SearchSleuth: The Conceptual Neighbourhood of an Web Query

Jon Ducrou and Peter Eklund

jrd990,peklund@uow.edu.au
School of Computer Science and Software Engineering
University of Wollongong
Northfields Ave
Wollongong, 2522, Australia.

Abstract. This paper presents **SearchSleuth**, a program developed to experiment with a form of automatic local analysis that extends the standard Web search interface to include a conceptual neighbourhood focused on a formal concept derived from the query. The conceptual neighbourhood is displayed with upper neighbours representative of a generalisation operation, and lower neighbours representative of a specialisation operation. **SearchSleuth** also introduces a notion of a *categorisation* operation, where the conceptual focus can shift to a sibling concept of the search concept.

1 Introduction and Background

Existing FCA-based Web search applications focus on providing automatic local analysis of search results for query refinement and labeled object (document or document fragmets) clustering. This is usually done via the creation of a conceptual space as a concept lattice from the search results which is then displayed in various ways. Such methods fail to create a concept representing the query itself within the space – meaning the information space is representative of the results returned by the query rather than of the query itself.

SearchSleuth differs in that it creates a conceptual space as a neighbourhood of the *search concept*. This neighbourhood is comprised of generalisations (upper neighbours), specialisations (lower neighbours) and categorisation (siblings). The design therefore relies on the idea of centering the conceptual space (concept lattice) around the focal search concept. The resulting query refinement operations are more closely coupled to the search terms used in the creation of the conceptual space.

Carpineto and Romano's application, *CREDO* [1], uses an iceberg lattice to generate clusters for given search terms. Searched terms are submitted to an search engine and results returned are organized into a formal context. The context is built with the search engine results as objects and the terms found in the result summaries as attributes. The most general concepts of the concept lattice are computed and displayed as a tree. The user can then interact with the tree to view named clusters of the results. *CREDO* provides a front-end that

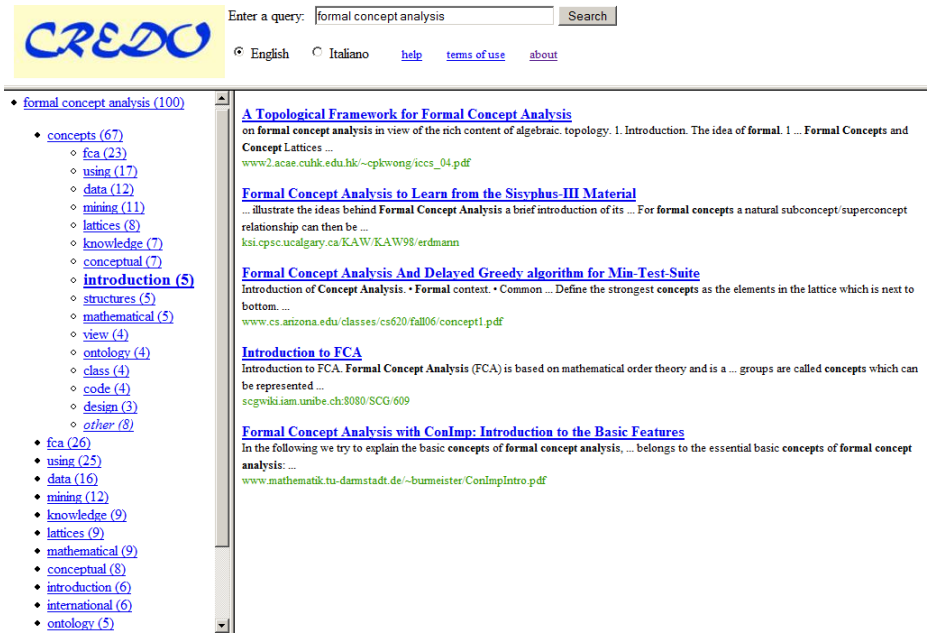

Fig. 1. Carpineto and Romano's *CREDO* Web Search Application [1]: An example of *search*. *CREDO* displays the uppermost concepts derived from keywords featured in search results.

extends the collected result set, and provides a landscape over the information space which can be used to aid in search retrieval. A screenshot of *CREDO* is shown in Fig. 1.

