 = 2/3$	$2/4 + 0/2 = 1/2$

Running a computer-program checking all values for g_i and m_i with a threshold of 8 yields:

$s(c,n) > s(c,x)$	$s(c,n) < s(c,x)$	$s(c,n) = s(c,x)$
804.068.208	913.112.127	2.746.449

Therefore the cases in which $s(c,n) > s(c,x)$ and $s(c,n) < s(c,x)$ do not significantly differ and we cannot conclude (at least for this toy-example) that siblings are *generally* more similar than non-siblings.

Similarly, repeating the analysis in terms of the distance metric d, we have:

$$d_n := s(c,n) = \frac{1}{2}\left(\frac{g_2+g_3}{g_1+\cdots+g_5} + \frac{m_3+m_4}{m_1+\cdots+m_5}\right)$$
$$d_x := s(c,x) == \frac{1}{2}\left(\frac{g_4+g_5}{g_1+\cdots+g_5} + \frac{m_2+m_4}{m_1+\cdots+m_5}\right)$$

$g_1g_2g_3g_4g_5$	$m_1m_2m_3m_4m_5$	$2 \cdot d_n = 2 \cdot d(c,n)$	$2 \cdot d_x = 2 \cdot d(c,x)$	result
0 1 1 0 2	0 1 1 2 0	$2/4 + 2/4 = 1$	$2/4 + 3/4 = 5/4$	$d_1 < d_2$
0 1 1 1 1	0 1 1 1 0	$2/4 + 2/3 = 7/6$	$2/4 + 2/3 = 7/6$	$d_1 = d_2$
0 1 1 0 1	0 1 1 1 0	$2/3 + 2/3 = 4/3$	$1/3 + 2/3 = 1$	$d_1 > d_2$

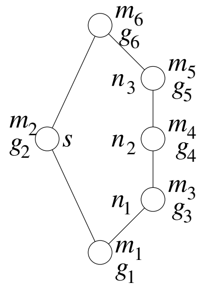
$d(c,n) > d(c,x)$	$d(c,n) < d(c,x)$	$d(c,n) = d(c,x)$
908.328.121	788.136.280	23.462.383

Again the cases $d(c,n) > d(c,x)$ and $d(c,n) < d(c,x)$ do not significantly differ.

To summarize this example: even for the most special case of being an exact sibling n of a given concept c, we cannot draw any conclusion that n is closer to c compared to a non-sibling.

6.3 The Proximity of Type II versus Type III Siblings

We have different strengths of being a sibling. We still could hope that this is reflected by the metrics. In the following example, we consider Type II siblings of a concept c with the more general Type III siblings and check whether the Type II siblings are closer to c than the Type III siblings.



In terms of similarity we have:

$$s_1 := s(c, n_1) = \frac{1}{2}\left(\frac{g_1}{g_1 + g_2 + g_3} + \frac{m_6}{m_2 + m_3 + m_4 + m_5 + m_6}\right)$$
$$s_2 := s(c, n_2) = \frac{1}{2}\left(\frac{g_1}{g_1 + g_2 + g_3 + g_4} + \frac{m_6}{m_2 + m_4 + m_5 + m_6}\right)$$
$$s_3 := s(c, n_3) = \frac{1}{2}\left(\frac{g_1}{g_1 + g_2 + g_3 + g_4 + g_5} + \frac{m_6}{m_2 + m_5 + m_6}\right)$$

In this example, there is no order relationship between s_1, s_2, and s_3. We can have $s_1 < s_2 < s_3$ or $s_3 < s_1 < s_2$ or $s_1 = s_2 < s_3$ etc. Any combination is possible. The following table shows examples for all possible strict orders of s_1, s_2, s_3 (examples for cases like $s_1 = s_2 < s_3$ are left out due to space limitations).

