Semantic querying of data guided by Formal Concept Analysis

Víctor Codocedo[1], Ioanna Lykourentzou[1,2⋆], and Amedeo Napoli[1]

[1] LORIA - CNRS - INRIA - Univesité de Lorraine, BP 239, 54506 Vandœuvre-les-Nancy.
victor.codocedo@loria.fr, amedeo.napoli@loria.fr,
[2] Centre de Recherche Public Henri Tudor - 29, avenue John F. Kennedy L-1855
Luxembourg-Kirchberg, Luxembourg
ioanna.lykourentzou@tudor.lu

Abstract. In this paper we present a novel approach to handle querying over a concept lattice of documents and annotations. We focus on the problem of "non-matching documents", which are those that, despite being semantically relevant to the user query, do not contain the query's elements and hence cannot be retrieved by typical string matching approaches. In order to find these documents, we modify the initial user query using the concept lattice as a guide. We achieve this by identifying in the lattice a formal concept that represents the user query and then by finding potentially relevant concepts, identified as such through the proposed notion of *cousin concepts*. Finally, we use a concept semantic similarity metric to order and present retrieved documents. The main contribution of this paper is the introduction of the notion of *cousin concepts* of a given formal concept followed by a discussion on how this notion is useful for lattice-based information indexing and retrieval.

1 Introduction

As the amount of information grows, the ability to retrieve documents relevant to the needs of the user increasingly becomes more important. Several applications have been proposed, regarding this task, in the field of Information Retrieval (IR). However, as the information becomes more complex (not only text, but also multimedia documents) and specific (domain-oriented), the capacity to organize it becomes as important as the capacity to retrieve it.

Formal Concept Analysis (FCA) is a robust and widely used framework to organize objects based on their relations through their attributes in a concept lattice [6]. Concept lattices have been used in the past to support Information Retrieval tasks and they have been found to have better or comparable performance in relation to traditional approaches, such as Hierarchical Clustering and Best-Match Ranking. We argue that this performance can be further enhanced considering features as concept and semantic similarities and lattice navigation techniques.

⋆ The work of Ioanna Lykourentzou in the present project is supported by the National Research Fund, Luxembourg, and cofunded under the Marie Curie Actions of the European Commission (FP7-COFUND).

In this work, we present an approach to retrieve documents from a document-term concept lattice, considering that concepts can be *close* with respect to their position within the lattice and semantically *similar* to one another. We use both of these notions to find which are the most relevant documents for a given user query.

The rest of this paper is organized as follows: Section 2 presents the related research literature. Section 3 briefly introduces FCA and presents our proposed approach for navigating the lattice using the notion of *cousin concepts*, as well as for ranking the selected concepts with respect to their semantic similarity. Section 4 presents and discusses the experimental results and finally section 5 presents the conclusions of our work.

2 Related Work

2.1 Concept lattice-based Information Retrieval

Formal concept analysis is a data representation, organization and management technique with applications in many fields of information science, ranging from knowledge representation and discovery to logic and AI [15]. In the last decade researchers have also focused on examining the potential of FCA addressing problems in the field of Information Retrieval [14]. Under this light, the term Concept lattice-based Information Retrieval is used to describe the problem of retrieving information relevant to a given user query, when the document collection that contains the information is organized within a concept lattice. Some of the IR tasks that FCA and concept lattices have so far been applied on, include query refinement and expansion, integration of query and navigation and support of faceted search ([4, 2]). Among the most representative works in the field are the works of Carpineto and Romano, who introduce the method of Concept lattice-based ranking (CLR) [1].

The CLR method consists of three main steps: i) construction of the formal context of documents-terms and building of the corresponding concept lattice ii) insertion in the lattice of a new concept that represents the user query, using a subset of the attributes of the formal context and iii) retrieval and ranking of the relevant concepts using a nearest-neighbour approach, which depends on their topological path distance, within the lattice, from the original concept. The topological path metric used is called distance ”ring”, and it measures the radius of distance between two concepts, using as distance metric the length of the shortest path between them. The ring metric provides a partially ordered retrieval output, according to which all the documents that are equally distant from the original concept, i.e. belong to the same distance ring, are given the same ranking score.

