Logic and Logical Philosophy
Volume 24 (2015), 61–78
DOI: 10.12775/LLP.2014.017

Paweł Łupkowski
Dorota Leszczyńska-Jasion

GENERATING COOPERATIVE QUESTION-RESPONSES BY MEANS OF EROTETIC SEARCH SCENARIOS*

Abstract. The concept of cooperative question-responses as an extension of cooperative behaviours used by interfaces for databases and information systems is proposed. A procedure to generate question-responses based on question dependency and erotetic search scenarios is presented. The procedure is implemented in Prolog.

Keywords: question-response; dependent question; databases, cooperative answering; inferential erotetic logic; erotetic search scenarios

Introduction

In this paper we are interested in extending the wide range of cooperative answering techniques for knowledge and database systems. The proposed extension allows such a system to respond with a question to a user's question. In order to obtain such responses we will use the concept of question dependency and erotetic search scenarios — a tool developed within A. Wiśniewski's Inferential Erotetic Logic (IEL).

The paper is structured as follows. In the first section we describe the very idea of cooperative answering. We also propose to extend standard

*This is a much extended version of the article "Cooperative question-responses and question dependency", which appeared in *Logica Yearbook 2012*. This work was supported by funds of the National Science Centre, Poland (DEC-2012/04/A/HS1/00715).

Received June 19, 2014. *Revised* August 6, 2014. *Published online* August 12, 2014

cooperative answering techniques with a capability of replying with a question. We point out motivations for this step which is rooted in research on natural language dialogues. In the second section necessary concepts taken from IEL are introduced along with the idea of dependency of questions. This logic is the main source of the tools we use for question analysis and automatic question generation. The third section contains a description of a system designed to generate cooperative responses using IEL concepts. In this section we also present our procedure and briefly discuss its implementation in Prolog. The last section covers summary and future research.

1. Cooperative Answers or Cooperative Responses?

The idea behind cooperative answering (in the context of databases and information systems) is to provide a user with an answer to his/her query which is not only correct, but also non-misleading and useful (cf. [3]). After [4, p. 2], let us consider a well known example, which shall shed some light on what counts as a cooperative answer. Imagine that a student wants to evaluate a course before enrolling in it. He asks the following question:

Q: How many students failed course number CS400 last semester?

Assume also that the course CS400 was not given last semester. For most database interface systems the answer to the student's question would be:

A_1^*: None.

This answer is correct according to the database information state. But on the other hand, it is easy to notice that it is also misleading for the student (who is not aware of the fact that the course was not given in the last semester) and thus uncooperative from our perspective. However, when we think about a secretary answering the same question we may imagine that the secretary would easily recognise the student's false assumption and correct it in her answer:

A_1: None, but the reason for this is that course number CS400 was not offered last semester.

The answer A_1 is not only correct, but it is also non-misleading and useful for the student. The cooperative answer given by the secretary facilitates student's further search. We may, for example, imagine that the next question — asked on the basis of A_1 — would be: *When was CS400 offered and how many students failed it then?*

The same idea is visible in the following examples [4, p. 142–143]. Cooperative parts of responses are emphasised.

Q: Does Smith teach Claire?
A: Yes;
by the way, Smith teaches only in the English-as foreign-language department.

Q: Does professor Smith teach in the hist department?
A: No.
Smith teaches only in the English-as-foreign-language department.

Yet another example might be the one actually provided by WEBCOOP (a cooperative question-answering system on the web) created by Benamara and Saint-Dizier [1, p. 3].

YOUR QUESTION: [Can I rent] A chalet in Corsica for 15 persons?
RESPONSE: A chalet capacity is less than 10 persons in Corsica
Flexible solutions to go further:
1. *2 close-by chalets in Corsica*
2. *Another accommodation type: hotel, pension in Corsica*
3. *A 15 person country cottage in another region in France*

A review of the literature reveals that techniques developed in the field of cooperative answering are focused on declarative sentences as reactions to the user's queries. In fact, many authors write simply about answers (as declarative sentences). As such, cooperative answering is well explored and a number of techniques have been developed in this area of research, such as: evaluation of presuppositions of a query; detection and correction of misconceptions in a query (other than a false presupposition), formulation of intensional answers or generalisation of queries and of responses. A detailed description of the above techniques is presented in [3] and in [7].