g_1	g_2	g_3	g_4	g_5	g_6	m_1	m_2	m_3	m_4	m_5	m_6	$2 \cdot s_1$	$2 \cdot s_2$	$2 \cdot s_3$	result
1	1	1	1	1	1	1	1	2	1	1	2	$1/3 + 2/7$	$1/4 + 2/5$	$1/5 + 2/4$	$s_1 < s_2 < s_3$
1	1	2	1	2	1	1	2	2	1	2	2	$1/4 + 2/9$	$1/5 + 2/7$	$1/7 + 2/6$	$s_1 < s_3 < s_2$
1	1	1	1	1	1	1	1	1	1	1	2	$1/3 + 2/6$	$1/4 + 2/5$	$1/5 + 2/4$	$s_2 < s_1 < s_3$
1	1	1	1	1	1	1	1	1	1	2	2	$1/3 + 2/7$	$1/4 + 2/6$	$1/5 + 2/5$	$s_2 < s_3 < s_1$
2	2	2	1	2	1	1	1	2	1	2	1	$2/6 + 1/7$	$2/7 + 1/5$	$2/9 + 1/4$	$s_3 < s_1 < s_2$
1	1	1	1	1	1	1	2	1	1	2	1	$1/3 + 1/7$	$1/4 + 1/6$	$1/5 + 1/5$	$s_3 < s_2 < s_1$

Similarly, repeating the analysis in terms of the distance metric, we have:

$$d_1 := d(c, n_2) = \frac{1}{2}\left(\frac{g_2 + g_3}{g1 + \cdots + g_6} + \frac{m_2 + m_3 + m_4 + m_5}{m_1 + \cdots + m_6}\right)$$
$$d_2 := d(c, n_2) = \frac{1}{2}\left(\frac{g_2 + g_3 + g_4}{g1 + \cdots + g_6} + \frac{m_2 + m_4 + m_5}{m_1 + \cdots + m_6}\right)$$
$$d_3 := d(c, n_3) = \frac{1}{2}\left(\frac{g_2 + g_3 + g_4 + g_5}{g1 + \cdots + g_6} + \frac{m_2 + m_5}{m_1 + \cdots + m_6}\right)$$

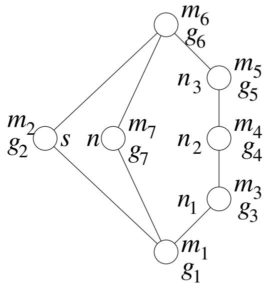
Again here is no relationship between d_1, d_2, and d_3, and any combination is possible, as the following table shows:

g_1	g_2	g_3	g_4	g_5	g_6	m_1	m_2	m_3	m_4	m_5	m_6	$2 \cdot d_1$	$2 \cdot d_2$	$2 \cdot d_3$	result
1	1	1	1	1	1	1	1	1	1	2	1	$2/6 + 5/7$	$3/6 + 4/7$	$4/6 + 3/7$	$d_1 < d_2 < d_3$
1	1	1	1	1	1	1	2	1	2	2	2	$2/6 + 7/10$	$3/6 + 6/10$	$4/6 + 4/10$	$d_1 < d_3 < d_2$
1	1	1	1	1	1	1	2	2	1	2	2	$2/6 + 7/10$	$3/6 + 5/10$	$4/6 + 4/10$	$d_2 < d_1 < d_3$
1	1	1	1	1	1	1	1	2	1	1	1	$2/6 + 5/7$	$3/6 + 3/7$	$4/6 + 2/7$	$d_2 < d_3 < d_1$
1	1	1	1	1	1	1	1	1	2	1	1	$2/6 + 5/7$	$3/6 + 4/7$	$4/6 + 2/7$	$d_3 < d_1 < d_2$
1	1	1	1	1	1	1	1	2	2	1	1	$2/6 + 6/8$	$3/6 + 4/8$	$4/6 + 2/8$	$d_3 < d_2 < d_1$

To summarize the analysis: we cannot draw any conclusion that Type II siblings of a concept c are closer to c, using s' or d, than the weaker Type III siblings. That is, we cannot say that Type II siblings better represent related categories than Type III siblings.

6.4 Type I versus Type II versus Type III Siblings

This example compares now all three types of siblings are now



In terms of similarity we have:

$$\begin{aligned}
s_n &:= s(c,n) = \frac{1}{2}\left(\frac{g_1}{g_1+g_2+g_7} + \frac{m_6}{m_2+m_6+m_7}\right)\\
s_1 &:= s(c,n_1) = \frac{1}{2}\left(\frac{g_1}{g_1+g_2+g_3} + \frac{m_6}{m_2+m_3+m_4+m_5+m_6}\right)\\
s_2 &:= s(c,n_2) = \frac{1}{2}\left(\frac{g_1}{g_1+g_2+g_3+g_4} + \frac{m_6}{m_2+m_4+m_5+m_6}\right)\\
s_3 &:= s(c,n_3) = \frac{1}{2}\left(\frac{g_1}{g_1+g_2+g_3+g_4+g_5} + \frac{m_6}{m_2+m_5+m_6}\right)
\end{aligned}$$

