Conceptual query expansion

F.A. Grootjen *, Th.P. van der Weide

Faculty of Science, Mathematics and Computer Science, Radboud University Nijmegen,
Toernooiveld 1, Nijmegen 6525 ED, The Netherlands

Received 3 February 2005; received in revised form 3 February 2005; accepted 24 March 2005
Available online 26 April 2005

Abstract

This article presents a new, hybrid approach that projects an initial query result onto global information, yielding a local conceptual overview. Concepts found this way are candidates for query refinement. We show that the resulting conceptual structure after a typical short query of 2 terms, contains refinements that perform just as well as a most accurate query formulation. Subsequently we illustrate that query by navigation is an effective mechanism which in most cases finds the optimal concept in a small number of steps. When an optimal concept is not found, the navigation process still finds an acceptable sub-optimum.

© 2005 Elsevier B.V. All rights reserved.

Keywords: Relevance; Feedback; Query; Expansion

1. Introduction

Formulating queries is not a simple task. Sometimes searchers seem to use a searching strategy where the search engine’s contribution is recall whereas the searcher is concerned with precision (see for example [1–3]). Typically, searchers carefully select a small number (3 or less) of keywords to describe their information need. Such queries are referred to as short queries. For example, in
The extensional approach is based on the materialization of the information need in terms of documents. This query enrichment method may be done with user intervention, for example using relevance feedback, or without user intervention, by a so-called local analysis. In case of relevance feedback, the searcher may be asked to assign relevance judgements to a number of documents. Another approach is to ask the searcher to indicate the highest ranked relevant document.

(2) The intensional approach is based on the meaning of the keywords. Thesauri are used to describe the meaning and relations of terms. By locating the short query within the thesaurus, a reformulation of the query can be obtained. The thesaurus may be based on the underlying collection (if any) or based on world knowledge (Wordnet). In the case of a restricted area of interest, a representative collection may serve as a meaning framework (knowledge base) for that area of interest.

Weather related fatalities.
Document will report a type of weather event which has directly caused at least one fatality in some location.
A relevant document will include the number of people killed and injured by the weather event, as well as reporting the type of weather event and the location of the event.

Fig. 1. tREC query 59.
In the collaborative approach the system tries to employ previous behavior of searchers to obtain a better idea of the intended meaning of a query. See for example [7–9]. A possibility is to introduce a similarity measure between searchers, providing the opportunity to carry over (implicit) relevance judgements.

In this paper we employ a mix between the extensional and the intensional approach. We discuss a hybrid form of query expansion called conceptual query expansion. In this approach, both the local initial query result and the global information from the complete collection are used. Rather than using the conceptual structure resulting from global collection information, we project this structure onto the local initial query result. The resulting concepts may be seen as a structured overview of the various interpretations of the query.

Index expressions have been introduced (see for example [10]) as a compromise between the richness of noun and verb phrases and the simplicity of representation. In our experiments on the TREC Associated Press collection, we use index expressions to describe the contents of documents. Recent research (see [11]) has shown that index expression perform better than word combinations (like n-grams). The main reason is that index expression omit useless word combinations but express word combinations that go beyond word position in the sentence. Besides, index expressions use a connector to clarify the relation between the combined words. This will lead to a better quality of the generated concepts. Index expressions are effectively obtained from a text with a high level of precision by a grammarless parser technique [12].

Our assumption is that the best ranked documents of the initial query result form a proper basis to generate a local thesaurus. Even when a top ranked document is not relevant for the searcher, it will still address the topics around the query. The top-ranked documents thus may be used to map out the meaning structure associated with the query to some reasonable extent. In order to investigate this principle and to get an impression of its potentials, we will restrict ourselves in this paper to the benefits of a single document, the top-ranked document. In our experiments, we will also consider the effect of using the highest ranked relevant document instead.

After generating the local thesaurus by projecting the global thesaurus onto this top-ranked document, we obtain a conceptual structure. Several techniques are possible to balance between granularity of concepts and algorithmic complexity. In this paper we will show that a substructure is sufficient, thus limiting the computational complexity. We argue that each concept in this structure relates to a specific and meaningful query expansion. The problem is to find the best concept, and then use the related query expansion.

An important result of our experiments is that this local thesaurus contains concepts that approximate the full text formulation very well. In the case of relevance feedback it is even superior to the full text formulation (see Section 4). As a consequence, the local thesaurus is a reasonable framework for query expansion.

However, a simple heuristic to locate a (nearly) optimal concept is not obvious. In practice, a searcher may navigate through this local thesaurus (Query by Navigation, see [10]) to locate the best suitable concept. To illustrate the power of this mechanism, we let a simulated searcher perform this process of Query by Navigation. Our experiments show that this searcher will effectively locate the best concept in only a few navigation steps. Usually this searcher takes a straight course. In some cases, however, finding the optimal concepts requires a bit of wandering around. The average navigation path in all cases is limited.
The structure of this paper is as follows. In Section 2 we discuss related work. In Section 3 the proposed model is described. In Section 4 we describe the experiments that show the presence of good quality concepts. In Section 5 the experimental results for query by navigation are presented and discussed. Section 6 contains conclusions and further directions for research.

2. Query expansion techniques

In this section we discuss in some more detail the query expansion techniques that will be used for comparison.

2.1. User relevance feedback

Probably the most popular query reformulation technique is Relevance Feedback. Using an initial retrieval result the user is asked to mark relevant documents in a list of 10 (or 20) ranked documents. Early experiments [13] have shown good improvements in precision for small test collections using relevance feedback. Although relevance feedback is relatively easy to implement, in practice it seems to be very difficult to persuade a searcher to tediously work through a list of documents and marking them relevant. At best, a searcher may be asked to select a relevant document when presented a list of document excerpts. Information like this is easy to collect, since this is exactly what we do when using a search engine like for example Google.