In our opinion the next step should be to extend these techniques with a question posing capability. Then we would rather consider a *cooperative response* than *cooperative answer* (see [18]).

Benamara and Saint-Dizier [2] gathered a corpus elaborated from Frequently Asked Questions sections of various web services. Besides well formed questions the corpus revealed some interesting types of questions like:

- questions including fuzzy terms (like *a cheap country cottage close to the seaside in the Cote d'Azur*);
- incomplete questions (like *What are flights to Toulouse?*);
- questions based on series of examples (like *I am looking for country cottages in mountains similar to Mr. Dupond cottage*).

These questions were not taken into account in WEBCOOP development process (because of early stage of the project at the moment). However, this type of questions asked by users suggests that questions should also be allowed as responses in such systems. Question posing ability would enable to ask a user for missing information not expressed in his/her question in a natural dialogue manner. The motivation comes from everyday natural language dialogues. As Ginzburg points out:

> Any inspection of corpora, nonetheless, reveals the undiscussed fact that many queries are responded with a query. A large proportion of these are clarification requests [...]. But in addition to these, there are query responses whose content directly addresses the question posed [...]. [5, p. 122]

This fact was also noticed by researchers working with databases. In [16, p. 444] we read:

> When presented with questions, the responses of humans often go beyond simple, direct answers. For example, a person asked a question may prefer to answer a related question, or this person may provide additional information that justifies or explains the answer.

As recent corpus study shows [15], question-responses are common in the natural language dialogues and they come in various types. One of the most common question-responses in the natural language conversations are dependent questions. The rationale behind dependent questions might be summarised as follows [5, p. 123]: question Q_1 depends on question Q_2 if discussion of Q_2 will necessarily bring about the provision of information about Q_1. This allows to say that Q_2 might be used to answer Q_1 — in other words, Q_2 is an acceptable response to Q_1.

The following examples illustrate this idea (question-responses are emphasized with a bold font):

A: *Any other questions?*
B: **Are you accepting questions on the statement of faith at this point?**
[*F85, 70–71*][1]
[*i.e. Whether more questions exist depends on whether you are accepting questions on the statement of faith at this point.*]

A: *Do you want to buy them?*
B: **How much are they?**
[*KC5 1389–1394*]
[*i.e. Whether I want to buy them depend on how much do they cost.*]

Also another type of question-responses is interesting from our perspective — namely so called FORM question-responses [15, p. 356]. FORM questions address the issue of the *way* the answer to the initial question should be given. In other words, whether the answer to the initial question will be satisfactory to a questioner depends on this kind of question-response.

We may observe this intuition in the following examples:

A: *Okay then, Hannah, what, what happened in your group?*
B: **Right, do you want me to go through every point?**
[*K75, 220–221*]
[*The way B answers A's question in this case will be dictated by A's answer to question-response — whether or not A wants to know details point by point.*]

A: [last or full name] you <pause> you,
you've been a communist yourself?
B: **Let me give you my family history shall I?**
A: Oh, oh, if you can do it in a sentence.
B: I'll do it very quickly.
[*KJS 245–248*]

The FORM type of question-responses constitutes an interesting candidate to use in the domain of cooperative responses. With this response we address directly the question asked and we establish the way the answer to this question should be given. This allows to produce better answer to the initial question — which, as a result, will be better suited to a user's needs.

[1]This notation indicates the British National Corpus file (F85) together with the sentence numbers (70–71).

In what follows we will present a procedure of generating such responses, the underling mechanism of which is based on the dependency relation between questions. Before we will do this, first we have to introduce the basic tools and concepts of the logic we will use.