Note that changing g_7 and m_7 does not affect the similarity measures between c and n_1, n_2, n_3, resp. According to Section 6.1, for high values of g_7 and m_7, the similarity between c and n (i.e., d) decreases. So we easily can use the values for $g_1, \ldots, g_6, m_1, \ldots, m_6$ of the last example to get all possible orderings of s_1, s_2, s_3, and choose g_7 and m_7 such that $d < s_1, s_2, s_3$. That is, Type II siblings)are not necessarily more similar to c than Type III siblings.

In fact, we have again that for s, all 24 strict orders of s, n, s_1, s_2, s_3 can appear. And the same holds for d. (As we have both for s and d 24 such strict orders, thus 48 examples, these examples are not provided due to space limitations). In short, no general statements which render some preference for siblings used as Related Categories in terms of similarity and distance.

7 Conclusion

The notion of Type I, II and II siblings is a purely lattice-theoretic notion, whereas the notion of distance and similarity is a purely set-theoretic notion. As our examples show, these notions are somewhat complementary. In order to find similar concepts to a given concept (related categories), there is no hint that one should start with the immediate sibling neighbors of that concept. This might sound disappointing at a first glance, but in practice our observations lead to an important design feature in SearchSleuth. Computationally, the neighboring siblings to the current formal concept (whether of Type I, II or III) are the easiest concepts to compute and therefore represent natural candidates for related category search. In this case the search of the sibling space proceeds by considering related categories with the best distance and similarity stored in each of the neighboring siblings concepts for the current concept.

SearchSleuth, extends current FCA Internet search engines by positioning the user within the information space, rather than placing the user arbitrarily or presenting the entire space. This allows generalisation and categorisation operations to be performed against the current query concept. SearchSleuth overcomes a number of practical difficulties in the use of FCA for Internet Search, namely a practical approach to the construction of a sparse context and the categorisation operation, where the conceptual focus is moved to a sibling concept of the search concept. These paper explains how related categories are derived using a combination of order-theoretic notions of neighborhood in combination of set-theoretic definitions of concept similarity.

References

1. Carpineto, C., Romano, G.: Exploiting the potential of concept lattices for information retrieval with credo. Journal of Universal Computer Science **10**(8) (2004) 985–1013
2. Koester, B.: FooCA - Web Information Retrieval with Formal Concept Analysis. Diploma, Technische Universität Dresden (2006)
3. Koester, B.: Conceptual knowledge processing with google. In: Contributions to ICFCA 2006. (2005) 178–183
4. Ducrou, J., Eklund, P.: Searchsleuth: the conceptual neighbourhood of an internet query. In: 5th International Conference on Concept Lattices and Their Applications (CLA 2007), http://CEUR-WS/Vol-331/Ducrou.pdf (2007) 249–260
5. Kim, M., Compton, P.: Formal Concept Analysis for Domain-Specific Document Retrieval Systems. In: Australian Joint Conference on Artificial Intelligence. (2001) 237–248
6. Kim, M., Compton, P.: The perceived utility of standard ontologies in document management for specialized domains. International Journal of Human-Computer Studies **64**(1) (2006) 15–26
7. Ducrou, J., Vormbrock, B., Eklund, P.: FCA-based Browsing and Searching of a Collection of Images. In: Proceedings of 14th International Conference on Conceptual Structures. LNAI4068, Springer (2006) 203–214
8. Ducrou, J., Eklund, P.: Faceted document navigation using conceptual structures. In Hitzler, P., Schärf, H., eds.: Conceptual Structures in Practice, CRC Press (2008) 251–278
9. Ducrou, J., Eklund, P.: An intelligent user interface for browsing and search mpeg-7 images using concept lattices. International Journal of Foundations of Computer Science **19**(2) (2008) 359–381
10. Lengnink, K.: *"Ahnlichkeit als Distanz in Begriffsverbänden.* In G Stumme, R.W., ed.: *Begriffliche Wissensverarbeitung: Methoden und Anwendungen*, Springer (2001) 57–71
11. Saquer, J., Deogun, J.S.: Concept aproximations based on rough sets and similarity measures. In: Int. J. Appl. Math. Comput. Sci. Volume 11. (2001) 655 – 674