If the initial query result is expected to be significant, one might consider an alternative form of relevance feedback, assuming for example that the top ranked document is relevant. This mechanism is called pseudo relevance feedback.

There are several ways to calculate an improved query. For the vector model, a single relevant document and plain positive feedback strategy (no non-relevant documents selected by user), Rocchio [14] and IDE [15] provide the same formula for the modified $\tilde{q}_m$.

$$\tilde{q}_m = \alpha \tilde{q} + \beta \tilde{d},$$

where $\tilde{q}$ is the original query, and $\tilde{d}$ the relevant document, and $\alpha$ and $\beta$ are tuning parameters. Assuming that both the query and the documents are normalized, taking $\alpha = 1$ and $\beta = 1$ seems to be a reasonable choice.

2.2. Global query expansion

Another way of query expansion is adding words (synonyms) or words that are related to the original query. By doing this the knowledge stored in a thesaurus or other (global) information source is used to increase recall. Thesauri have frequently been incorporated in information retrieval systems as a device for the recognition of synonymous expressions and linguistic entities that are semantically similar but superficially distinct. Note that unlike the previous expansion techniques, global query expansion does not require a relevant document.

Automatic query expansion using thesauri has been the target of research for nearly four decades, and a lot of methods have been proposed. Mandala et al. [16] presents an concise overview of these methods and distinguishes three categories:
• Hand-crafted thesauri,
• Co-occurrence based thesauri,
• Head modifier based thesauri.

Query expansion based on hand-crafted thesauri is only successful if a the thesaurus is domain-specific and corresponds closely to the domain-specific document collection being searched [17]. The use of general purpose hand-crafted thesauri for automatic query expansion has not been very successful [18,19]. Experiments with co-occurrence based thesauri show a gain of 20% in retrieval performance [20] on small test collections, but are less effective on larger collections [21]. More linguistically motivated approaches like head modifier based thesauri show similar results [22,23]. As shown in [16], a combination of query expansion techniques yields better performance than the techniques on their own.

In this paper we show that it is feasible to create local thesauri on-the-fly, tailored to both the query and the collection being searched. Using this method we combine the advantages of local and global query expansion.

3. Conceptual query expansion

Obviously, the words in a (pseudo) relevant document are good candidates for query expansion. Combinations of words can give a more precise indication of meaning, especially when (as in index expressions, [10]) the relation between the words is made explicit. For example, the index expression pollution of rivers in Holland combines the terms pollution, rivers and Holland but also specifies their intended mutual relation. We will use descriptor and term as generic terms for words, word combinations and index expressions. Compound descriptors are combinations of words (such as index expressions).

The question then is: which compound descriptors are the most suitable to expand a query? A rigid indexing schema that covers all compound descriptors is not feasible, even in small collections of small documents (say a couple of hundred document, each containing only a couple of hundreds of words only), many combinations are possible, most of which will be meaningless.

The technique proposed in this article, called Conceptual Query Expansion, uses a special notion to drastically limit the number of possible term combinations: the notion of formal concepts. The key thought is to consider only those combination of terms that make sense in the collection, that is: only consider combinations of terms that form a formal concept.

3.1. Formal concept analysis

Before continuing, we will shortly discuss the elements of Formal Concept Analysis [24,25].

3.1.1. Context

Suppose we have a collection D of documents. Individual members of this collection (documents) are written with small letters like $d, d_1, d_2$, while subsets are written in capitals
During the indexing process, descriptors (attributes) are attached to documents. We write $A$ to denote the set of all attributes, $a, a_1, a_2$ for individual attributes and $A, A_1, A_2$ for attribute sets (subsets of $A$). The result of indexing process is reflected in the binary relation $\sim$: we write $a \sim d$ iff attribute $a$ describes document $d$. The tuple $(D, A, \sim)$ is called a context.

The context relation $\sim$ is overloaded to cover set arguments in the following way:

\[
\begin{align*}
    a \sim D &\equiv \forall d \in D [a \sim d], \\
    A \sim d &\equiv \forall a \in A [a \sim d], \\
    A \sim D &\equiv \forall a \in A, d \in D [a \sim d].
\end{align*}
\]

3.1.2. Properties of contexts

Using the context relation a classification of documents and attributes can be generated such that each class can be seen as a concept in terms of properties of the associated documents and attributes. In our interpretation, documents and attributes assign meaning to each other via the context relation: within the limits of this view, we can not distinguish between document with identical properties, while attributes having the same extensionality are assumed to be identical. Sharing document meaning thus can be seen as sharing attributes:

**Definition 1.** The common attributes of a set of documents are found by the right polar function $\text{ComAttr}: \mathcal{P}(D) \to \mathcal{P}(A)$ defined as follows:

\[
\text{ComAttr}(D) = \{ a \in A | a \sim D \}.
\]

Documents may also be shared by attributes:

**Definition 2.** The documents sharing properties are captured by the left polar function $\text{ComDocs}: \mathcal{P}(A) \to \mathcal{P}(D)$ defined by:

\[
\text{ComDocs}(A) = \{ d \in D | A \sim d \}.
\]

3.1.3. Concepts

A special situation is when the duality of meaning between a set $D$ of documents and a set $A$ of attributes is symmetric: $A \sim D$. It is easily verified that $D \subseteq \text{ComDocs}(A)$ and $A \subseteq \text{ComAttr}(D)$. If the sets $D$ and $A$ are maximal, then this combination is referred to as a concept:

**Definition 3.** A concept is a pair $(D, A) \in \mathcal{P}(D) \times \mathcal{P}(A)$ such that $D$ and $A$ are their mutual meaning:

\[
\begin{align*}
    \text{ComAttr}(D) &= A, \\
    \text{ComDocs}(A) &= D.
\end{align*}
\]

Obviously not every set of documents (attributes) forms a 2concept. But when it does, at most one concept can be associated with it. So a concept is uniquely identified by its set of documents or by its set of attributes.
Definition 4. Let $c = (D, A)$ be a concept, we will write $\delta(c)$ to denote its extensionality $D$ and $\alpha(c)$ for its intension $A$.