2. IEL tools and concepts

One of the ways in which dependent questions may be modelled is by the use of erotetic implication (e-implication), which is one of the key concepts of IEL (see [19, 23]).[2] IEL is a logic which focuses on inferences whose premises and/or conclusion may be a question, and which gives criteria of validity of such inferences. IEL gives a very useful and natural framework for analyses of the questionning process. What is more, IEL-based concepts proved useful when applied in the problem solving area[3] as well as in proof-theory.[4]

In the sequel we will use formal language L specified as follows. The language has declarative and erotetic part. The declarative part of L is a first-order language with identity and individual constants, but without function symbols. The concepts of well-formed formula (d-wff for short), freedom and bondage of variables, and that of sentential function and sentence are defined as usual.[5] The vocabulary of the erotetic part of L consists of the signs: ?, {, }, and the comma.

[2] It is worth to mention other approaches here, like compliance in inquisitive semantics [8], [14]; topicality [17] or KoS [6], [15].

[3] [9] use an implementation of a proof method based on IEL for generation of abductive hypotheses. Abductive hypotheses are evaluated by multi-criteria dominance relations.

[4] There are two proof-methods grounded in IEL — the method of Socratic proofs and the synthetic tableaux method. A theorem-prover (an implementation of the method of Socratic proofs for Classical Propositional Logic described in [21]) written in Prolog by Albrecht Heefer is available at: http://logica.ugent.be/albrecht/socratic.html. A theorem-prover (an implementation of the method of Socratic proofs for 15 basic propositional modal logics described in [10, 11]) written in Prolog by Albrecht Heefer and Dorota Leszczynska-Jasion, is available at: http://logica.ugent.be/albrecht/socratic-modal.htm.

[5] In [20, pp. 20–21] the reader may find a similar description of a first-order language with questions. However, the language considered by Wiśniewski is monadic, and we need predicates of higher arity. For the syntax and semantics of the language of the first-order logic (in the form in which it is usually presented with IEL in the background) the reader may also see [24, pp.148–149].

Questions of L are expressions of the following form:

$$?\{A_1, A_2, \dots, A_n\}$$

where $n > 1$ and $A_1, A_2, \dots, A_n$ are nonequiform, that is, pairwise syntactically distinct, d-wffs of L. If $?\{A_1, A_2, \dots, A_n\}$ is a question, then each of the d-wffs $A_1, A_2, \dots, A_n$ is a *direct answer* to the question. If Q is a question, then by $\boldsymbol{d}Q$ we refer to the set of all the direct answers to Q.

A question $?\{A_1, A_2, \dots, A_n\}$ can be read, 'Is it the case that A_1, or is it the case that A_2, ..., or is it the case that A_n?'.

As to the semantics we assume the standard model-theoretical approach in the case of the declarative part of L. In definitions 1 and 2 we refer to the entailment relation as defined on the grounds of model-theoretical semantics. The central semantic concept for the erotetic part of L is that of erotetic implication:[6]

Definition 1 ([19, cf. p. 25]). A question Q *implies* a question Q^* *on the basis of* a set of d-wffs X (in symbols: $\mathsf{Im}(Q, X, Q^*)$) iff

1. for each direct answer A to the question Q: $X \cup \{A\}$ entails the disjunction of all the direct answers to the question Q^*, and
2. for each direct answer B to the question Q^* there exists a non-empty proper subset Y of the set of direct answers to the question Q such that $X \cup \{B\}$ entails the disjunction of all the elements of Y.

If $X = \emptyset$, then we say that Q implies Q^* and we write $\mathsf{Im}(Q, Q^*)$.

[6]Semantics of the erotetic part may be based on the ideas of Minimal Erotetic Semantics (MiES for short). Explaining the details of MiES goes beyond the scope of this paper, the reader may refer to chapters 3 and 4 of [23] and the literature cited there. We will only observe that MiES is a very rich and flexible background for semantical analysis of erotetic languages. On pages 31–32 of [23] the reader may find MiES semantics for the language of Monadic First-Order Logic with questions. In order to obtain the MiES semantics for our language L it is sufficient to take a more general notion of the model of the declarative part of L, that is, one in which the interpretation function assigns appropriate values to the predicates of higher arities. Then the concept of admissible partition defined on page 32 may be applied in the present approach. However, in order to make this paper self-contained, we decided to present here all the necessary concepts in a simplified version which does not require the MiES analysis.