Another FCA Web search application is Koester's FooCA [2, 3]. FooCA provides one of the richest interfaces for the construction of a formal context based on Web search results. By taking results as formal objects and terms found within the results as formal attributes, a formal context is created and presented to the user. A screenshot of FooCA is shown in Fig. 2.

2 Navigation and Conceptual Neighborhoods

Kim and Compton [4, 5], and their prototype *KANavigator*, presented a document navigation paradigm using FCA and a neighborhood display. *KANavigator* uses annotated documents that can be browsed by keyword and displayed the direct neighborhood (in particular the lower neighbours) to the user. This system emphasized the use of textual labels as opposed to visualised structure.

ImageSleuth [6] used a similar interaction approach to allow exploration of image collections. By showing upper and lower neighbors of the current formal concept and allowing the inclusion/exclusion of these formal concepts, users can refine or generalize their position in the information space by navigating the

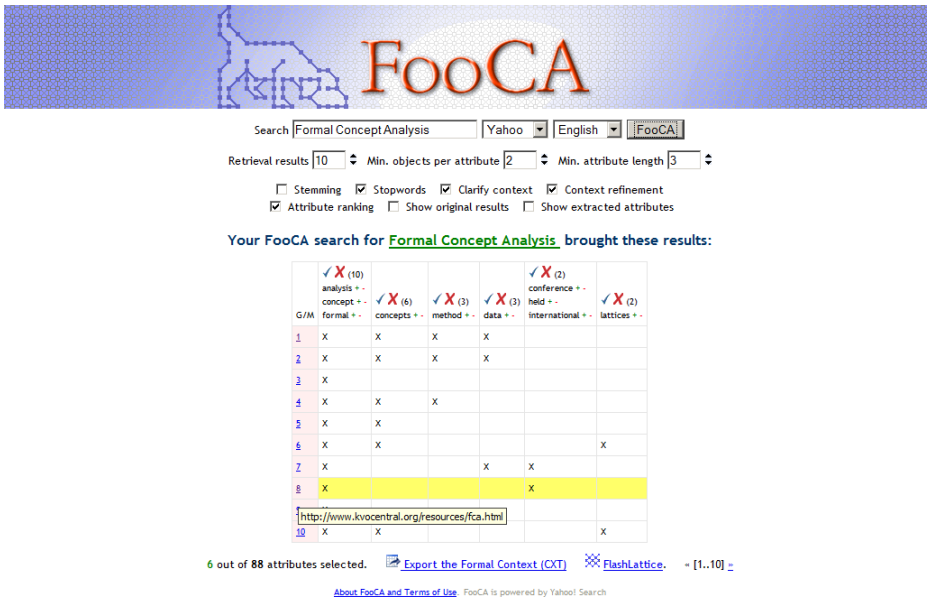

Fig. 2. Koester's FooCA Web Search Application [2, 3]: An example of *improvement*. By allowing users see and access the quality of the various terms in the current search results, successive search results increase in quality.

concept lattice. ImageSleuth was additionally aided by the use of *perspectives* – another name for conceptual scales – that could be combined to define the attribute set of the lattice which formed the information space.

ImageSleuth uses most of its interface 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 subconcepts. For every upper neighbour (C, D) of the current concept (A, B) the user is presented with an interface that allows them to remove the set $B \setminus D$ of attributes from the current intent. Dually, for every lower neighbour (E, F) the user may include the set $F \setminus B$ of attributes which takes her to this lower neighbour. 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 used aided in navigation of image collections and it therefore gives rise to the re-use of this interaction paradigm in **SearchSleuth**.

A formal concept A is said to be the upper neighbour (or cover) of a formal concept B iff we have $A > B$, and there is no intermediate formal concept C with $A > C > B$. A formal concept A is said to be the lower neighbour of (or covered by) a formal concept B iff we have $A < B$, and there is no formal

concept C with $A < C < B$. Upper and lower neighbour of a formal concept C are written as $UN(C)$ and $LN(C)$ respectively in this paper.

3 The Domain of SearchSleuth

Web search is a difficult problem given the Internet is immeasurably large and constantly being changed and altered, both in content and structure. Traditionally, search is done by entering one or more keywords into a search engine, then reviewing the results the appropriate Web site is found. This process requires a good understanding of the link between the search terms and their results.

