

On the discovery of the semantic context of queries by game-playing*

Alessandro Agostini
ITC-irst Trento

Paolo Avesani
ITC-irst Trento

Abstract

To model query answering, a question arises out of how the meaning of an user's query is functional to get a valuable answer. In this paper, (1) we investigate the question within an existing peer-to-peer architecture for knowledge exchange—called KEx, (2) we extend the query answering functionality of KEx by a co-evolutive process based on the user's preference information, (3) we model query answering as a language game.

Keywords: formal modeling of query answering, search, self-organization, language games, coordination.

1 Introduction

Responses to queries may not contain the information the user really wants. If we are to build information systems that meet users' expectations, it is necessary to determinate what it means for an answer to be appropriate. To this aim, Grice's maxims of cooperative conversation [Gri75] provided a starting point for the field of cooperative answering. For answers to be appropriate, or relevant, the user of databases and information systems in general must ask the appropriate queries in order to arrive at the desired information [GGM92]. To do so, the user must know something about the information source's schema, namely, the way the information is stored and classified. This is often hard to happen in real-world scenarios. Following [Tri02], moreover, a system "is not user-friendly when it requires a detailed knowledge of the database structure, especially in the presence of large amounts of heterogeneous data" (p. 356). But cooperative answering systems are expected to be user-friendly, and users "usually have [only] a high-level knowledge of the database structure, based on their natural perception of data correlations" (p. 357). The lack of knowledge about the database structure "does not preclude the need to enforce specific selection criteria on the results" (*ibidem*). To build information systems that meet users' expectations, we agree with [Tri02] on the tradeoff between allowing natural queries and receiving relevant answers it may be well obtained by using an interactive, cooperative and adaptive query modeling.

*Electronic correspondence to: agostini@irst.itc.it, avesani@irst.itc.it.

In this paper, we present a model whereby the ability to do appropriate queries and provide useful answers emerges as the result of a complex, dynamic and co-evolutive interaction.

1.1 Motivation: Knowledge Exchange

An increasingly common technique to solve an user’s query on the web is “guessing” its context—that is, the semantic content, conceptualisation or meaning of the query from the user [Law00]; see also [NPSR99]. Rather than explicitly requiring the user either to directly enter context information or know something about the information source’s schema, this technique guesses when such context may be relevant to the user. The problem here is that few of the results returned may be *valuable* to the user [Bar93, Miz97]. Which documents are valuable to the user depends on the semantic content of the user’s query. Under such premises, guessing relevance in such *semantic context discovery* problem becomes a true challenge.

As far as we know, today search techniques “by guessing” are limited to cases where potential contexts can be identified based on the keyword query; cf. [Law00]. Unfortunately, these cases rarely happen in the highly distributed environment of the Web, where users can manage huge amount of data and use several kinds of queries. Appropriate queries, in short, are difficult to ask. In this paper, we face the problem of appropriate query formulation in KEx.

KEx (for “Knowledge Exchange”, see [BBMN03, BBMN02]) is a distributed, knowledge management system implemented as a peer-to-peer architecture, where: “(i) each peer (called K-peer) provides all the services needed to create and organize local knowledge from an individual’s or a group’s perspective, and (ii) social structures and protocols of meaning negotiation are introduced to achieve semantic coordination among autonomous peers” [BBMN03, from the *Abstract*]. In the current version of KEx, it is indeed possible and very common that the meaning of a query, advanced by an user selecting some “seeker” from a set of available peers, is conceptualised by the seeker in a even very different way from the potential providers suitable to respond.

Now, suppose that meaning conceptualization affects query formulation. For example, this would happen to an user who looks at the names of her data directories to decide the query expression to use. A question then arises out of how query meaning and query evolution and refinements are functional to achieve query success. In this paper, we investigate this question. In particular, we take the experimental conclusion of [MBR01] as a starting point, where it claims that “a robust solution [of the schema matching problem, *viz* cooperative query answering for us] will need a module to *incrementally learn synonyms and abbreviations* from [linguistic] mappings that are performed over time” (p. 9, our emphasis).

The cooperative model we propose is based on the philosophy of language games [Wit53], the methodology and seminal extensive work by Steels on discrimination games [Ste96a], guessing games [SK99, SKML02, Ste01] and classification games [SK02], and some of the ideas about advertising games [AA03a, AA03b].

From now on we proceed as follows. The main components of our model,

basically seekers and providers, are defined in Section 2. The algorithm that determines the main steps of query answering (“query-answering game”) is advanced in the details and discussed in Section 3. The functionality of the algorithm in a real application like KEx is presented in Section 4. We end this paper with some related work (Section 5) and the conclusion (Section 6).