The first condition requires that if the implying question is sound[7] and all the declarative premises are true, then the implied question is sound as well. This property may be conceived as an analogue to the truth-preservation property of deductive schemes of inference. The second condition requires that each answer to the implied question is potentially useful, on the basis of declarative premises, for finding an answer to the implying question. To put it informally: each answer to the implied question Q^*, on the basis of X, narrows down the set of plausible answers to the implying question Q.

Now we may introduce a definition of an e-scenario.[8]

Definition 2 (E-scenario, [23, p. 116]). A finite labelled tree Φ is an erotetic search scenario for a question Q relative to a set of d-wffs X iff

(1) the nodes of Φ are labelled by questions and d-wffs; they are called e-nodes and d-nodes, respectively;
(2) Q labels the root of Φ;
(3) each leaf of Φ is labelled by a direct answer to Q;
(4) $\mathbf{d}Q \cap X = \emptyset$;
(5) for each d-node γ_δ of Φ: if A is the label of γ_δ, then
  (a) $A \in X$, or
  (b) $A \in \mathbf{d}Q^*$, where $Q^* \neq Q$ and Q^* labels the immediate predecessor of γ_δ;
  (c) $\{B_1, \ldots, B_n\} \models A$, where B_i $(1 \leq i \leq n)$ labels a d-node of Φ that precedes the d-node γ_δ in Φ;
(6) each d-node of Φ has at most one immediate successor;
(7) there exists at least one e-node of Φ which is different from the root;
(8) for each e-node γ_ε of Φ different from the root: if Q^* is the label of γ_ε, then $\mathbf{d}Q^* \neq \mathbf{d}Q$ and
  (a) $\mathsf{Im}(Q^{**}, Q^*)$ or $\mathsf{Im}(Q^{**}, \{B_1, \ldots, B_n\}, Q^*)$, where Q^{**} labels an e-node of Φ that precedes γ_ε in Φ and B_i $(1 \leq i \leq n)$ labels a d-node of Φ that precedes γ_ε in Φ, and
  (b) an immediate successor of γ_ε is either an e-node or is a d-node labelled by a direct answer to the question that labels γ_ε, moreover
    - if an immediate successor of γ_ε is an e-node, it is the only immediate successor of γ_ε,

[7] A question Q is *sound* iff it has a true direct answer (with respect to the underlying semantics).

[8] Here we present e-scenarios as trees. Also other approach is possible (see [12]).

- if an immediate successor of γ_ε is not an e-node, then for each direct answer to the question that labels γ_ε there exists exactly one immediate successor of γ_ε labelled by the answer.

The pragmatic intuition behind the e-scenario is that it

> [...] provides information about possible ways of solving the problem expressed by its principal question: it shows what additional data should be collected if needed and when they should be collected. What is important, an e-scenario provides the appropriate instruction for every possible and just-sufficient, i.e. direct answer to a query: there are no "dead ends". [22, p. 110]

An exemplary e-scenario is presented in Figure 2.

For the proposed approach the idea of a query of an e-scenario is also important.

DEFINITION 3. A *query* of an e-scenario Φ is an e-node Q^* of Φ different from the root of Φ and such that the immediate successors of Q^* are the direct answers to Q^*.

As it might be noticed e-scenarios are constructed in such a way that all queries are closely related to the initial question by the dependency relation (in IEL expressed in terms of e-implication). We will use this feature in order to generate question-responses. This will ensure that question-responses generated on the basis of e-scenarios will be relevant to the user's question.

3. E-scenarios in Generating Question-responses

3.1. IEL-based system architecture

In this section we will introduce a simple technique of generating cooperative question-responses on the basis of question dependency check in an e-scenario. Let us assume that we are dealing with a deductive database. It consists of an extensional database (EDB) — built out of facts, intensional database (IDB) — built out of rules, and integrity constraints (IC). We will also assume that there is a cooperative layer between the database and a user where e-scenarios are stored and processed. Figure 1 illustrates the idea of an architecture of such a system.