Most search engines return a ranked list of results and these usually take the form of a document fragment or snippet that includes the URL of the result, its title, a short summary of the document and various details such as date last accessed. In CREDO and FooCA, it is these text-based snippets of the search results that provide the material to create a concept lattice that forms the basis for a conceptual information space to navigate the search results. One problem with the transition from Web search results to FCA is that the search result ranking information is lost. All results are treated equally, this issue is usually addressed by reproducing the rank ordering on any result set that is realized but this can be costly to performance.

Another problem, particularly with Web search, is that page ranking methods use techniques such as link structure analysis, page popularity and referring pages to condition page proximity to a search query. As such, that it cannot be assumed that all results of a multiple term query will contain all search terms from the original query. Even a single search term may yield search results that do not contain the search term. This may seem counter intuitive, but if there are enough Web pages linked to the result page that *does* contain that search term, that page's rank may be inflated high enough to feature in the result set.

Like *CREDO* [1] and FooCA [2, 3], **SearchSleuth** uses the 'result has term' representation for building a formal context. This means each result from the search is considered an object, and all terms contained in the results title and summary are considered attributes. In the case of **SearchSleuth** (and optionally in FooCA) all words in the result are stop-word filtered and stemmed to their lexical root. This reduces the size and complexity of the formal content by the reduction of terms with common lexical roots; displaying 'car' and 'cars' as a single term. By extension this reduces the size of the concept lattice formed from the formal context.

CREDO does not include the original query terms when creating the context. This omission is made because in most cases it would be expected that all results contain the original query terms. As Web search does not behave exclusively as a boolean search for keywords in pages, this assumption does not always hold and the original query terms may be absent from the context as previously discussed. *CREDO*, unlike **SearchSleuth**, displays two levels of the lattice as a tree with users initially placed at the top-most concept. The display is initially restricted to a single level, by with user interaction a single concept can be expanded at a

time. This reduces clutter and also user confusion; users are not displayed with multiple tree branches at the same level with the same label.

The FooCA application builds and displays the entire formal context derived from the query terms returned from the search engine. The formal context is built without the originating query terms and the interface provides numerous controls over the information space created by the concept lattice. The user is shown the entire formal content in one cross-table. Cross-tables are less than perfect for human interpretation but the display does permit great detail. By viewing the entire formal content, the user is never positioned within the induced information space, and thus is without a perspective that is defined by the query.

SearchSleuth attempts to immerse the user in the information landscape, with a perspective centered on the query used to create it. The design logic is that this should give more insight into the meaning of the query with respect to other terms that appear in the result set.

Also, **SearchSleuth** is the only FCA-based Web search that uses multiple searches per query. This is done to expand the information space. The ancillary searches each yield half the results of the main query search. This expands the bounds of the information space with more general queries and provides for better clustering of facts as is explained in the next section.

4 Design Approach of SearchSleuth

SearchSleuth follows from the usability testing of **ImageSleuth** and employs the same conceptual neighbourhood 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 Web, as a conceptual neighborhood would be computationally prohibitive.

The formal context for **SearchSleuth** is created on demand for each query; this suits the dynamic nature of the Web. The formal objects are induced from the top X results from the query, and the formal attributes are the terms contained in the title and summary of each result (like *CREDO* and *Fooca*). Terms are extracted from the title and summary after stemming and stop-word filtering has been performed. As mentioned, 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 any noticeable reduction in semantic quality. For presentation purposes the stemming is reversed, this way terms rendered unintelligible by stemming are reformed in a meaningful manner.

The formal 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 greatly decreases the computational overhead involved in most FCA algorithms. The reduction rarely effects the computed conceptual neighborhood as the terms removed are scarce within the information space.

Once the formal context is constructed, the search concept is created. This is done by taking the original query terms as attributes and deriving the formal concept. The upper neighbours of this formal concept are then derived and used to expand the formal 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 search results.

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

Once the formal context is expanded, the search concept is recomputed as it may have been invalidated by this process. The upper and lower neighbours are computed next, then the sibling concepts. The explanation of neighbor is found in Section 2. Sibling concept are then calculated by finding all of the lower neighbours of upper neighbors which are upper neighbours of lower neighbors. Put another way, siblings are the removal of an attribute (or attributes) that defines an upper neighbor, and the inclusion of an attribute (or attributes) that defines a lower neighbor.