Carpineto and Romano, also compare the CLR method with two other Information retrieval methods, namely Hierarchical Clustering-based Ranking (HCR)[7] and Best-match ranking (BMR). CLR is found to produce better results compared to HCR. Compared to BMR, it produces worst results when compared on the retrieval over the total document collection and better results, when only the first documents of the retrieval result are considered. However, CLR was better over both BMR, HCR when considering the retrieval of non-matching documents, (i.e. documents that do not match

the user query but share common terms with documents that do match the user query) The main advantage of the CLR method is that, in contrast to other statistical similarity measures that calculate the distance between two document representations using only the characteristics of those representations, the lattice allows to also incorporate the similarity that two document representations have in regards to the context, i.e. the whole document of collections, in which they are found.

The limitations of the traditional CLR method include firstly, the need to build the whole lattice before retrieving the related concepts. This issue determines the complexity and computational time required to address the problem, and it may result in non-realistic solutions for large document collections, like for instance the TREC dataset ([4]). Another issue, identified by the authors is that perhaps CLR should be combined with BMR, since they perform well in different types of documents (non-matching and matching respectively). Another set of limitations, more related to the present work, refers to the ranking method used and specifically to the fact that the retrieval and ranking of the related concepts is made taking into account only their topological relation with the original user query concept. Specifically, due to the use of topological distance rings as a metric of concept similarity, the CLR method does not distinguish between generalization and particularization, when moving from the concept of the original user query to other concepts. This limitation is critical, as it may lead to a loss of the semantic similarity between the retrieved and the original concept and it is explained in more detail in the section 3 when introducing our proposed method for concept-based information indexing and ranking.

To address these limitations, in this paper we propose a novel approach, which seeks to ensure semantic similarity with the original user query, both through the way that the lattice is traversed and through the way that the concepts are ranked. In particular, we introduce a new topological-based concept characteristic, called *cousin concepts*, to navigate the lattice and retrieve candidate related concepts. In parallel, for ranking the retrieved concepts we do not rely only on structural concept similarity features, but instead we use a metric that allows the weighting of both structural and semantic similarity aspects [5].

3 Methodology

In order to present our approach, first we present a brief description to Formal Concept Analysis (FCA). The basics of FCA are introduced in [6], but we recall some notions useful for its understanding in the following.

Data is encoded in a formal context $\mathcal{K} = (G, M, I)$, i.e. a binary table where G is a set of objects, M a set of attributes, and $I \subseteq G \times M$ an incidence relation. Two derivation operators, both denoted by $'$, formalize the sharing of attributes for objects, and, in a dual way, the sharing of objects for attributes:

$' : \wp(G) \longrightarrow \wp(M)$ with $A' = \{m \in M \mid \forall g \in A, gIm\}$

$' : \wp(M) \longrightarrow \wp(G)$ with $B' = \{g \in G \mid \forall m \in B, gIm\}$,

where $\wp(G)$ and $\wp(M)$ respectively denote the powersets of G and M. The two derivation operators $'$ form a *Galois connection* between $\wp(G)$ and $\wp(M)$. The maximal sets of objects which are related to the maximal sets of attributes correspond to closed

sets of the composition of both operators $'$, denoted $''$, for $\wp(G)$ and $\wp(M)$ respectively. A pair $(A, B) \in \wp(G) \times \wp(M)$, where $A = B'$ and $B = A'$, is a *formal concept*, A being the *extent* and B being the *intent* of the concept. The set $\mathcal{C}_\mathcal{K}$ of all concepts from $\mathcal{K}$ is ordered by extent inclusion, denoted by $\leq_\mathcal{K}$, i.e. $(A_1, B_1) \leq_\mathcal{K} (A_2, B_2)$ when $A_1 \subseteq A_2$ (or dually $B_2 \subseteq B_1$). Then, $\mathcal{L}_\mathcal{K} = \langle \mathcal{C}_\mathcal{K}, \leq_\mathcal{K} \rangle$ forms the *concept lattice* of $\mathcal{K}$.