Cooperative layer is responsible for:

- e-scenarios generation,

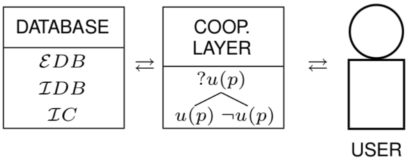

Figure 1. Scheme of the cooperative database system using e-scenarios

- e-scenarios storage,
- questions' analysis,
- e-scenarios execution against a database.

We assume that e-scenarios stored in the cooperative layer are generated for a given database. Rules from IDB are used as declarative premises for e-scenarios.

The layer copes with interactions between a user and a database:

- First a user's question is analysed in the layer.
- Then — when it is needed — question-response is generated.
- After the user's reaction the next step is executed.
- On the basis of this interaction the cooperative layer may interact with the database.
- After the interaction it analyses the data obtained and it may supplement the data with additional explanations useful for the user (for the details and examples of this step see [13]).

It is worth to stress that e-scenarios are used on nearly all of these steps.

3.2. The procedure

Let us consider a simple (toy) example of a deductive database presented in Table 1.

EDB	IDB	IC
$usr(a)$ $usr(b)$ $usr(c)$ $live(a,p)$ $live(b,zg)$ $live(c,p)$	$locusr(x) \to usr(x)$ $locusr(x) \to live(x,p)$ $usr(x) \wedge live(x,p) \to locusr(x)$	$\neg(\exists x(live(x,zg) \wedge live(x,p)))$

Table 1. Example of deductive database

As it might be noticed IDB contains rules for the database. Also new concepts might be introduced here (see the concept *locusr* in Table 1). E-scenarios stored in the cooperative layer are built on the basis of IDB rules (IDB rules are used as premises). For example a relevant e-scenario for a question of the form 'Is a_i a local user?' would fall under the schema presented in Figure 2.

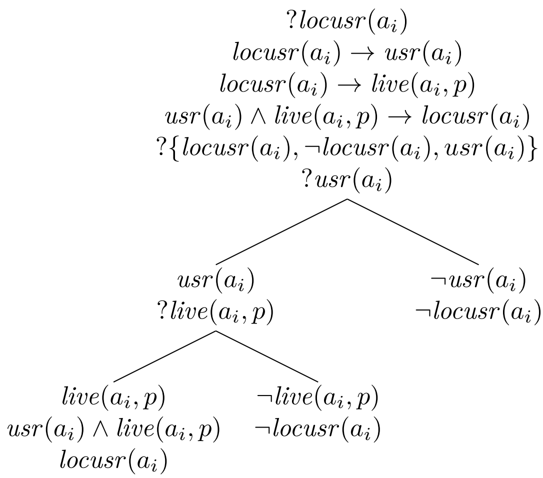

Figure 2. Schema of an e-scenario for a question of the form 'Is a_i a local user?'

The following logical facts were used in designing this e-scenario:

1. $\mathsf{Im}(?A, C \to A, ?\{A, \neg A, C\})$
2. $\mathsf{Im}(?\{A, \neg A, C\}, ?C)$
3. $\mathsf{Im}(?A, B_1 \wedge B_2 \to A, B_1, A \to B_2, ?B_2)$

As e-scenarios are stored in the cooperative layer between a user and a database, each question of a user might be processed and analysed against these e-scenarios. We will consider two types of user's questions: (i) about facts (i.e. concerning EDB part) and (ii) about concepts introduced in the IDB part of the database. In both cases the procedure would be the same. The task will be to find user's query in e-scenarios stored in the cooperative layer. When a query is found, its position in the e-scenario should be checked. There are two possibilities:

1. the user's question is the initial question of the e-scenario (in our example, e.g. questions of the form $?locusr(a_i)$);

2. the user's question is one of the queries of the e-scenario (e.g. questions of the form $?live(a_i, p)$).