Consider the set of concepts X, $UN(X)$ is defined as the union of all upper neighbours of the concepts in X.

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

Dually, consider the set of concepts X, $LN(X)$ is defined as the union of all lower neighbours of the concepts in X.

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

For a lattice with concept, C, the set of concepts, S is the siblings of C, defined:

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

An example is shown in Fig. 3; concepts with a grey backing are siblings of the concept marked with a 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. The primary feature of the display is the text entry box in which the query is entered. It is considered representative of the derived search concept, and thus is centered in the display. This is shown is Fig. 4.

Upper neighbors are shown above this text entry box, displayed as text labels. The labels are the attributes which would be removed to navigate to that upper neighbor. These labels are preceded by a minus symbol (`-`) – the standard exclusion operator in most search engine interfaces – to reinforce the notion of *removal*: shown in Fig. 4.

Lower neighbors are similarly displayed, but placed below the text entry box. These labels are the attributes which would be added to navigate to that lower

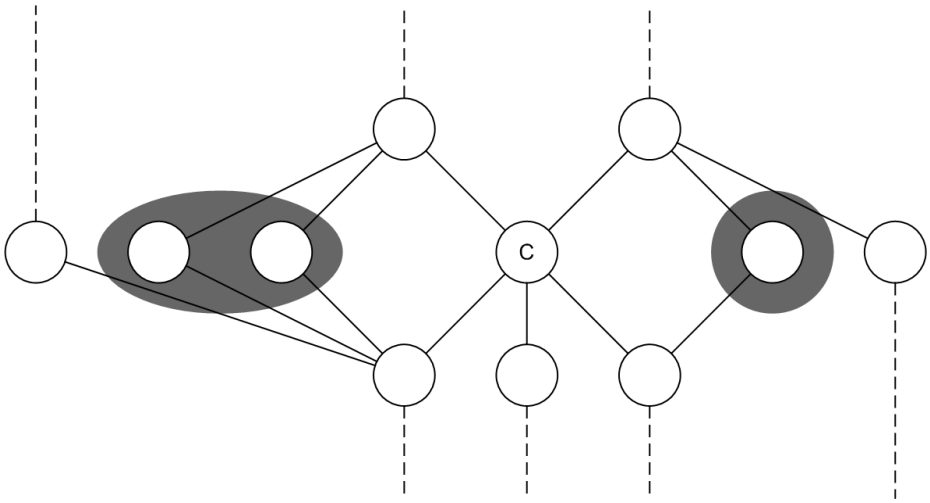

Fig. 3. Diagram demonstrating the *sibling* concepts of the concept labelled with a C.

neighbour. Like upper neighbor labels, these labels are preceded by a symbol to reinforce the labels meaning, namely the plus symbol (+) and the notion of *include*: shown in Fig. 4.

The display order of the upper and lower neighbors is determined by extent size, the concepts with larger extents displayed first (left-most). Extent is representative of the importance or prominence of a term within the current information space. This means the inclusion of the first upper or lower neighbor shown will have the least effect on the extent size current concept, if the transition were to be made on a static formal context. Extent is also used to aid in the coloring of the labels background. The larger the extent on a lower neighbor, the deeper the blue behind that concepts label. Upper neighbors are displayed with the same principle but with red.

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. Another method is to apply a vector space model of MPEG-7 images [8] and then apply similarity measures for multi-dimensional feature spaces. **ImageSleuth** explores a different approach by using variations on defined distance [9] and similarity [10] metrics in the FCA literature in order to find relevant concepts.

Siblings are shown to the right of the text entry box, see shown in Fig. 4. The complete intent of these concepts is displayed within square brackets preceded by a tilde symbol (`~[...]`). This helps groups the concept intents and aid distinguishing between the concepts. Unlike upper and lower neighbours, which are sorted by extent size, siblings are ordered by *similarity* to the search concept. The similarity metric is based on work by Lengnink[9] and adapted for

practical application to information retrieval and browsing in **ImageSleuth**. Similarity so defined uses the size of the common objects and attributes of the concepts. For two concepts (A, B) and (C, D):

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

The similarity metric is used to order the sibling concepts, while highlighting remains based on the extent size for each concept. Coloring on sibling labels is based on grey shades, the darker the larger the concept's extent.