2 Seekers & Providers

We image that a system (“community”) of agents, or “peers,” is available to an user. The user may employ a peer to play the role of the seeker or the provider. Intuitively, a peer in the role of the seeker incorporates the functions dealing with knowledge management and query making. These functions allow the user to perform some manipulation of documents, mainly to create clusters by using distinctive features of its documents and compare documents to judge their similarity, and do queries. On the other hand, a peer in the role of the provider has the same organizational structure of the seeker—both are “peers”—plus the ability to answer queries by example.

The peers have reasoning abilities. We illustrate these abilities along the basic components, terminology and notation of our framework. The “knowledge domain,” or domain of data we consider is a set of documents—called the universal set of documents. We denote this set by \mathcal{D} . One might consider a document as any object provided with a textual content. For example, a document may be a web page, a journal article, any finite string of terms written in a previously specified language. For the aim of this paper, however, what matters is the interpretation of a document, say its semantics, and the local classification of the document, as we will see below. Each peer p has a repository consisting of a proper¹ subset of the universal set of documents, denoted by: \mathcal{D}_p .

The peers share a minimal language \mathcal{L} , defined as follows. Let $\text{Text}(\cdot)$ be a text extraction function. We refer to the keywords extraction function of [TRPG03, Sec. 4]. Given a document d , $\text{Text}(d)$ lists all the keywords in d , precisely, the most frequent bigrams that were found in a list of frequent (Italian) bigrams. Thus, $\text{Text}(\cdot)$ combines simple statistical measures with elementary linguistic knowledge. Applied to a document d , $\text{Text}(\cdot)$ produces a set $\text{Text}(d)$ of **words**. We call $\text{Text}(d)$ the (textual) **semantics of d** .²

(1) **DEFINITION:** Let $\text{Text}(\cdot)$ and \mathcal{D} be given. We define $\mathcal{L} = \text{Text}(\mathcal{D})$, where $\text{Text}(\mathcal{D}) = \bigcup_{d \in \mathcal{D}} \text{Text}(d)$.

Of course, each peer in the system may have an additional set of words to express autonomous concepts and queries (labels of nodes of a context and keywords in [BBMN03], respectively). We denote peer p ’s language by \mathcal{L}_p . In this paper,

¹We note that if p ’s repository and \mathcal{D} coincide, then it would not make sense for p to play the role of the seeker, because no documents to seek would exist in this case.

²Other text extraction functions are ammissible in our framework. Although important, a specific treatment of keywords extraction is out of the scope of this paper.

we deal with words from natural language without further specification. The use of more abstract languages, however, does not change the basic features of our approach.

(2) *Remark:* Although much of the initial work in cooperative answering focused on natural language and dialogue systems (see [GGM92] for references) and, on the other hand, language games were originally thought of to study natural language semantics—see for instance [Wit53, Ste96b, Ste97, SK02], the tenets of a cooperative query/answering system ought to be universal to queries themselves, regardless of whether the queries are cast in natural language, in formal logic, or what else.

Documents may be organized by the peers in “clusters.” In particular, documents may be aggregated by individual peers according to a general schema, or similarity measure, which is applied by each peer p locally to classify its documents repository \mathcal{D}_p . Informally, a cluster is a set of similar documents. To define a cluster formally, we rely on the following terminology and notation.

A peer manages and classifies its documents by using “features” and preferences over its “lexicon” (see subsection 2.1). Intuitively, a feature of a document is a pair composed by a (key)word in the document, to be thought of as a standardized text from procedure $\text{Text}(\cdot)$ of text extraction, and a relative value. For example, think to the frequency of occurrence of that word in the document. Following [TRPG03], given a document d we rank a word $w \in \text{Text}(d)$ on the basis of a slightly better method than just counting the number of occurrences of the word in the document. More precisely, we use the inverse document frequency [MS99].