3.1.4. The concept lattice
Let $C$ be the set of all concepts that can be derived from the set of documents $D$ and the set of attributes $A$ and their relation $\sim$. These concepts are ordered in the following way:

Definition 5. A concept $c_1$ is more specific than concept $c_2$ if it has a restricted extensional meaning:

$$c_1 \subseteq c_2 \equiv \delta(c_1) \subseteq \delta(c_2).$$

Having a restricted extensional meaning is equivalent with having an augmented intensional meaning: $c_1 \subseteq c_2 \iff \alpha(c_1) \subseteq \alpha(c_2)$. The fact that $(C, \subseteq)$ is a partial order follows directly from the fact that $(P(D))$ is a partial order.

Let $C$ be a set of concepts. A lowerbound of $C$ is a common subconcept. If there exists a greatest element in the set of lower bounds of $C$, then this element is called greatest lowerbound, and denoted as $\wedge(C)$. Likewise the smallest element in the set of upperbounds is called smallest upperbound, denoted as $\vee(C)$.

It can be proven that for each set of concepts $C$:

$$\delta(\wedge(C)) = \bigcap_{c \in C} \delta(c) \quad \alpha(\wedge(C)) = \text{ComAttr}(\delta(\wedge(C))),$$

$$\alpha(\vee(C)) = \bigcap_{c \in C} \alpha(c) \quad \delta(\vee(C)) = \text{ComDocs}(\alpha(\vee(C))).$$

The proof are straight forward, see for example [26,27] or [28].

Lemma 1
(1) $D_1 \subseteq D_2 \Rightarrow \text{ComAttr}(D_1) \subseteq \text{ComAttr}(D_2)$,
(2) $A_1 \subseteq A_2 \Rightarrow \text{ComDocs}(A_1) \subseteq \text{ComDocs}(A_2)$.

Proof. As the second property is the dual version of the first property, we will only prove the first one. Let $D_1 \subseteq D_2$, and suppose $a \sim D_2$. As $D_1 \subseteq D_2$, we conclude $a \sim D_1$, and thus $\text{ComAttr}(D_1) \subseteq \text{ComAttr}(D_2)$. $\square$

Mutual sharing of meaning between documents and attributes is a special case. First we provide a better characterization of this situation. In the next section, mutual sharing of meaning will be the basis for the introduction of concepts.

Lemma 2. $A \sim D \iff D \subseteq \text{ComDocs}(A) \land A \subseteq \text{ComAttr}(D)$.

Proof. First assume $A \sim D$, and let $d \in D$. Then $A \sim d$ and thus $d \in \text{ComDocs}(A)$. Analogously we conclude $A \subseteq \text{ComAttr}(D)$. 

Next assume $D \subseteq \text{ComDocs}(A) \land A \subseteq \text{ComAttr}(D)$, and let $a \in A$. Then also $a \in \text{ComAttr}(D)$, and this $a \sim D$. Analogously we conclude $d \in D \Rightarrow A \sim d$. As a consequence $A \sim D$.  

It will be useful to decompose the polar functions in terms of elementary set operations, showing how these operations distribute over the polar functions. These properties motivate operations on concepts in the next section.

**Lemma 3**

(1) $\text{ComAttr}(D_1 \cup D_2) = \text{ComAttr}(D_1) \cap \text{ComAttr}(D_2)$,

(2) $\text{ComDocs}(A_1 \cup A_2) = \text{ComDocs}(A_1) \cap \text{ComDocs}(A_2)$.

**Proof.** The expression $a \in \text{ComAttr}(D_1 \cup D_2)$ is equivalent with $a \sim D_1 \cup D_2$, which is equivalent with $a \sim D_1$ and $a \sim D_2$.  

The compositions of sharing attributes and sharing documents are next under consideration. For each set $D$ of documents we can find its minimal extension $\text{DocsClass}(D)$ that is closed under common attributes. We make the analogous observation for sets $A$ of attributes.

**Definition 6**

(1) $\text{DocsClass}(D) = \text{ComDocs}(\text{ComAttr}(D))$,

(2) $\text{AttrClass}(A) = \text{ComAttr}(\text{ComDocs}(A))$.

In order to motivate this definition, we first remark that both document class and attribute class are extensions of their argument set.

**Lemma 4**

(1) $D \subseteq \text{DocsClass}(D)$,

(2) $A \subseteq \text{AttrClass}(A)$.

**Proof.** We will only prove the first statement, the second can be proven analogously. From Lemma 3 we conclude $\text{ComAttr}(D) \sim D$, applying Lemma 2 yields the result.  

The minimality of these extensions is a direct consequence of the following lemma.

**Lemma 5**

(1) $A \sim D \Rightarrow \text{DocsClass}(D) \subseteq \text{ComDocs}(A)$,

(2) $A \sim D \Rightarrow \text{AttrClass}(A) \subseteq \text{ComAttr}(D)$.