Now — on the basis of this search — we may generate two types of question-response before executing user's query against the database:

1. *Were you aware that your question depends on the following questions ...? Would you also like to know answers to them?*
   This question-response allows a user to decide how many details he/she wants to obtain in the answer. This also might be potentially useful for future search.
2. *Your question influences a higher level question. Will you elaborate on this subject (follow search in this topic)? May I offer a higher level search?*

The procedure boils down to matching user's question with a question present among the stored e-scenarios. Then generating question-response is rather simple. It should be stressed that it is possible due to the properties of e-scenarios. It is the logical background that facilitates the procedure.

We assume that e-scenarios may be indicated by the user (e.g. if the user wants his/her question to be analysed in connection with some specific issues) or not. In the other case the e-scenarios may be generated when executing the procedure (it may be done in the case of e-scenarios which do not contain any initial declarative premises, see also the last section) or, finally, the e-scenarios stored in the cooperative layer may be used. The algorithm presented below models the last kind of situation — when the e-scenarios are stored in some data base and a question to be analysed is given as input. It is assumed that the e-scenarios are numbered and that they are indicated by their numbers. (Actually, this is how it works in the Prolog implementation.)

If the user, informed about the dependency relation between his/her question and some other questions, is further interested in detailed description of connections between the questions, then the program should perform an analysis of such connections and return its results.[9] In our case this step is modeled by returning the whole e-scenario to the user.

[9] In the case of really big e-scenarios (with many auxiliary questions) it would be useful to allow a user to decide how many questions influencing the initial question to report.

Algorithm 1 presents a simplified schema of our program's work.

Algorithm 1 User's question analysis and question-response generation

```
Require: question Q to be analysed, e-scenarios (numbered)
Ensure: question-response of an appropriate type
  n ← the number of e-scenarios in the data base
  for i = 1 to n do
    if question Q is the initial question of e-scenario i then
      find the queries of e-scenario i;
      L ← the list of the queries;
      print "Were you aware that your question depends on the fol-
      lowing questions:"
      print L
      print "Are you interested in connections between these ques-
      tions?"
      if the user answers YES then
        return e-scenario i
      end if
    else
      if question Q is one of the queries of e-scenario i then
        Q* ← the initial question of e-scenario i;
        print "Your question influences a higher level question:"
        print Q*
        print "Will you elaborate on this subject? Are you interested
        in connections between these questions?"
        if the user answers YES then
          return e-scenario i
        end if
      end if
    end if
  end for
```

As we can see, the procedure is very simple — it goes through the e-scenarios and looks for its questions trying to match them with the user's query. The main structure used in our procedure is the for-loop and the number of possible loop's iterations is the number of e-scenarios stored in the database. So the running time of the implementation depends mainly on the size of the database. Inside the loop one of the e-scenarios is examined with respect to the questions that occur in it, thus the running time of one iteration of the loop is a function of the size of the e-scenario. Obviously, e-scenarios and databases are finite objects, thus the procedure terminates. Actually, in the Prolog implementation the answer is returned after finding each single e-scenario with a question that fits, and then the next e-scenarios are examined by forcing Prolog to backtrack.

Now let us consider some simple examples of questions evaluated against the exemplary database (Table 1). By uQ we designate the user's question, and by Q_i we refer to the initial question of Φ_i (i.e. the root of Φ_i).

Example 1. uQ_1: Is c a local user? $(?locusr(c))$.

On the basis of a schema presented in Figure 2 we generate an e-scenario for question $?locusr(c)$ by substituting c for a_i. Let us refer to this e-scenario as Φ_1. Consequently we will refer to its initial question $?locusr(c)$ as Q_1.

In this case $uQ_1 = Q_1$ so the procedure will generate a question-response of the first type. To report all questions on which uQ_1 depends the procedure returns all queries of Φ_1, i.e.: $?usr(c)$ and $?live(c, p)$. So the response in the natural language form would be:

Were you aware that your question depends on the following questions: 'is `c` *a user?' and 'does* `c` *live in* `p`*?'? Would you also like to know their answers?*

Example 2. uQ_1: Does c live in p? $(?live(c, p))$.