(3) **DEFINITION:** Let peer p , document repository \mathcal{D}_p and word $w \in \text{Text}(d)$ for document $d \in \mathcal{D}$ be given. We define the **weight of w in d for p** as follows.

$$W_p[w, d] = N[w, d] \cdot \log \frac{\text{Card}(\mathcal{D}_p)}{\text{doc}_p[w]},$$

where $N[w, d]$ is the total number of occurrences of w in d , $\text{Card}(\mathcal{D}_p)$ is the number of documents in \mathcal{D}_p (\mathcal{D}_p ’s cardinality), and $\text{doc}_p[w]$ is the total number of documents in \mathcal{D}_p containing w . Note that $W_p[w, d]$ is a real number, so $W_p[\cdot, \cdot]$ is a function from $\mathcal{L} \times \mathcal{D}$ to R .

$$\log \frac{\text{Card}(\mathcal{D}_p)}{\text{doc}_p[w]}$$

is called **inverse document frequency of w for p** , in symbols: $\text{IDF}_p[w]$.

The intuitive meaning of $\text{IDF}_p[\cdot]$ is that terms which rarely occur over a collection of documents in peer p ’s repository are the most relevant. The importance of each term is assumed to be inversely proportional to the number of documents that contain the term. Since $\text{IDF}_p[\cdot]$ represents word specificity in the view of a peer p , it is expected to improve the precision of p ’s classification of documents. In particular, it has been proved (Salton and Yang, 1973; see [MS99]) that the combination

$W_p[w, d]$ of term frequency of a word in a document $N[w, d]$ with the inverse document frequency $IDF_p[w]$ of Definition (3) better characterizes documents' content than term frequency.³

We are now ready to define the “features of a document for a peer.”

(4) DEFINITION: Let peer p and document $d \in \mathcal{D}$ be given. The **features set of d for p** is the set $F_{p,d} = \{(w, v) \mid w \in \text{Text}(d), v = W_p[w, d]\}$. The **features set for p** is the set $F_p = \bigcup_{d \in \mathcal{D}_p} F_{p,d}$.

We call the pair (w, v) a **feature** (of d for p). We say that w is the attribute (or name) of the feature (w, v) , and that v is its value. Some features are “distinctive.” Informally, features are distinctive when refer to the same attribute with different values. We rely on the following definition.

(5) DEFINITION: Let features $(w_1, v_1), (w_2, v_2)$ for a peer p be given. We say that (w_1, v_1) and (w_2, v_2) are **distinctive** if $w_1 = w_2$ and $v_1 \neq v_2$.

We use distinctive features to distinguish documents. Intuitively, two documents are similar if they have similar content. To be more precise, we need some notation. Given peer p and document $d \in \mathcal{D}_p$, let V_d denote the finite sequence over real numbers—we call it **feature vector**, obtained by listing the value of each feature of d for p . That is, given $d \in \mathcal{D}_p$, we define feature vector $\langle v_1, v_2, \dots, v_k \rangle$, where v_i for $i = 1, 2, \dots, k = \text{Card}(F_{p,d})$ is the value of the i th feature of d for p .

For all peers in the system, we now define $\text{Sim}^p(\cdot, \cdot)$ to be a similarity measure defined over “normalized feature vectors.” To normalize two feature vectors, we order the attributes of each feature in each vector according to the same ordering relation, and extend the shorter vector by adding zeros. (See references in [TRPG03].)

(6) DEFINITION: Let peer p and documents $d, d' \in \mathcal{D}$ be given. We define

$$\text{Sim}^p(d, d') = \frac{\langle V_d, V_{d'} \rangle}{\|V_d\| \cdot \|V_{d'}\|},$$

where $\langle V_d, V_{d'} \rangle$ denotes the result of the scalar product of normalized feature vectors V_d and $V_{d'}$, and $\|V_d\| \cdot \|V_{d'}\|$ is the product of the vectors' norm.⁴ (See also [TRPG03, Sec. 5].)

Note that function $\text{Sim}^p(\cdot, \cdot)$ is partial recursive. We say that documents d, d' have similar content, or that are **similar for peer p** just in case $\text{Sim}^p(d, d') \neq 0$. We say documents d, d' are **similar** if d, d' are similar for some peer. Note that $\text{Sim}^p(d, d') \in [0, 1]$.

(7) *Remark:* The similarity measure $\text{Sim}^p(\cdot, \cdot)$ provides the peer p with the main mechanism to update clusters, in the sense that only retrieved documents similar to the document $d \in C$ taken by p as the example to support the query (“query by

³We refer the interested reader to [MS99] for further discussion.

⁴Recall that $\|V\| = \text{Sqr}(\langle V, V \rangle)$.

example”) are added to the cluster C . Moreover, $\text{Sim}^p(\cdot, \cdot)$ is defined according to the user needs for all peers p . Thus, we assume that whenever a peer p is selected by an user u to play the role of the seeker—in other words, u has employed p to search some documents—, we have $\text{Sim}^p(\cdot, \cdot) \equiv \text{Sim}^u(\cdot, \cdot)$. As an obvious consequence, an user with a similarity measure that differs from that considered in Definition (6) is supposed to be unable to employ p as the seeker suitable to pursue the query.