**Proof.** We will only prove the first statement, the second is proven analogously. Assume $A \sim D$, then from Lemma 2 we conclude $A \subseteq \text{ComAttr}(D)$. Applying Lemma 1.2 then yields $\text{DocsClass}(D) \subseteq \text{ComDocs}(A)$.  

As a consequence subsequent application of the sharing operations quickly lead to fixed points:

**Lemma 6** *(Maximal Sharing)*

(1) $\text{ComDocs}(\text{ComAttr}(\text{ComDocs}(A))) = \text{ComDocs}(A)$,

(2) $\text{ComAttr}(\text{ComDocs}(\text{ComAttr}(D))) = \text{ComAttr}(D)$.
Proof. We will only prove the first statement, the second is proven analogously. From Lemma 4.2 we conclude $A \subseteq \text{AttrClass}(A)$, and thus by Lemma 1 we get: $\text{ComDocs}(A) \subseteq \text{ComDocs(AttrClass}(A))$.

On the other hand, using Lemma 4.1, substituting $D$ by $\text{ComDocs}(A)$, we get: $\text{ComDocs}(A) \subseteq \text{DocsClass} (\text{ComDocs}(A))$.

As a consequence:

\[
\text{ComDocs}(\text{ComAttr}(\text{ComDocs}(A))) = \text{ComDocs}(A).
\]

\[\square\]

Lemma 7. Let $A$ be a set of attributes, then the pair $(\text{ComDocs}(A), \text{AttrClass}(A))$ is a concept. This concept is referred to as $\text{Concept}(A)$.

Lemma 8. Let $c_1$ and $c_2$ be concepts, then:

\[
\text{Concept}(\delta(c_1) \cup \delta(c_2)) = \text{Concept}(\exists(c_1) \cap \exists(c_2)),
\]

\[
\text{Concept}(\exists(c_1) \cup \exists(c_2)) = \text{Concept}(\delta(c_1) \cap \delta(c_2)).
\]

Proof. As usual, we only prove the first statement. The set of attributes associated with $\text{Concept}(\delta(c_1) \cup \delta(c_2))$ equals: $\exists(\text{Concept}(\delta(c_1) \cup \delta(c_2))) = \text{ComAttr}(\delta(c_1) \cup \delta(c_2)) = \text{ComAttr}(\delta(c_1)) \cap \text{ComAttr}(\delta(c_2))$ Concept $\text{Concept}(\exists(c_1) \cap \exists(c_2))$ has the same attributes. Consequently these are the same concepts. \[\square\]

As (in our case) each set of concepts has a unique lower and upper bound, the resulting lattice $(\mathcal{C}, \subseteq)$ is a complete lattice. This property is important when generating concept lattices as we will see in Section 4.5.

4. Evaluating conceptual query expansion

In order to test the expressiveness of the concept lattice, we investigate the quality of the concepts as possible query expansions bridging the gap between the lower bound and the upper bound. Our intention is to show that the concept lattice contains concepts that approximate the full text query reasonably well.

To test the different query expansion techniques we ran a number of experiments on the Associated Press collection used in TREC competitions. This collection is approximately 800 Mb big, contains 250,000 documents and is accompanied by 50 queries with their relevance judgements. It consists of more than 100,000,000 words of which were 300,000 unique. All tests were done by BRIGHT, a SMART like vector model based tool using tf-idf document weighting.

BRIGHT uses linguistic stemming (lemmatizing the words to their base form) without stop-word removal. The indexer is capable of generating both single word descriptors as well as index expressions (with length 2). This improves retrieval results (see [11]). Index expressions go beyond linguistic head-noun modifiers (see [29,30]). Furthermore, BRIGHT contains a concept lattice builder.
The retrieval results are measured on recall levels 0.0, 0.1, …, 1.0 and averaged over all queries. For each run we calculate the Mean Average Precision.

Our experiments are based on queries 51–100 (a special case is query 65, as it has no relevant document in the AP collection). In the next subsections we will describe the various elements of our experiments. The results are summarized in Table 1. Note that experiments 6.1 and 6.2 are discussed in Section 5.

4.1. The full query retrieval experiment

For each selected query the retrieval result is determined on the original full text query. This run will probably yield the best retrieval results since all information in the original query is used.

The results of this retrieval experiment will be used as a baseline for comparison with the query expansion runs (see Table 1).

4.2. The two-word query experiment

Since full text queries are not so common, we manually produced for each query a two-word query alternative (see Table 2). The selection of two keywords is straightforward for most queries. The difficulty of describing the information need in two words becomes apparent while performing this exercise. Although the choice for some of keywords seems to be arbitrary, it is the goal of the experiment to see what happens in different situations.

The retrieval performance of this sophisticated searcher using two-word queries is summarized in Table 1. In Fig. 2 lowerbound and upperbound are presented in a recall-precision graph.

4.3. Global query expansion

In this sub-experiment the two-word query is expanded with global information. We use Wordnet [31] to expand the query in two ways:

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Type</th>
<th>Mean average precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>full</td>
<td>0.1331 (100%)</td>
</tr>
<tr>
<td>2</td>
<td>two-word</td>
<td>0.0726 (55%)</td>
</tr>
<tr>
<td>3.1</td>
<td>wordnet1</td>
<td>0.0551 (41%)</td>
</tr>
<tr>
<td>3.2</td>
<td>wordnet2</td>
<td>0.0490 (37%)</td>
</tr>
<tr>
<td>4.1</td>
<td>rftop</td>
<td>0.0761 (57%)</td>
</tr>
<tr>
<td>4.2</td>
<td>rfrel</td>
<td>0.1140 (86%)</td>
</tr>
<tr>
<td>5.1</td>
<td>cqetop</td>
<td>0.1085 (82%)</td>
</tr>
<tr>
<td>5.2</td>
<td>cqerel</td>
<td>0.2628 (197%)</td>
</tr>
<tr>
<td>6.1</td>
<td>navtop</td>
<td>0.0977 (73%)</td>
</tr>
<tr>
<td>6.2</td>
<td>navrel</td>
<td>0.2132 (160%)</td>
</tr>
</tbody>
</table>
Adding all synonyms (words from the same Wordnet-sense),

Adding all synonyms and related words.