Also in this case we will use Φ_1. This time $uQ \neq Q_1$, so the procedure tries to match uQ with queries of Φ_1. Such matching is successful for question: $?live(c, p)$. The higher level question reported in question-response will be simply Q_1. So the response in the natural language form would be:

Your question influences a higher level question: 'is `c` *a local user?'. Will you elaborate on this subject (follow search in this topic)? May I offer a higher level search?*

The following examples are direct outputs of our implementation.[10]

Example 3. `?- analyse(?usr(a)).`

```
Your question influences a higher level question: ?locusr(a)
Will you elaborate on this subject?
You may search for e-scenario no 1.
true.
```

[10]The Prolog program with an exemplary database is available to download from `http://kognitywistyka.amu.edu.pl/intquestpro/`; Resources.

```
?- analyse(?live(a,p)).
Your question influences a higher level question: ?locusr(a)
Will you elaborate on this subject?
You may search for e-scenario no 1.
true.

?- analyse(?locusr(a)).
Were you aware that your question depends on the following
questions: ?usr(a), ?live(a,p),
You may ask me these questions. You may also search
for e-scenario no 1.
true.
```

What is important, all the necessary data needed to generate question-responses of the analyses kinds are obtained on the basis of e-scenarios analysis before their execution against the database (i.e. the analysis is performed in the cooperative layer).[11]

Of course it might be the case that user's question will be identified in more than one e-scenario. Then presented question-responses should report all e-scenarios found. This will have the effect that a user will be aware of contexts in which his/her query is involved in the database. As these are question-responses, the system is expecting user's answer. This answer might be negative, i.e. a user will not use proposed suggestions. In this way the interaction with the system will be closer to the natural language interaction.

Summary

The proposed simple technique of generating question-replies will enrich cooperative interfaces with question posing capability. What is important, functionality offered by question-replies will be analogical to the functionality of cooperative answers, i.e. it will:

- inform a user (in an indirect manner) about the database schema (this will influence his/her future search and should allow to avoid wrongly formulated questions);
- adjust the level of generality of provided answers to the user's current needs;
- personalise the user's questioning process.

[11] It is worth to mention that also other techniques of cooperative answering might be used with e-scenarios (after their execution, see [13]).

Last but not least, involving questions into the cooperative answering process is motivated with natural language dialogues. As a result, interactions with databases and information systems may become more 'natural' and somehow closer to the real-life conversations.

Future work will be focused on implementing techniques presented in [13]. These techniques — also based on e-scenarios analysis — allow to supplement a response given to a user with additional information (useful for a user confronted with a querying failure).

One of the most demanding tasks will be to develop the procedure of automatic e-scenarios generation on the basis of a given database. It is worth to notice that we have already implemented a procedure to generate ESS without declarative premises involved (see http://kognitywistyka.amu.edu.pl/intquestpro/; Resources).

Acknowledgement. We would like to give our thanks to M. Urbański for helpful feedback and comments on a draft of this article. We also thank an anonymous referee for her/his helpful remarks.

References

[1] Benamara, F., and P. Saint-Dizier, "Dynamic generation of cooperative natural language responses", in *Proceedings of the EACL Workshop on Natural Language Generation*, 2003.

[2] Benamara, F., and P. Saint-Dizier, "WEBCOOP: a cooperative question-answering system on the web", in *Proceedings of the tenth conference of European chapter of the Association for Computational Linguistics*, 2003.

[3] Gaasterland, T., P. Godfrey, and J. Minker, "An overview of cooperative answering", *Journal of Intelligent Information Systems*, 1 (1992): 123–157. DOI: 10.1007/BF00962280

[4] Gal, A., *Cooperative Responses in Deductive Databases*, PhD thesis, University of Maryland, Department of Computer Science, 1988.

[5] Ginzburg, J., "Relevance for dialogue", pages 121–129 in *Aspects of Semantics and Pragmatics of Dialogue. SemDial 2010, 14th Workshop on the Semantics and Pragmatics of Dialogue*, P. Łupkowski and M. Purver (eds.), Polish Society for Cognitive Science, Poznań, 2010.