A “distinctive feature set” is a set of features distinguishing a document from a set of other documents. In a system of peers, as our system, we need to associate a distinctive feature set to peers, so that a distinctive feature set becomes a kind of local “ontology” used by a peer to classify its documents. We will see the relationship between distinctive feature set and clusters once we have defined these two concepts. We begin to define a “distinctive feature set of a document for a peer with respect to a set of documents,” in symbols: $D_{p,d}^C$.

(8) DEFINITION: Let peer p , document $d \in \mathcal{D}$, set of documents $C \subseteq \mathcal{D}_p$ and features set $F_{p,d}$ be given. We define:

$$D_{p,d}^C = \{(w, v) \in F_{p,d} \mid \forall d' \in C, \nexists (w', v') \in F_{p,d'} \ w' = w \text{ or } \exists (w, \tilde{v}) \in F_{p,d'} \ \tilde{v} \neq v\}.$$

Note that $D_{p,d}^C \subseteq F_{p,d}$, $D_{p,d}^{\{d\}} = \emptyset$, and $D_{p,d}^\emptyset = F_{p,d}$. Thus, the elements of $D_{p,d}^C$ are all features suitable by p to distinguish a document d from other documents in C . Of course, a distinctive feature set is a feature set.

We now are ready to define clusters of documents by distinctive features. Given a peer p , we denote the **relative complement** $\mathcal{D}_p \setminus C$ of a set C by \overline{C}_p .

(9) DEFINITION: Let peer p be given. A **cluster of p** is the set $C \subseteq \mathcal{D}_p$ such that for all $d, d' \in C$, the distinctive feature sets of d and d' for p with respect to \overline{C}_p are defined, and

$$D_{p,d}^{\overline{C}_p} = D_{p,d'}^{\overline{C}_p}.$$

As will be clearer by the next definition, we use distinctive feature sets to label clusters. Definition (9) has an important consequence on this, as it allows us to define a way to assign a “name” to a cluster by using preferences over documents’ distinctive features. Whenever d in Definition (9) is “the example of a query” (see also Step 2 in Section 3), we say that cluster C is the **query cluster**.

We need some notation. Given a peer p , let $\text{Pref}^p(\cdot, \cdot)$ be a total function from $\mathcal{L}_p \times (F_p \cup \mathcal{D}_p)$ to integers (denoted by Z). Positive, negative and neutral preferences are possible. Note that Definition (9) implies that for all peers p , all words w in p ’s language, and for every set of documents $C \subset \mathcal{D}_p$, $\text{Pref}^p(w, D_{p,d}^{\mathcal{D}_p \setminus C}) = \text{Pref}^p(w, D_{p,d'}^{\mathcal{D}_p \setminus C})$ for all $d, d' \in C$. This is equivalent to say that a peer’s preferences over feature sets that distinguish a document from the complement of a cluster do not depend on the choice of the document. So the next definition is sound and gives a way to decide how to name a cluster.

(10) DEFINITION: Let peer p , cluster C of p and word $w \in \mathcal{L}_p$ be given. We define:

$$\text{Pref}^p(w, C) = \text{Pref}^p(w, D_{p,d}^{\mathcal{D}_p \setminus C}),$$

if there is $d \in C$ such that $\text{Pref}^p(w, D_{p,d}^{\mathcal{D}_p \setminus C})$ is defined, and $\text{Pref}^p(w, \emptyset) = 0$. Otherwise, $\text{Pref}^p(w, C)$ is undefined.

We say that C is **labelled with w by p** just in case $\text{Pref}^p(w, C)$ is maximum, that is, $\text{Pref}^p(w', C) \leq \text{Pref}^p(w, C)$ for every $w' \in \mathcal{L}_p$ with $w' \neq w$.

2.1 Lexicon

Each agent has a “lexicon.” Given a peer p , we denote the lexicon of p by Lx_p . Informally, a peer’s lexicon is a set of pairs, where the first component is a word, or “label,” build from the peer’s language, and the second component is a feature set for the peer. Formally, we define the lexicon of peer p as follows:

$$Lx_p = \{(w, F_{p,d}) \mid w \in \mathcal{L}_p, d \in \mathcal{D}_p, \text{Pref}^p(w, F_{p,d}) \text{ defined}\}.$$

A lexicon may be either empty or incomplete, that is, there may be some word with no associated feature set or some feature set with no associated word. Notice that there is not one lexicon for all peers, but each peer has its local lexicon.