The conceptual neighborhood representation is followed by the results of the search shown in the lower half of the screen shown in Fig. 4.

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 in this way changes the result set, and in order for the information to be valid it needs to be recomputed.

An example of the interface is show in Fig. 4. The search concept is based on the three query terms `formal concept analysis`. It shows a single upper neighbor labeled with `-analysis` (which represented the concept with the intent (`formal, concept`)) 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 and provide some intuitive validation of the approach.

This neighborhood is based on 115 formal objects. The initial number of formal attributes for this example was 623, after reducing the formal context this was decreased to 40. Thus, stemming and stop-word removal offer a tremendous reduction in context complexity, and therefore computation time.

Performance of the prototype shows that the vast majority of the time taken to display the results of a query is spent transferring search results from the search server. For the page shown in Fig. 4, computation of formal concepts took a total of 422ms, while transfer between Yahoo servers and the **SearchSleuth** host, comprising of two search requests, took 4062ms.

A more detailed example is shown in the next section.

4.1 An Example Interaction for SearchSleuth

The following exemplifies a possible navigation of a dynamic information space centered with **SearchSleuth**[1], initially, on the search term '`tiger`'[2].

In Fig. 5, it can be seen that the space has no generalisations or categorisations. This is because the search concept is the top-most concept of the lattice, and therefore *all* objects have the attribute `tiger`. Examples of what may be expected as specialisations of this search concept are:

[1] the reader is encouraged to follow the example by testing **SearchSleuth** which can be accessed from http://www.kvocentral.org/software/searchsleuth.html

[2] Searching for big cat codenames for Mac OS X versions is an unofficial, but well established, standard for FCA Web-search examples.

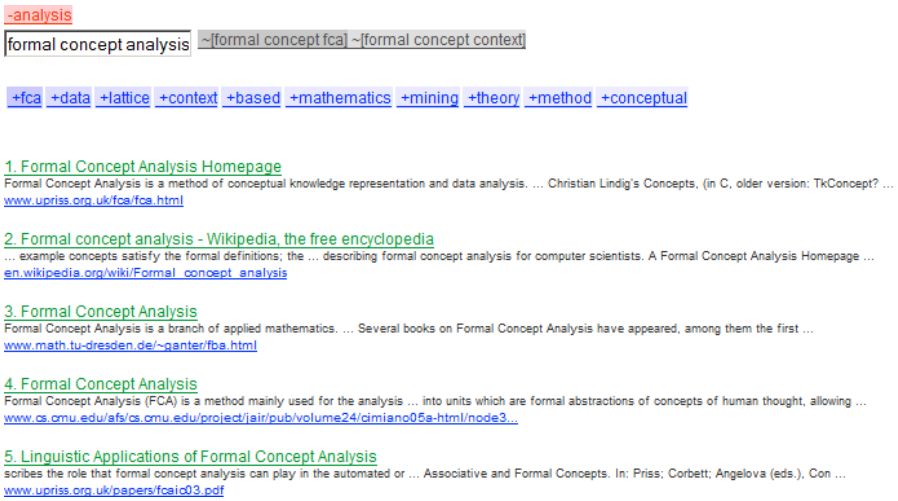

Fig. 4. SearchSleuth display, including top results, after a search for '`formal concept analysis`'.

Big Cat	Adding `+cat` would probably focus on the Tiger species.
Mac OS	Adding `+os` would probably focus on the operating system (OS) used by Mac computers.
Result Facets	`+information`, `feature`, `+photo` and `+facts` are types of results, and can guide the user.

Fig. 5. SearchSleuth display after a search for '`tiger`'.

Fig. 6 shows the addition of `information` via specialisation. It can be seen that this has changed the information space surrounding the query with more generic terms, such as `+pictures` and `site`. Also information about the animal tiger is made available, such as `endangered`, `conserving` and `white`. Cateogisations in this space focus on combining `tiger` with elements found by combining `tiger` with `information`, such as `tiger species` and `tiger wild`.

Fig. 7 shows the addition of `os` via specialisation from Fig. 5. This specialisation has created an information space focusing on the Apple Mac operating system, in particular version 10.4 which is also known as 'tiger'. The first two specialisations are representative of the company which makes the OS. The next two are in reference to the version of the OS. Categorizations in this space tend toward the removal of the term `tiger` in place of the focus on `os`.