[6] Ginzburg, J., *The Interactive Stance: Meaning for Conversation*, Oxford University Press, Oxford, 2012.

[7] Godfrey, P., "Minimization in cooperative response to failing database queries", *International Journal of Cooperative Information Systems*, 6, 2 (1997): 95–149. DOI: 10.1142/S0218843097000070

[8] Groenendijk, J., "Inquisitive semantics: Two possibilities for disjunction", pages 80–94 in *Logic, Language, and Computation*, P. Bosch, D. Gabelaia, and J. Lang (eds.), volume 5422 of "Lecture Notes in Computer Science", Springer Berlin–Heidelberg, 2009.

[9] Komosinski, M., A. Kups, D. Leszczyńska-Jasion, and M. Urbański, "Identifying efficient abductive hypotheses using multi-criteria dominance relation", *ACM Transactions on Computational Logic*, 2014 (in print).

[10] Leszczyńska. D., "Socratic proofs for some normal modal propositional logics", *Logique et Analyse*, 47, 185–188 (2004): 259–285.

[11] Leszczyńska-Jasion, D., *The Method of Socratic Proofs for Normal Modal Propositional Logics*, Adam Mickiewicz University Press, Poznań, 2007.

[12] Leszczyńska-Jasion, D., "Erotetic search scenarios as families of sequences and erotetic search scenarios as trees: two different, yet equal accounts", Technical report, Adam Mickiewicz University, Poznań, 2013. http://kognitywistyka.amu.edu.pl/intquestpro

[13] Łupkowski, P., "Cooperative answering and inferential erotetic logic", pages 75–82 in *Aspects of Semantics and Pragmatics of Dialogue. SemDial 2010, 14th Workshop on the Semantics and Pragmatics of Dialogue*, P. Łupkowski and M. Purver (eds.), Polish Society for Cognitive Science, Poznań, 2010.

[14] Łupkowski, P., "Compliance and pure erotetic implication", pages 105–114 in *Logica Yearbook 2013*, V. Punčochář (ed.), College Publications, London, 2014.

[15] Łupkowski, P., and J. Ginzburg, "A corpus-based taxonomy of question responses", pages 354–361 in *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*, Potsdam, Germany, Association for Computational Linguistics, March 2013.

[16] Motro, A., "Intensional answers to database queries", *IEEE Transactions on Knowledge and Data Engineering*, 6, 3 (1994): 444–454. DOI: 10.1109/69.334858

[17] van Kuppevelt, J., "Discourse structure, topicality and questioning", *Journal of Linguistics*, 31 (1995): 109–147. DOI: 10.1017/S002222670000058X

[18] Webber, B. L., "Questions, answers and responses: Interacting with knowledge-base systems", pages 365–401 in *On Knowledge Base Management Systems*, M. Brodie and J. Mylopoulos (eds.), Springer, 1985.

[19] Wiśniewski, A., *The Posing of Questions: Logical Foundations of Erotetic Inferences*, Kluwer AP, Dordrecht, Boston, London, 1995.

[20] Wiśniewski, A., "Questions and inferences", *Logique et Analyse*, 173–175 (2001): 5–43.

[21] Wiśniewski, A., "Socratic proofs", *Journal of Philosophical Logic*, 33, 3 (2004): 299–326. DOI: 10.1023/B:LOGI.0000031374.60945.6e

[22] Wiśniewski, A., *Essays in Logical Philosophy*, LIT Verlag, Berlin–Münster–Wien–Zürich–London, 2013.

[23] Wiśniewski, A., *Questions, Inferences and Scenarios*, College Publications, London, 2013.

[24] Wiśniewski, A., and V. Shangin, "Socratic proofs for quantifiers" *Journal of Philosophical Logic*, 35, 2 (2006): 147–178. DOI: 10.1007/s10992-005-9000-0

Paweł Łupkowski and Dorota Leszczyńska-Jasion
Department of Logic and Cognitive Science
Institute of Psychology
Adam Mickiewicz University
Poznań, Poland
{Pawel.Lupkowski,Dorota.Leszczynska}@amu.edu.pl