Typically, a concept lattice to index documents is created from a formal context $\mathcal{K}_{index} = (G, M, I)$ where G is a set of documents and M is a set of terms. Thus, the set I represents document *annotations* (i.e. gIm indicates that the document g is annotated with the term m). In a nutshell, to retrieve documents given a conjunctive query[3] $q = \{m_i\}, m_i \in M$ (m_i in the query are hereafter referred as *keywords*), the goal is to find those formal concepts (A, B) where $B \sim q$ and to retrieve the documents in A. The usual approach is to insert into the lattice a *query concept* $C_q = (\{\emptyset\}, q)$ [11, 10, 1] the intent of which contains all the keywords in the user query. Different techniques have been proposed to navigate the lattice, however they rely on topological properties (navigating the super-concepts and sub-concepts if C_q) of the concept lattice to search for documents. Although topology-based measures are useful to retrieve *related documents* from a query, there are some drawbacks that could be overcome with the use of semantic similarity.

The first disadvantage with the navigating in the hierarchy of C_q refers to the generalization of the query. By obtaining the super-concepts of the *query concept* inserted in the lattice, a level of granularity already provided by the user is lost. For example, for a query of the form *"complications, arthroscopy"*, a query concept $C_q = (A_q = \{\emptyset\}, B_q = \{complications, arthroscopy\})$ is created within the lattice. Any super-concept $C_{sup} = (A_{sup}, B_{sup})$ of the query concept has to comply with $B_{sup} \subset B_q$. In this case, only three super-concepts can be obtained: $C_{sup1} = (A_{sup1}, \{$*complications*$\})$, $C_{sup2} = (A_{sup2}, \{$*arthroscopy*$\})$ and $C_{sup3} = (A_{sup3}, \{\emptyset\})$. However, A_{sup1} contains documents about *complications* in any aspect leading to a decrease in precision. The same happens with documents in A_{sup2} containing documents about *arthroscopy* in general whether the user had already specified a restriction for them. C_{sup3} represents the supremum where A_{sup3} contains every possible document, this, of course, is the worst case scenario where the system has no restrictions to retrieve documents.

The second disadvantage is about the specification of the query. By obtaining the whole set of sub-concepts of the query concept the system assumes restrictions not provided by the user. While this is the main idea behind *query expansion* [3] the problem is that there are no discrimination with the sub-concepts that should be used to retrieve documents. For example, given the same query used in the last example and the sub-concepts $C_{sub1} = (A_{sub1}, \{complications, arthroscopy, infection\})$ and $C_{sub2} = (A_{sub2}, \{complications, arthroscopy, practice\})$, the system cannot decide whether the documents in A_{sub1} or the documents in A_{sub2} are the most relevant. From a human perspective, it could be assumed that documents in A_{sub1} may be of more interest for the user since an *infection* is a possible *complication* in the context of a surgery such as an *arthroscopy* and hence they should be retrieved first. On the other side, *practice* is a general word which may lead to non-relevant documents.

[3] Keywords and the conjunction operator $\wedge$

Regarding these problems we propose a technique to improve information retrieval based on concept lattices using the idea of "concept similarity" provided by Formica. We combine this idea with a novel heuristic to navigate the lattice in order to find those concepts holding relevant documents for a given query.

3.1 Navigating the lattice

Given the formal context $\mathcal{K}_{index}$ and the query $q = m_i$ at the beginning of this section, a very simplistic approach to retrieve documents relevant to the query is to find those concepts (A_j, B_j) where $m_i \in B_j : \forall m_i \in q$ defined in [13] as *retrieve algorithm*. Actually, it is possible to find a single concept $C_q = (A_q, B_q)$, where B_q holds the minimal set of words containing all keywords. Subsequently, A_q contains the maximal set of documents containing all the keywords. We refer to $C_q = (A_q, B_q)$ as the matching concept.