Each peer p in the system has two functions, denoted by $query(\cdot, \cdot)$ and $answer(\cdot, \cdot)$. Informally, $query(\cdot, \cdot)$ provides a peer in the role of the seeker with the way to translate the local knowledge about a document—i.e., the document used as the example for a query, into a language suitable to express the query. For simplicity, from now on we assume that each peer has a *finite* language.

(11) DEFINITION: Let peer p and $d \in \mathcal{D}_p$ be given. We define $query(F_{p,d}, Lx_p)$ to be the finite sequence $\langle w_1, w_2, \dots, w_n \rangle$ over \mathcal{L}_p such that $(w_i, F_{p,d}) \in Lx_p$ ($i = 1, 2, \dots, n$), and for all $i, j \leq n$, if $i \leq j$ then $\text{Pref}^p(w_i, F_{p,d}) \geq \text{Pref}^p(w_j, F_{p,d})$.

We call the finite sequence $\langle w_1, w_2, \dots, w_n \rangle$ **query expression**. Informally, by using $query(F_{p,d}, Lx_p)$ the peer p exploits and communicate existing high-level knowledge of the user about the document d .

Clusters’ names and function $query(\cdot, \cdot)$ are employed by peers to convey clusters’ meaning to query language. How a cluster’s name may be used to export some of the cluster’s meaning to a query is the content of the next theorem.

(12) THEOREM: Suppose that a nonempty cluster C is labelled with w by peer p . Then there is a document $d \in C$ such that $query(D_{p,d}^{\overline{C}}, Lx_p) = \langle w, w_2, \dots, w_n \rangle$ for arbitrary w_2, \dots, w_n .

Proof: Let C be a nonempty cluster of p and let C be labelled by w . Then there is $d \in C$ such that $\text{Pref}^p(w, D_{p,d}^{\mathcal{D}_p \setminus C})$ is defined, hence $(w, D_{p,d}^{\overline{C}}) \in Lx_p$ and $query(F_{p,d}, Lx_p)$ is defined. Moreover, $\text{Pref}^p(w, D_{p,d}^{\mathcal{D}_p \setminus C})$ is maximum over \mathcal{L}_p , so $query(F_{p,d}, Lx_p)$ is of the form $\langle w, \dots \rangle$. \dashv

Answering by using $answer(\cdot, \cdot)$ allows a peer in the role of the provider to interpret a query, according to the provider’s lexicon. We rely on the following definition. (Recall that \mathcal{L} denotes the universal system language, see Def. (1). Let Λ denote the set of peers in the system.)

(13) DEFINITION: Let peer p and finite sequence $\langle w_1, w_2, \dots, w_n \rangle$ over $\bigcup_{q \in \Lambda} \mathcal{L}_q \cup \mathcal{L}$ be given. We define $answer(\langle w_1, w_2, \dots, w_n \rangle, Lx_p)$ to be the set of feature sets F such that for all $i = 1, 2, \dots, n$ there is a feature of the form (w_i, \cdot) such that $(w_i, \cdot) \in F$ and $(w_i, F) \in Lx_p$.

So, each w_i is used by the peer as the input to find (“guess”) the cluster (if any) suitable to answer to the query expression $\langle w_1, w_2, \dots, w_n \rangle$.

3 Interactive Querying Algorithm

In normal operation, the user selects “the seeker” from the system of peers. The seeker selects one or more providers as the target peers for a query. The user, the seeker and the target provider(s) continuously go through a loop performing the following actions:

- Step 1. A “context” \mathcal{C} for the user is delineated (by the user itself). The context consists of a set of documents from the user’s document repository currently in the field of attention of the user. Without loss of generality, we assume $\mathcal{C} = \mathcal{D}_u = \mathcal{D}_s$ ($u =$ the user, $s =$ the seeker).
- Step 2. One document d in the context is chosen (by the user) according to the user’s distinctive feature sets, which determine the **query cluster**.⁵ The document may be thought of as “the example” of the user’s query.
- Step 3. The selected cluster C is labelled by the seeker with a word w in the seeker’s language. The seeker’s action is performed with maximum preference $\text{Pref}^s(w, C)$. Intuitively, w is the recommended name of the cluster provided (with preference $\text{Pref}^s(w, C)$) by the seeker to the user.
- Step 4. The seeker sends to the target provider(s) the feature set $F_{s,d}$ of the example and the sequence of words $\langle w_1, w_2, \dots, w_n \rangle$ (“query expression”) produced by applying function $query(\cdot, \cdot)$ to the seeker’s “knowledge” about d . Precisely, $\langle w_1, w_2, \dots, w_n \rangle = query(F_{s,d}, Lx_s)$. The query expression is a kind of recommended query to the user.
- Step 5. An attempt is made by the target provider(s) to find possible features distinguishing the example from the set of all documents in its document repository. In other words, an attempt is made by the target provider(s) to find possible distinctive feature sets on the basis of $F_{s,d}$ and the received query