The effects on the retrieval performance for global query expansion using Wordnet synonyms (wordnet1) and both synonyms and related words (wordnet2) are shown in Table 1.

As can be seen in Fig. 3 the results for global query expansion are poor. This is consistent with other research (for a detailed study why Wordnet expansion is not working see [32]). Still, the experiment is included to make the contrast between feedback mechanisms on the same document collection more explicit.

<table>
<thead>
<tr>
<th>#</th>
<th>Query</th>
<th>#</th>
<th>Query</th>
<th>#</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>Airbus, government</td>
<td>52</td>
<td>Africa, sanction</td>
<td>53</td>
<td>Leveraged, buyout</td>
</tr>
<tr>
<td>54</td>
<td>Satellite, launch</td>
<td>55</td>
<td>Insider, profit</td>
<td>56</td>
<td>Prime, rate</td>
</tr>
<tr>
<td>57</td>
<td>MCI, financial</td>
<td>58</td>
<td>Rail, strike</td>
<td>59</td>
<td>Weather, fatality</td>
</tr>
<tr>
<td>60</td>
<td>Merit, seniority</td>
<td>61</td>
<td>Israel, affair</td>
<td>62</td>
<td>Military, coup</td>
</tr>
<tr>
<td>63</td>
<td>Machine, translation</td>
<td>64</td>
<td>Hostage, political</td>
<td>66</td>
<td>Language, processing</td>
</tr>
<tr>
<td>67</td>
<td>Disturbance, political</td>
<td>68</td>
<td>Fiber, hazard</td>
<td>69</td>
<td>SALT, revive</td>
</tr>
<tr>
<td>70</td>
<td>Surrogate, motherhood</td>
<td>71</td>
<td>Border, incursion</td>
<td>72</td>
<td>US, movement</td>
</tr>
<tr>
<td>73</td>
<td>Country, movement</td>
<td>74</td>
<td>Conflicting, policy</td>
<td>75</td>
<td>Automation, cost</td>
</tr>
<tr>
<td>76</td>
<td>Constitution, intent</td>
<td>77</td>
<td>Poaching, wildlife</td>
<td>78</td>
<td>Greenpeace, protest</td>
</tr>
<tr>
<td>79</td>
<td>German, party</td>
<td>80</td>
<td>Platform, president</td>
<td>81</td>
<td>Scandal, broadcaster</td>
</tr>
<tr>
<td>82</td>
<td>Genetic, engineering</td>
<td>83</td>
<td>Protect, atmosphere</td>
<td>84</td>
<td>Alternative, energy</td>
</tr>
<tr>
<td>85</td>
<td>Corrupt, official</td>
<td>86</td>
<td>Bank, failure</td>
<td>87</td>
<td>Criminal, officer</td>
</tr>
<tr>
<td>88</td>
<td>Oil, price</td>
<td>90</td>
<td>Oil, gas</td>
<td>91</td>
<td>Advanced, weapon</td>
</tr>
<tr>
<td>92</td>
<td>Military, sale</td>
<td>93</td>
<td>Nra, backing</td>
<td>94</td>
<td>Computer, crime</td>
</tr>
<tr>
<td>95</td>
<td>Computer, detective</td>
<td>96</td>
<td>Computer, medical</td>
<td>97</td>
<td>Fiber, application</td>
</tr>
<tr>
<td>98</td>
<td>Fiber, manufacturer</td>
<td>99</td>
<td>Iran, affair</td>
<td>100</td>
<td>Technology, transfer</td>
</tr>
</tbody>
</table>

Fig. 2. Comparing lowerbound and upperbound.

(1) Adding all synonyms (words from the same Wordnet-sense),

(2) Adding all synonyms and related words.

The effects on the retrieval performance for global query expansion using Wordnet synonyms (wordnet1) and both synonyms and related words (wordnet2) are shown in Table 1.

As can be seen in Fig. 3 the results for global query expansion are poor. This is consistent with other research (for a detailed study why Wordnet expansion is not working see [32]). Still, the experiment is included to make the contrast between feedback mechanisms on the same document collection more explicit.
4.4. Relevance feedback

This experiment uses classical user relevance feedback to expand the query. The query is expanded according to the Rocchio method in two ways:

1. Assuming the top ranked document is relevant (pseudo relevance feedback),
2. Using the top relevant document from the retrieval result (user relevance feedback).

The outcome is summarized in Table 1, and displayed in Fig. 4.

4.5. Conceptual query expansion

The use of Formal Concept Analysis for Information Retrieval purposes looks appealing, but due to the size of collections a straightforward calculation of the entire lattice is not feasible. Even
with the fastest algorithms known today, calculating the lattice for the Associated Press context, spanning 250,000 documents and 4,000,000 attributes is out of the question. However, we will show, that there is no need to calculate the entire lattice: it is possible to calculate a sublattice containing the relevant concepts with respect to the current query.