It should be noted here that, for a given query q, the matching concept C_q may not exist. This is more likely to happen if the number of keywords is high. In a complete concept lattice (not filtered through any means and constructed using the total amount of information), such a case would actually mean that there are no documents which comply with all the restrictions provided in the user's query. While some strategies can be implemented to overcome this issue (asking the user to provide a simpler query or manipulating the query in order to answer) for the scope of this work we do not elaborate on this and we rather consider the case of an existing matching concept.

Once the matching concept $C_q = (A_q, B_q)$ is found, all documents in A_q can be retrieved to the user. Since the number of documents in A_q may be not sufficient, what is important, in the following, is how to complete the answer with more documents using the lattice.

A simple strategy would involve the hierarchy of C_q, however every sub-concept $(A_{sq}, B_{sq}) \leq_{\mathcal{K}} (A_q, B_q)$ will provide no different documents than those in A_q since $A_{sq} \subset A_q$. Super-concepts of C_q are not useful either because of the problems described in the beginning of this section regarding generalization. Hence, in order to complete the answer with more documents, it is necessary to obtain from the concept lattice some formal concepts which are neither super- nor sub-concepts of (A_q, B_q). To achieve this, we use the notion of *cousin concepts* defined as follows.

Definition of cousin concepts: Two concepts (A_1, B_1) and (A_2, B_2) which are not comparable for $\leq_{\mathcal{K}}$ are said to be *cousins* iff there exists $(A_3, B_3) \neq \perp$ such that $(A_3, B_3) \leq_{\mathcal{K}} (A_1, B_1)$ and $(A_3, B_3) \leq_{\mathcal{K}} (A_2, B_2)$ and $d_{\mathcal{K}}((A_2, B_2), (A_3, B_3)) = 1$ (where $\perp$ is the bottom concept and $d_{\mathcal{K}}$ measures the minimal distance between two formal concepts in the lattice $\mathcal{K}$). Intuitively, this means that (A_1, B_1) and (A_2, B_2) do not subsume each other and that (A_3, B_3) can be either the lower bound or be subsumed by the lower bound $(A_1, B_1) \sqcap (A_2, B_2)$ (where $(A_1, B_1) \sqcap (A_2, B_2)$ denotes the lower bound of (A_1, B_1) and (A_2, B_2).

The use of cousin concepts allows us to move in the lattice from one concept to another using the relations that the elements in their intents possess and that are expressed through their common subsumer. In the example on Figure 1, C_2 is a cousin concept of C_1 because of concept C_3. The attributes *"arthroscopy"*, *"complication"* and *"infection"* are all related through the intent of concept C_3. In this small example, if C_1 is

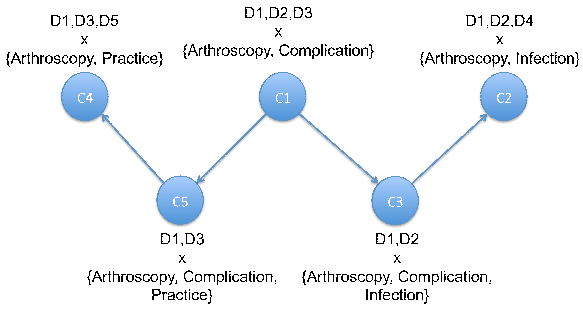

Fig. 1. Example. Five concepts within a lattice, extents and intents are shown. Arrows indicate query *expansion* and *modification*

the matching concept, moving from it to concept C_2 is the same as replacing the word *"complication"* with the word *"infection"* in the query. The extent of concept C_2 will contain documents, some of which are different from those of C_1 and therefore they can be used to complete the answer provided by C_1.

We may also notice that the use of C_3 works as a *query expansion*, adding attributes to the original user query, while the use of C_2 works as a *query modification*, since its attributes are a subset of the attributes of C_3.