⁵For simplicity, we assume that a nonempty distinctive feature set suitable to form a cluster of similar documents exists. Otherwise, we may think to extend the model to allow more features than those of the form (w, v) we have considered in this paper.

expression. The question the provider p poses is, roughly: “there are a cluster C' of p , some document(s) $d' \in \mathcal{D}_p$ and a distinctive feature set $D_{p,d'}^{\mathcal{D}_p \setminus C'}$ such that $D_{p,d'}^{\mathcal{D}_p \setminus C'} \neq \emptyset$ and $D_{p,d'}^{\mathcal{D}_p \setminus C'} \in \text{answer}(\langle w_1, w_2, \dots, w_n \rangle, Lx_p)$?” If the answer is yes, then C' is the **answering cluster** of the provider that hopefully matches with the query cluster C of the seeker.

- Step 6. The provider p sends document(s) $d' \in \mathcal{D}_p$ to the seeker, who evaluates it against the example d by using $\text{Sim}^s(d, d')$. If the provider’s result is relevant, that is, $\text{Sim}^s(d, d') > \delta$ for some fixed, user dependent relevance parameter δ , preferences over the peers’ lexicon are updated to success. Otherwise, preferences over the peers’ lexicon are updated to failure.

On the basis of the foregoing steps, that we refer to as the rules of the **query-answering game** $\mathcal{G} = \langle u, s, p, d, \mathcal{C} \rangle$, a number of query formation or query refinement situations arise. (In the following, C denotes the query cluster produced in Step 2.)

1. *The seeker is unable to express the example d by words* (Step 4 fails): Either the seeker’s lexicon Lx_s is not developed enough or the seeker’s naming function $\text{query}(\cdot, \cdot)$ is undefined over $(D_{s,d}^{\mathcal{D}_s \setminus C}, Lx_s)$. The seeker (possibly by interacting with the user) then creates a new element $(w, D_{s,d}^{\mathcal{D}_s \setminus C})$ in its lexicon with preference $\text{Pref}^s(w, D_{s,d}^{\mathcal{D}_s \setminus C}) = 0$.
2. *The provider is unable to interpret the query from the seeker* (Step 5 fails): Either the provider is unable to find possible features distinguishing the example from the set of all documents in its database or the provider does not understand the query expression from the seeker. In the first case, the game ends in failure—the provider’s response is an empty set of documents together with a message error.⁶ In the second case—suppose that the provider has found at least a set of distinguishing features, the provider extends its lexicon and creates associations between the seeker’s query expression and each suitable set of features. More precisely, the following actions are done. (Let $\langle w_1, w_2, \dots, w_n \rangle$ be the query expression from the seeker and let $D_{p,d'}^{\mathcal{D}_p \setminus C'}$ denote a set of distinguishing features found by the provider, cf. Step 5. Observe that such set depends on d' and C' , so many distinguishing features sets may be found.) Then, for all $i = 1, \dots, n$:

- the provider adds the new element $(w_i, D_{p,d'}^{\mathcal{D}_p \setminus C'})$ to its lexicon with fixed preference:

$$\text{Pref}^p(w_i, D_{p,d'}^{\mathcal{D}_p \setminus C'}) = 0.$$

⁶At this stage of the game, it would be possible for the provider to create new features to distinguish the example of the seeker from the provider’s documents. Our model development at now, however, does not allow to do this. We have planned to extend our model to deal with features out of the form (w, v) . We leave it for future work.