Suppose that the initial query produced a relevant (or pseudo relevant) document. The terms in this document are probable candidates for query expansion. By calculating the sublattice generated from these terms as attributes, we find concepts (that is combinations of terms) that have a conceptual meaning in terms of the collection. The calculation of this sublattice is easy, and may be split into two steps:

1. **Base concepts**: For each attribute \( a \) calculate the concept \( \text{ComDocs}(\text{ComAttr}([a]),[a]) \).
2. **Compound concepts**: Take two concepts \( c_1 \) and \( c_2 \), and generate their join \( c_1 \lor c_2 \).
3. Repeat step 2 until more new concepts can be generated.

The concept lattice generated by the attributes of a document from the Associated Press collection typically contains a few hundred concepts. In this experiment, we evaluate all concepts in the lattice and use the concept with the best 11 point average precision/recall value.

Table 3
Example expansions

<table>
<thead>
<tr>
<th>Query</th>
<th>Original query</th>
<th>Top relevant</th>
<th>Top ranked</th>
</tr>
</thead>
<tbody>
<tr>
<td>q51</td>
<td>Airbus, government</td>
<td>Germany, Spain</td>
<td>Germany</td>
</tr>
<tr>
<td>q53</td>
<td>Leveraged, buyout</td>
<td>Repay</td>
<td>Use</td>
</tr>
<tr>
<td>q54</td>
<td>Satellite, launch</td>
<td>Commercial, the company, the department, government, license, launch license, rocket, transportation</td>
<td>Rocket</td>
</tr>
<tr>
<td>q58</td>
<td>Rail, strike</td>
<td>Commuter</td>
<td>Commuter</td>
</tr>
<tr>
<td>q59</td>
<td>Weather, fatality</td>
<td>Central, destroy, flood, home, kill, more, people, province, the storm</td>
<td>–</td>
</tr>
<tr>
<td>q72</td>
<td>Movement, US</td>
<td>Bureau, census bureau</td>
<td>Edge</td>
</tr>
<tr>
<td>q73</td>
<td>Movement, country</td>
<td>Emigration</td>
<td>Here</td>
</tr>
</tbody>
</table>

Fig. 5. Conceptual query expansion.
Table 3 shows some example expansions generated for both top ranked as top relevant documents. The power of conceptual query expansion is illustrated by the fact that even non-relevant documents may lead to good expansion: both the relevant document ap900914-0105 as the non-relevant document ap890203-0058 expand rail and strike to commuter for query 58. The difference in expansion for query 72 and 73 is also remarkable. The results of the conceptual query expansion are presented in Table 1 and in Fig. 5.

5. Finding an optimal concept

From the previous section we know that the generated concept lattice contains concepts that can be used to create appropriate query expansion terms. The results show that both in the case of selecting the top ranked and top relevant document the lattice contains such optimal concepts.

A good heuristic to find an optimal concept is not obvious. This section discusses how to find good concepts in the lattice by navigation.

Fig. 6 shows (for query 51) the distribution of the concepts according to their Mean Average Precision they would produce if they were used for query expansion. It is clear that some concepts degrade retrieval performance, while others improve it. The question is, how do we find the best concept?

5.1. The navigation heuristic

Concept lattices are structured; the concepts in the lattice form a partial order. This partial order can be used for navigation: a user may select a subconcept and navigate down in the lattice making his query more specific by adding terms, or losing terms by navigating up to a superconcept and making the query more general. We will call the current concept the user’s focus. The process of navigating down is called refinement, and navigating up is called enlargement.

We will illustrate the navigation process for query 59 (see Fig. 7). Navigation starts at the top node (empty expansion). The user selects flood as expansion candidate. By doing so the mean

![Fig. 6. Concept distribution for query 51.](image-url)
average precision rises from 0.0231 to 0.0725. Subsequently the next step adds both the terms central and storm, with the accompanying score of 0.1088. Finally the end concept is reached by adding (in one single step) the terms destroy, home, kill, more, people, province and the storm, good for an mean average precision of 0.1867.

In the next section we will we try to make it plausible that a searcher is able to efficiently find a good query expansion concept by navigating the concept lattice in only a few steps.

5.2. Simulating query by navigation

In order to simulate a searcher, we wrote a simple program program called autonav that is capable of navigating the concept lattice. The program starts in the top concept of the lattice (the least specific concept) and iteratively chooses the best sub or super concept (in terms of the mean average precision) and changes its focus to that new concept. The navigation process ends if there is no neighboring concept with a better score than the concept in focus.

The results of the simulated navigation for the selected top ranked document and top relevant document are presented in Table 4. For top ranked document selection, the average number of navigation steps is \( \approx 1 \), the average approximation of the best concept is 90\%. For top relevant document selection the average number of navigation steps is somewhat higher (\( \approx 2 \)). The average approximation of the best concept is 82\%.

Fig. 8 shows the results together with standard relevance feedback. From the figure it is clear that:

- Conceptual relevance feedback delivers a significant gain in retrieval performance, even if the top document is not relevant.
<table>
<thead>
<tr>
<th>Query</th>
<th>Steps</th>
<th>Score</th>
<th>Approx. (%)</th>
<th>Query</th>
<th>Steps</th>
<th>Score</th>
<th>Approx. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>q51</td>
<td>1</td>
<td>0.1616</td>
<td>82</td>
<td>q51</td>
<td>2</td>
<td>0.2560</td>
<td>100</td>
</tr>
<tr>
<td>q52</td>
<td>1</td>
<td>0.2889</td>
<td>100</td>
<td>q52</td>
<td>1</td>
<td>0.2925</td>
<td>100</td>
</tr>
<tr>
<td>q53</td>
<td>1</td>
<td>0.2503</td>
<td>86</td>
<td>q53</td>
<td>1</td>
<td>0.2521</td>
<td>86</td>
</tr>
<tr>
<td>q54</td>
<td>2</td>
<td>0.1982</td>
<td>100</td>
<td>q54</td>
<td>1</td>
<td>0.2671</td>
<td>79</td>
</tr>
<tr>
<td>q55</td>
<td>3</td>
<td>0.2462</td>
<td>100</td>
<td>q55</td>
<td>1</td>
<td>0.2123</td>
<td>91</td>
</tr>
<tr>
<td>q56</td>
<td>0</td>
<td>0.1288</td>
<td>100</td>
<td>q56</td>
<td>1</td>
<td>0.3739</td>
<td>100</td>
</tr>
<tr>
<td>q57</td>
<td>1</td>
<td>0.1616</td>
<td>96</td>
<td>q57</td>
<td>2</td>
<td>0.1966</td>
<td>85</td>
</tr>
<tr>
<td>q58</td>
<td>1</td>
<td>0.2209</td>
<td>89</td>
<td>q58</td>
<td>1</td>
<td>0.2277</td>
<td>100</td>
</tr>
<tr>
<td>q59</td>
<td>0</td>
<td>0.0213</td>
<td>100</td>
<td>q59</td>
<td>3</td>
<td>0.1867</td>
<td>100</td>
</tr>
<tr>
<td>q60</td>
<td>1</td>
<td>0.0008</td>
<td>100</td>
<td>q60</td>
<td>4</td>
<td>0.2118</td>
<td>100</td>
</tr>
<tr>
<td>q61</td>
<td>1</td>
<td>0.0124</td>
<td>67</td>
<td>q61</td>
<td>1</td>
<td>0.0941</td>
<td>28</td>
</tr>
<tr>
<td>q62</td>
<td>1</td>
<td>0.1013</td>
<td>84</td>
<td>q62</td>
<td>3</td>
<td>0.1334</td>
<td>98</td>
</tr>
<tr>
<td>q63</td>
<td>1</td>
<td>0.0687</td>
<td>100</td>
<td>q63</td>
<td>2</td>
<td>0.2528</td>
<td>87</td>
</tr>
<tr>
<td>q64</td>
<td>1</td>
<td>0.1065</td>
<td>63</td>
<td>q64</td>
<td>1</td>
<td>0.1140</td>
<td>69</td>
</tr>
<tr>
<td>q66</td>
<td>1</td>
<td>0.0005</td>
<td>100</td>
<td>q66</td>
<td>1</td>
<td>0.5030</td>
<td>50</td>
</tr>
<tr>
<td>q67</td>
<td>1</td>
<td>0.0180</td>
<td>88</td>
<td>q67</td>
<td>3</td>
<td>0.2021</td>
<td>100</td>
</tr>
<tr>
<td>q68</td>
<td>2</td>
<td>0.2055</td>
<td>100</td>
<td>q68</td>
<td>2</td>
<td>0.2196</td>
<td>100</td>
</tr>
<tr>
<td>q69</td>
<td>1</td>
<td>0.0036</td>
<td>100</td>
<td>q69</td>
<td>5</td>
<td>0.9174</td>
<td>100</td>
</tr>
<tr>
<td>q70</td>
<td>1</td>
<td>0.2904</td>
<td>100</td>
<td>q70</td>
<td>7</td>
<td>0.7424</td>
<td>100</td>
</tr>
<tr>
<td>q71</td>
<td>1</td>
<td>0.2500</td>
<td>100</td>
<td>q71</td>
<td>1</td>
<td>0.2500</td>
<td>100</td>
</tr>
<tr>
<td>q72</td>
<td>1</td>
<td>0.0005</td>
<td>100</td>
<td>q72</td>
<td>3</td>
<td>0.0863</td>
<td>100</td>
</tr>
<tr>
<td>q73</td>
<td>1</td>
<td>0.0014</td>
<td>100</td>
<td>q73</td>
<td>1</td>
<td>0.0401</td>
<td>67</td>
</tr>
<tr>
<td>q74</td>
<td>1</td>
<td>0.0036</td>
<td>100</td>
<td>q74</td>
<td>1</td>
<td>0.0116</td>
<td>88</td>
</tr>
<tr>
<td>q75</td>
<td>0</td>
<td>0.0299</td>
<td>100</td>
<td>q75</td>
<td>1</td>
<td>0.1063</td>
<td>91</td>
</tr>
<tr>
<td>q76</td>
<td>1</td>
<td>0.0101</td>
<td>100</td>
<td>q76</td>
<td>1</td>
<td>0.0409</td>
<td>50</td>
</tr>
<tr>
<td>q77</td>
<td>1</td>
<td>0.0303</td>
<td>65</td>
<td>q77</td>
<td>1</td>
<td>0.3030</td>
<td>65</td>
</tr>
<tr>
<td>q78</td>
<td>0</td>
<td>0.4711</td>
<td>90</td>
<td>q78</td>
<td>0</td>
<td>0.4711</td>
<td>90</td>
</tr>
<tr>
<td>q79</td>
<td>0</td>
<td>0.0146</td>
<td>100</td>
<td>q79</td>
<td>1</td>
<td>0.0344</td>
<td>53</td>
</tr>
<tr>
<td>q80</td>
<td>1</td>
<td>0.0238</td>
<td>66</td>
<td>q80</td>
<td>1</td>
<td>0.0471</td>
<td>30</td>
</tr>
<tr>
<td>q81</td>
<td>1</td>
<td>0.0101</td>
<td>100</td>
<td>q81</td>
<td>2</td>
<td>0.1018</td>
<td>100</td>
</tr>
<tr>
<td>q82</td>
<td>1</td>
<td>0.1852</td>
<td>100</td>
<td>q82</td>
<td>2</td>
<td>0.4364</td>
<td>100</td>
</tr>
<tr>
<td>q83</td>
<td>0</td>
<td>0.0326</td>
<td>98</td>
<td>q83</td>
<td>2</td>
<td>0.1701</td>
<td>80</td>
</tr>
<tr>
<td>q84</td>
<td>2</td>
<td>0.0369</td>
<td>88</td>
<td>q84</td>
<td>1</td>
<td>0.0716</td>
<td>66</td>
</tr>
<tr>
<td>q85</td>
<td>1</td>
<td>0.0270</td>
<td>21</td>
<td>q85</td>
<td>2</td>
<td>0.1296</td>
<td>100</td>
</tr>
<tr>
<td>q86</td>
<td>1</td>
<td>0.0309</td>
<td>100</td>
<td>q86</td>
<td>1</td>
<td>0.1483</td>
<td>28</td>
</tr>
<tr>
<td>q87</td>
<td>1</td>
<td>0.0016</td>
<td>89</td>
<td>q87</td>
<td>5</td>
<td>0.2299</td>
<td>100</td>
</tr>
<tr>
<td>q88</td>
<td>0</td>
<td>0.0073</td>
<td>86</td>
<td>q88</td>
<td>2</td>
<td>0.1317</td>
<td>100</td>
</tr>
<tr>
<td>q90</td>
<td>1</td>
<td>0.0424</td>
<td>60</td>
<td>q90</td>
<td>1</td>
<td>0.1133</td>
<td>70</td>
</tr>
<tr>
<td>q91</td>
<td>1</td>
<td>0.0001</td>
<td>100</td>
<td>q91</td>
<td>2</td>
<td>0.2411</td>
<td>88</td>
</tr>
<tr>
<td>q92</td>
<td>1</td>
<td>0.0226</td>
<td>100</td>
<td>q92</td>
<td>4</td>
<td>0.0999</td>
<td>100</td>
</tr>
<tr>
<td>q93</td>
<td>1</td>
<td>0.3367</td>
<td>69</td>
<td>q93</td>
<td>1</td>
<td>0.3400</td>
<td>69</td>
</tr>
<tr>
<td>q94</td>
<td>0</td>
<td>0.0307</td>
<td>100</td>
<td>q94</td>
<td>1</td>
<td>0.0586</td>
<td>80</td>
</tr>
<tr>
<td>q95</td>
<td>0</td>
<td>0.0028</td>
<td>100</td>
<td>q95</td>
<td>1</td>
<td>0.0342</td>
<td>64</td>
</tr>
<tr>
<td>q96</td>
<td>1</td>
<td>0.0034</td>
<td>100</td>
<td>q96</td>
<td>1</td>
<td>0.0888</td>
<td>83</td>
</tr>
<tr>
<td>q97</td>
<td>1</td>
<td>0.0144</td>
<td>50</td>
<td>q97</td>
<td>1</td>
<td>0.1111</td>
<td>44</td>
</tr>
<tr>
<td>q98</td>
<td>1</td>
<td>0.0289</td>
<td>96</td>
<td>q98</td>
<td>3</td>
<td>0.2674</td>
<td>100</td>
</tr>
<tr>
<td>q99</td>
<td>1</td>
<td>0.0371</td>
<td>100</td>
<td>q99</td>
<td>3</td>
<td>0.3555</td>
<td>100</td>
</tr>
<tr>
<td>q100</td>
<td>1</td>
<td>0.0312</td>
<td>96</td>
<td>q100</td>
<td>1</td>
<td>0.0570</td>
<td>45</td>
</tr>
</tbody>
</table>
Since it is not guaranteed that the best concept is found during navigation, the performance is somewhat lower, but still significantly better than without feedback or standard relevance feedback (Fig. 9).