Using the entire sub-hierarchy of the matching concept (excluding the infimum) allows us to retrieve several cousin concepts which can be used to complete the answer far beyond the initial set of documents contained in the matching concept's extent. Each cousin concept is a possible *query modification* obtained from a *query expansion*, represented by the sub-concepts of the matching concept.

Although cousin concepts are useful to expand the answer by representing query modifications, their use may entail the same problem described in the beginning of this section, as the second disadvantage of structure-based concept retrieval. In the same example on Figure 1 concepts C_2 and C_4 are cousin concepts of C_1. However, in this scenario the system cannot decide which set of documents, between those of C_2 and C_4, should be retrieved first.

A way to rank cousin concepts is therefore necessary in order to decide which documents should be retrieved to the user first. In this paper we do so using the measure of concept similarity proposed by Formica [5].

3.2 Concept ranking through similarity

The ranking of the retrieved cousin concepts is performed using a semantic similarity metric proposed by Formica[5]. That is, given two formal concepts $C_1 = (A_1, B_1)$ and $C_2 = (A_2, B_2)$ the similarity between them is defined as:

$$sim(C_1, C_2) = \frac{|A_1 \cap A_2|}{max(|A_1|, |A_2|)} * w + \frac{\mathcal{M}(B_1, B_2)}{max(|B_1|, |B_2|)} * (1 - w) \qquad (1)$$

where $0 \leq w \leq 1$ is a weighting parameter and $\mathcal{M}(B_1, B_2)$ is the maximization of the sum of the *information content* similarities between each possible pair of terms created using one term from B_1 and another from B_2. *Information content* similarity between two terms is measured using their distance in a lexical hierarchy and/or their co-occurrence in a text corpus. The full explanation of this metric is beyond the scope of this paper. For further information, the reader is referred to the original work of Formica [5].

Consider the example of Figure 1: Concepts C_2 and C_4 are both cousin concepts of the matching concept C_1, and they have the exact same structural features, i.e. the cardinalities of the intersections of their extents/intents with the matching concept are the same, as well as their extent/intent cardinalities. However, when using the semantic similarity metric defined above with $w = 0.5$ and Wordnet[4] as the external lexical hierarchy, we observe that $sim(C_1, C_2) = 0.7275$, while $sim(C_1, C_4) = 0.45$, because the pair *(complication, infection)* has a higher semantic relation than the pair *(complication, practice)*. In this way, we may rank and retrieve the documents of concept C_2, higher than those of concept C_4. Differentiations in the weight value w allow for differentiations in the preference over the structural (from the extents) and semantic (from the intents) similarities of the compared concepts.

4 Experimental results and discussion

We applied our approach using the MuchMore[5] dataset, which contains annotated medical document abstracts (7822 documents, 9485 single or multi-word terms). In order to answer a given user query we follow a 3-step knowledge discovery process, as follows.

Step 1 - Data preprocessing: Pre-filter the set of documents and terms Since the creation of a lattice containing the full set of documents/terms would be computationally expensive, we create a reduced lattice for each given user query. To do so, we implement a simple pre-filtering strategy of iterative expansion similar to the one described in [4]. Given a conjunctive query $q = \{t_i\}$ we fetch all documents d_n that contain all the keywords in the query. Afterwards, we obtain all the terms t_j that these documents contain. Finally, we fetch all the additional documents d_m which contain any of these terms. At the end we obtain a set of $d_n + d_m$ documents and $t_i + t_j$ terms which is used to create a formal context. For the query $q_s = \{$*"complication","arthroscopy"*$\}$, this process returns a set of 11 initial documents, which in turn leads to an expanded set of 177 terms and 7560 documents.

Unfortunately, this strategy yields more than 95% of the corpus' documents because of highly frequent terms. To avoid this, documents with a number of terms below the average (in the above example, 7 terms) are not included in the expanded set of documents (3485 documents for the example). It should be noted that the pre-filtering strategy can be further improved considering weighting techniques such as tf.idf, pivoted normalized document length [9] or heuristic approaches specifically focusing on the reduction of irrelevant concepts in a FCA lattice [4].