3. *The seeker is able to send a query expression for the example and the provider is able to interpret it* (Step 6; no linguistic failures arise): We distinguish two cases, that are also the possible outcomes of the query-answering game \mathcal{G} .
 - (a) *The provider’s response is compatible (“relevant”) with the seeker’s query.* This means that $\text{Sim}^s(d, d') > \delta$ for a fixed, previously defined, user dependent relevance parameter δ . The query succeeds, the game ends in success. Note that it is possible, indeed probable at the beginning of the game, that the query cluster and the answering cluster are labelled with different words and that the seeker and the provider(s) use different feature sets. The seeker increases the value of its lexicon preferences over the winning associations $(w_i, D_{s,d}^{\mathcal{D}_s \setminus C})$ and decreases lexicon preferences over competing associations $(w, D_{s,d}^{\mathcal{D}_s \setminus C})$ for all words w not present in the query expression. The seeker’s motivation to adapt preferences is to update cluster label. Similarly, the provider increases lexicon preferences over winning associations $(w_i, D_{p,d'}^{\mathcal{D}_p \setminus C'})$ and decreases preferences of competing associations $(w_i, D_{p,d'}^{\mathcal{D}_p \setminus C'})$ in its lexicon.
 - (b) *The provider’s response is not compatible with the seeker’s query.* This means that $\text{Sim}^s(d, d') \leq \delta$. The query fails, the game ends in failure.

4 KEx Integration: Similar Documents Service

We now describe the functionality of the algorithm, which allows us to add a new service to KEx—we call it *Similar Documents Service* (SDS).

The Service provides the user with a document retrieval semi-automatic “strategy”. When the user clicks on the *Similar Pages & Documents* link for a search result—say “give me a set of documents related to document d ”—, after a finite training period KEx automatically scouts the system of peers, or a subset of previously selected (available, preferred, etc.) peers, for documents that are related (i.e., similar in the sense of Definition (6)) to this result. In general, the similarity measure used by the user depends on the seeker employed to query the system—recall Remark (7).

The Similar Documents Service can be used for many purposes. If the user likes a particular document’s content, but wish it had more to say, Similar Pages & Documents can find documents with similar content with which the user may be unfamiliar. If the user is interested in researching a particular area or topic, say information about a particular concept or “meaning,” Similar Pages & Documents can help to find a large number of resources very quickly, without having to worry about selecting the right keywords. These are automatically generated (“recommended”) as the most successful names for the concept under attention. The more specialized a document is, the fewer results our algorithm will be able to find.

5 Related Work

Our approach to query answering is related to clustering of search results, as performed by Northern Light (*www.northernlight.com*), for example. Northern Light dynamically clusters search results into categories previously defined. These may be changed by new interactions to the user, who may narrow search results to any of these categories. More generally, there is a growing tendency of clustering techniques being used in web search (the system user - search engine is a query answering system, although not always cooperative!) to present the user with some query refinement options alongside the matching documents, that is, the documents supposed by the engine/provider to be relevant to the user. Other examples are Teoma (*www.teoma.com*) and AltaVista (*www.altavista.com*). However, an inherent problem of these techniques is naming of the clusters—that is, to select the right word or label to convey the cluster’s meaning or to select useful expressions that can serve as query refinement terms. A recent work [Kru03] studies this problem in the specific case of document collections being searched are domain-specific or limited in size.

6 Conclusion

We have presented a model whereby the ability to do appropriate queries emerges as the result of a complex, dynamic and co-evolutive interaction among peers, where each peer may play the role of the user-seeker or the provider. Our model is adaptive. We have argued that adaption, as basically influenced by the language usage, is not distinct from the categorization process behind query answering. In order to investigate the dynamic refinement of a query and query flexibility, we have advanced a model in which queries and answers are a consequence of language evolution and meaning construction. The result of this paper is a specific cooperative technique aimed at identifying, after a finite number of steps, the most relevant answer with respect to the user’s intentions and queries’ meaning. Our aim was the tight coupling of knowledge discovery and classification with query processing and evolution, as a mean to provide the user with the most relevant possible answer. For doing this, we have exploited existing high-level knowledge of the user and low-level text retrieval techniques.

References

- [AA03a] A. Agostini and P. Avesani. Advertising games for web services. In R. Meersman, Z. Tari, and D. Schmit, editors, *Eleventh International Conference on Cooperative Information Systems (CoopIS-03)*, pages 93–110, Berlin Heidelberg, 2003. Springer-Verlag LNCS 2888. Electronically available at <http://sra.itc.it/people/agostini/earchive.html>.