6. Conclusions

In this paper we discussed a way to overcome the inherent shortcomings of short queries, and discussed its potential effectiveness. We showed that it is even possible to benefit from non-relevant documents. We proposed a mechanism to help searchers finding their way in the semantical richness of the meaning of a short query by exploiting latent knowledge stored in the collection.
A possible direction for further research is to incorporate collaborative aspects, or, previous behavior of searchers. Using formal concept analysis, searcher classes can be derived from the behavior log. This may lead to heuristics for query reformulation without searcher interaction.

Another direction can be to investigate ‘reference lattices’. A reference lattice is constructed off-line from a sufficiently rich collection concerning some specific topics. This lattice then can be used to support a searcher during query formulation within that domain.

Acknowledgements

The authors are grateful to the anonymous referees for their valuable suggestions.

References


Franc Grootjen received his master degree at the Catholic University in Nijmegen in 1992 and his Ph.D. in Mathematics, Physics and Computing Science at 26 January 2005 at the Radboud University in Nijmegen. He is currently associate professor at the Radboud University. His main interest is the broad application of mathematical and informatical techniques in a linguistic environment such as Information Retrieval.
Theo van der Weide received his masters degree at the Technical University Eindhoven, The Netherlands in 1975, and the degree of Ph.D. in Mathematics and Physics from the University of Leiden, The Netherlands in 1980. He is currently professor in the Nijmegen Institute for Computing and Information Sciences at the Radboud University in Nijmegen, the Netherlands, and head of the department IRIS (Information and Knowledge Systems). The IRIS department focusses its research on Information and Knowledge Systems. The group contributes to the educational activities of the Informatics program and the Information Science program, both on bachelor and master level. His main research interests include information systems, information retrieval, hypertext and knowledge based systems. Natural language based information modelling in the context of information-system development, including the modelling of the temporal evolution of information objects is another strand of his research.