Fig. 6. SearchSleuth display after clicking the '`+information`' link in Fig. 5.

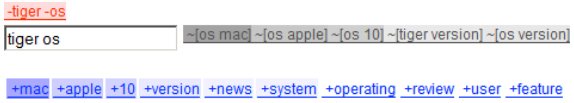

Fig. 7. SearchSleuth display after clicking the '`+os`' link in Fig. 5.

Specialising on `cat` from Fig. 5 gives the conceptual neighborhood shown in Fig. 8.

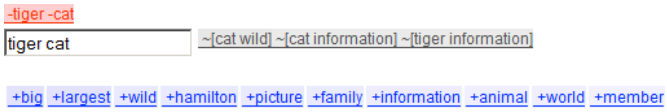

Fig. 8. SearchSleuth display after clicking the '`+cat`' link in Fig. 5.

Specialising on `feature` from Fig. 5 reveals an interesting categorisation shown in Fig. 9. Namely a categorization concept with the label `~[tiger woods]` which most likely refers to the golfer. This reveals another possible meaning for the term `tiger`.

The `tiger woods` search has a definite golf focus thus confirming the navigation intuitions.

5 Conclusion

SearchSleuth, extends current FCA Web search engines by positioning the user within the information space induced by the search formal contest, rather than placing the user arbitrarily within the space or presenting the entire space. This allows generalization and categorization operations to be performed against the current query concept induced by the search terms. **SearchSleuth** overcomes a number of practical difficulties in the use of FCA for Web Search, namely a practical approach to the construction of a sparse formal context and the categorization operation, where the conceptual focus can shift to a sibling concept of the induced search concept.

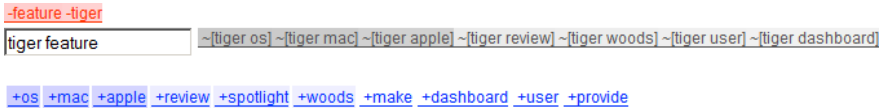

Fig. 9. SearchSleuth display after clicking the '`+feature`' link in Fig. 5.

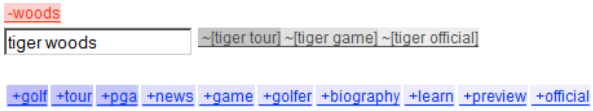

Fig. 10. SearchSleuth display after clicking the '∼`[tiger woods]`' link in Fig. 9.

References

1. Carpineto, C., Romano, G.: Exploiting the potential of concept lattices for information retrieval with credo. J. UCS **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. (2005) 178–183
4. Kim, M., Compton, P.: Incremental development of domain-specific document retrieval systems. In: 1st International Conference on Knowledge Capture. (2001)
5. Kim, M., Compton, P.: Formal concept analysis for domain-specific document retrieval systems. In: Lecture Notes in Computer Science. Volume 2256. (2001) 237
6. 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
7. Cigarrán, J.M., Gonzalo, J., Peñas, A., Verdejo, F.: Browsing search results via formal concept analysis: Automatic selection of attributes. In: Concept Lattices. Volume 2961/2004., Springer (2004) 74–87
8. Park, K.W., Lee, D.H.: Full-automatic high-level concept extraction from images using ontologies and semantic inference rules. In: The Semantic Web - ASWC 2006. LNAI4185, Springer (2006) 307–321
9. Lengnink, K.: Ähnlichkeit als Distanz in Begriffsverbänden. In G Stumme, R.W., ed.: Begriffliche Wissensverarbeitung: Methoden und Anwendungen, Springer (2001) 57–71
10. 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

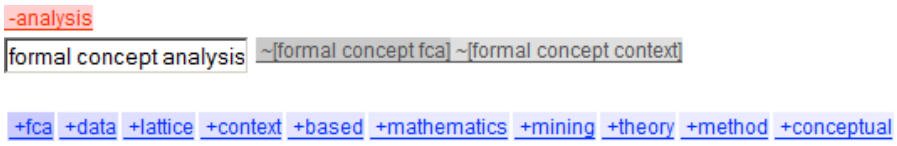

Fig. 11. SearchSleuth display after a search for '`formal concept analysis`'.