- [AA03b] P. Avesani and A. Agostini. A peer-to-peer advertising game. In M. Orlowksa, M. Papazoglou, S. Weerawarana, and J. Yang, editors, *First International Conference on Service Oriented Computing (ICSOC-03)*, pages 28–42, Berlin Heidelberg, 2003. Springer-Verlag LNCS 2910.
- [Bar93] C. L. Barry. *The Identification of User Criteria of Relevance and Documents Characteristics: Beyond the Topical Approach to Information Retrieval*. PhD thesis, Syracuse University, 1993.
- [BBMN02] M. Bonifacio, P. Bouquet, G. Mameli, and M. Nori. KEx: A Peer-to-Peer solution for distributed knowledge management. In *Proceedings of the Fourth International Conference on Practical Aspects of Knowledge Management (PAKM-02)*, pages 490–500, Heidelberg, 2002. Springer-Verlag LNAI 2569.
- [BBMN03] M. Bonifacio, P. Bouquet, G. Mameli, and M. Nori. Peer - mediated distributed knowledge management. In *Proceedings of AAAI Spring Symposium on Agent Mediated Knowledge Management (AMKM-03)*, Stanford, CA, 2003. Stanford University.
- [GGM92] T. Gaasterland, P. Godfrey, and J. Minker. An overview of cooperative answering. *Journal of Intelligent Information Systems*, 1(2):123–157, 1992.
- [Gri75] H. Grice. Logic and Conversation. In P. Cole and J. Morgan, editors, *Syntax and Semantics*. Academic Press, New York, 1975.
- [Kru03] U. Kruschwitz. An adaptable search system for collections of partially structured documents. *IEEE Intelligent Systems*, 18(4):44–52, 2003.
- [Law00] S. Lawrence. Context in Web Search. *IEEE Data Engineering Bulletin*, 23(3):25–32, 2000.
- [MBR01] J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with Cupid. In P. M. Apers, P. Atzeni, S. Ceri, S. Paraboschi, K. Ramamohanarao, and R. T. Snodgrass, editors, *Proceedings of the 27th International Conference on Very Large Data Bases (VLDB-01), September 11-14, 2001, Roma, Italy*, pages 49–58. Morgan Kaufmann, 2001.
- [Miz97] S. Mizzaro. Relevance: The whole history. *Journal of the American Society for Information Science*, 48(9):810–832, 1997.
- [MS99] C. D. Manning and H. Schtze, editors. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, MA, 1999.
- [NPSR99] R. Navarro-Prieto, M. Scaife, and Y. Rogers. Cognitive strategies in web searching. In *Proceedings of the Fifth Conference on Human Factors & the Web (HFWEB-99)*. NIST, 1999.

- [SK99] L. Steels and F. Kaplan. Situated grounded word semantics. In T. Dean, editor, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 862–867, San Francisco, CA, 1999. Morgan Kaufmann.
- [SK02] L. Steels and F. Kaplan. AIBO’s first words. The social learning of language and meaning. *Evolution of Communication*, 4(1):3–32, 2002.
- [SKML02] L. Steels, F. Kaplan, A. McIntyre, and J. Van Looveren. Crucial factors in the origins of word-meanings. In A. Wray, editor, *The Transition to Language*, pages 214–217. Oxford University Press, Oxford, UK, 2002.
- [Ste96a] L. Steels. Perceptually grounded meaning creation. In Y. Demazeau, editor, *Proceedings of the Second International Conference on Multi-agent Systems (ICMAS-96)*, pages 338–344, Los Alamitos, CA, 1996. IEEE Computer Society.
- [Ste96b] L. Steels. Self-organizing vocabularies. In C. Langton and T. Shimohara, editors, *Proceedings of the V Alife Conference*, Cambridge, MA, 1996. The MIT Press.
- [Ste97] L. Steels. The synthetic modeling of language origins. *Evolution of Communication*, 1(1):1–34, 1997.
- [Ste01] L. Steels. Language games for autonomous robots. *IEEE Intelligent Systems*, 16(5):16–22, 2001.
- [Tri02] A. Trigoni. Interactive query formulation in semistructured databases. In T. Andreasen, A. Motro, H. Christiansen, and H. L. Larsen, editors, *Fifth International Conference on Flexible Query Answering Systems (FQAS-02)*, pages 356–369, Berlin Heidelberg, 2002. Springer-Verlag LNAI 2522.
- [TRPG03] P. Tonella, F. Ricca, E. Pianta, and C. Girardi. Using keyword extraction for web site clustering. In Ken Wong, editor, *Fifth International Workshop on Web Site Evolution (WSE-03)*, pages 41–48, Amsterdam, The Netherlands, September 22, 2003. IEEE Computer Society.
- [Wit53] L. Wittgenstein. *Philosophical Investigations*. Blackwell, Oxford, UK, 1953.