[4] Wordnet is a widely-used free semantic dictionary organized in a hierarchical manner [12]

[5] `http://muchmore.dfki.de/`

Step 2 - Transformation: Concept lattice creation The creation of the concept lattice is straightforward since we rely on a fixed framework (Coron Toolkit[6]). For the example of query q_s we obtain 134718 formal concepts without using support pruning.

Step 3 - Data mining & Evaluation: Retrieving documents from the lattice The retrieval step consists of three sub-steps, described in the following.

1. **Find the matching concept.** We search for the matching concept C_q in the lattice using a level-wise algorithm, starting from the supremum. The matching concept C_q is the closer concept to the supremum which contains in its intent all the keywords provided in the query. The existence of the matching concept is predicated in the assumption of a conjunctive query to pre-filter the dataset and create the formal context. In the case that there are no documents containing at least all the keywords, the query is consider unsuccessful and the retrieval process is stopped at step 1. The documents in the extent of the matching concept are retrieved to the user and they are hereafter referred to as *exact answer*.
2. **Find the cousin concepts of the matching concept.** Cousin concepts are obtained for the matching concept and for each of its sub-concepts C_i. A list called *candidate answers* is created storing the pair (C_i, C_j) where $C_i \leq C_q$ and C_j is a cousin concept of C_i. For q_s, the *candidate answers* list contains 2301 (concept, cousin concepts) pairs.
3. **Rank the cousin concepts.** The ranking process is performed using the similarity measure described in section 3.2. Every pair (concept, cousin concept) from the *candidate answers* list is compared, or what is the same, each *query expansion* is compared to its correspondent *query modification*. Formica's concept similarity was implemented using Wordnet [12] as a lexical hierarchy[7], the *Brown corpus* as a base to obtain term frequencies and a modified version of the Hungarian algorithm [8] to match terms from both intents[8]. The experiments here presented where performed with a value of $w = 0.5$.

Table 2 shows the results for two queries executed using the described approach. For $q_s = \{$*"arthroscopy"*,*"complication"*$\}$ the *exact answer* retrieved 11 documents of which 7 are relevant to the user. The *close answer*, composed of the documents retrieved from the ranked cousin concepts, contains 100 documents of which 6 are relevant to the user. Therefore, out of the 21 documents relevant to the user the approach was able to retrieve 13.

It is of special interest to analyse the characteristics of the obtained results. The cousin concept with intent *joints, surgical aspects, complication, diagnostic* has a similarity of 0.71 with the concept with intent *arthroscopy, surgical aspects, complication, diagnostic* which is a sub-concept of the matching concept created for q_s and hence, does not have additional documents than those already retrieved. What can be appreciated here is that the algorithm works firstly by expanding the original query with related

[6] `http://coron.loria.fr/site/index.php`

[7] Wordnet is a dictionary where terms are grouped by synonymia (synset) and ordered in a hierarchical tree by the hypernym relation.

[8] The Hungarian algorithm minimizes the sum of values in the diagonal of a square matrix.

terms (from *arthroscopy* to *surgical aspects*[9]) and secondly by modifying the expanded query with a semantically similar term (from *arthroscopy* to *joints*). The above process is illustrated in Table 1.

Table 1. Query expansion and modification.

matching concept *query*		sub-concept *expansion*				cousin concept *modification*
arthroscopy complication	→	arthroscopy complication surgical aspects diagnostic	→	~~arthroscopy~~ complication surgical aspects diagnostic	→	joints complication surgical aspects diagnostic

The second query in Table 2 is also of interest in the sense that it indicates algorithm robustness. The word laparoscopic is not present in Wordnet, making it not suitable for the comparison in the similarity measure. This means that *laparoscopic* can be replaced with any other term since the algorithm is not able to measure the difference. However, since the similarity measure relies also in extent intersection, the algorithm will try to replace *laparoscopic* with terms used by documents similar to those in the exact answer. In that way, the first ranked close answer is correct and its intent is *complication, risk, cholecystectomy*. Notice that in this case the algorithm does not conclude that the term *risk* is semantically close to the term *laparoscopic*, but that it is the best term to replace the latter in the query.

Table 2. Results for two queries.

Query	**Exact answer** correct/found	**Close Answer** correct/found	**Total Answers** correct/expected
arthroscopy, complication	7/11	6/100	13/21
complication, laparoscopic cholecystectomy	3/3	3/100	6/7

5 Conclusions

In this paper we present a technique to use a concept lattice for the retrieval of documents from a given user query. The proposed technique differs from previous approaches in two main aspects: the lattice navigation algorithm is not restricted to the

[9] an arthroscopy is a knee surgery

hierarchy of the query concept and the ranking algorithm is based on the semantic similarity, rather than on the structural characteristics of the compared concepts.

In terms of navigation, we introduce the notion of *cousin concepts*, which represents *query modifications* that can be used to retrieve documents different from those directly related to the query. In terms of ranking, we use external knowledge sources (a lexical hierarchy and a text corpus) to measure semantic similarity and order the retrieved *cousin concepts* by relevance to the initial query.

We illustrate our approach using two examples from a dataset of medical document abstracts. We also explain certain limitations of the proposed approach, mainly regarding the performance the concept lattice construction and the availability of the terms of the user query in the dataset. Currently, we are applying this approach on the same dataset but on a full scale, in order to measure precision and recall, as well as to compare the proposed technique with other Information Retrieval state-of-the-art techniques.

References

1. Claudio Carpineto and Giovanni Romano. Order-theoretical ranking. *Journal of the American Society for Information Sciencies*, 51(7):587–601, May 2000.
2. Claudio Carpineto and Giovanni Romano. Using concept lattices for text retrieval and mining. In *Formal Concept Analysis*, pages 161–179. Springer-Verlag, Berlin, Heidelberg, 2005.
3. Claudio Carpineto and Giovanni Romano. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys (CSUR)*, 44(1), January 2012.
4. Claudio Carpineto, Giovanni Romano, and Fondazione Ugo Bordoni. Exploiting the potential of concept lattices for information retrieval with credo. *Journal of Universal Computer Science*, 10:985–1013, 2004.
5. Anna Formica. Concept similarity in formal concept analysis: An information content approach. *Knowledge-Based Systems*, 21(1):80 – 87, 2008.
6. Benrhard Ganter and Rudoph Wille. *Formal Concept Analysis*. Springer, Berlin, 1999.
7. Marti A. Hearst and Jan O. Pedersen. Reexamining the cluster hypothesis: scatter/gather on retrieval results. In *Proceedings of SIGIR 1996*, SIGIR '96, pages 76–84. ACM, 1996.
8. H. W. Kuhn and Bryn Yaw. The hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, pages 83–97, 1955.
9. Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.
10. Nizar Messai, Marie-Dominique Devignes, Amedeo Napoli, and Malika Smaïl-Tabbone. Using domain knowledge to guide lattice-based complex data exploration. In *Proceedings of the 2010 conference on ECAI 2010*, pages 847–852. IOS Press, 2010.
11. Nizar Messai, Marie-Dominique Devignes, Amedeo Napoli, and Malika Smal-Tabbone. Querying a bioinformatic data sources registry with concept lattices. In *Proceedings of ICCS 2005*, pages 323–336. LNCS 3596, Springer, 2005.
12. George A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, November 1995.
13. Ibtissem Nafkha, Samir Elloumi, and Ali Jaoua. Using concept formal analysis for cooperative information retrieval. In *Concept Lattices and their Applications 2004*, volume 110 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2004.
14. Uta Priss. Lattice-based information retrieval. *Knowledge Organization*, 27:132–142, 2000.
15. Uta Priss. Formal concept analysis in information science. *Annual Review of Information Science and Technology.*, 40(1):521–543, December 2006.