Available online at www.sciencedirect.com
ScienceDirect
Journal of Applied Logic 5 (2007) 92–103
JOURNAL OF APPLIED LOGIC
www.elsevier.com/locate/jal

Using logical relevance for question answering

Marco De Boni [1]

School of Computing, Leeds Metropolitan University, Leeds LS6 3QG, UK

Available online 18 January 2006

Abstract

We propose a novel method of determining the appropriateness of an answer to a question through a proof of logical relevance rather than a logical proof of truth. We define logical *relevance* as the idea that answers should not be considered as absolutely true or false in relation to a question, but should be considered true more flexibly in a sliding scale of aptness. This enables us to reason rigorously about the appropriateness of an answer even in cases where the sources we are getting answers from are incomplete or inconsistent or contain errors. We show how logical relevance can be implemented through the use of measured simplification, a form of constraint relaxation, in order to seek a logical proof than an answer is in fact an answer to a particular question. We then give an example of such an implementation providing a set of specific rules for this purpose.

Keywords: Question answering; Logical relevance; Relevance

1. Introduction

Research in Question Answering (QA) systems aims to find a method for answering a question by searching for a precise response to a natural language question in a collection of documents. Unlike search engines, QA systems seek to understand users' questions and aim to present a user with *an answer* as opposed to a set of documents *containing* an answer.

Research in automated QA has been carried out for a number of years, and a number of different approaches have been proposed in the past (see for example [6,13,14,24,35,39]). Research on the logical and philosophical foundations of Question Answering also spans a number of years (see for example [4,5,14,15,21,27,34]). These proposals have generally focussed on limited domains and "toy" systems and been found to be of limited use in "real" systems. Most current applied research, which aims to produce working general-purpose ("open-domain") systems, is based on a (relatively) simple architecture, combining Information Extraction and Retrieval, as exemplified by the systems presented at the standard evaluation framework given by the Text Retrieval Conference (TREC) QA track [37,38].

In this paper we aim to move beyond both the ideas of logical proof of answerhood developed in linguistics and mathematical logic, which have been shown to have limited applicability in actual systems, proposing instead a practical system of logical relevance, the idea that answers should not be considered as absolutely true or false answers

E-mail address: marcodeboni@hotmail.com (M. De Boni).
[1] Currently at: Unilever Corporate Research, Unilever Colworth, Sharnbrook, Bedford, MK44 1LQ, UK.

1570-8683/$ – see front matter
doi:10.1016/j.jal.2005.12.003

to a question, but should be considered true in a sliding scale of aptness to answer a question. We do this by providing a method for applying the idea of logical relevance into "real" open-domain systems.

2. Previous work: logic for QA

2.1. Computational approaches to finding the logical connection between a question and an answer

A number of researchers have examined the logical form of questions (e.g. [17–19]) and questions as logical entities distinct from statements (see for example Prior and Prior [33], who introduced the term "erotetic logic" to describe this field of study; see [14] and [21] for a comprehensive review of work in this area). The limitation of this work is however that it only considers a limited part of the problem of question answering, neglecting usage and taking into account only very simple examples of questions, usually closed-concept questions such as "Who did Verdi write his Requiem for?", ignoring the more complicated open-ended questions such as "Why is Manzoni important in the history of the Italian language?".

In actual systems, theorem proving has long been used as a model for question answering. Some of the earliest examples of this approach are [15,16,27,34] who used resolution refutation to prove answers; in this framework, answers to a "yes/no"-type questions Q are given by adding the addition of the negation of Q to a knowledge base K: if this addition renders the knowledge base inconsistent, then the answer to the question is positive. Question answering systems such as those described by Schank, Abelson and Dyer [13,35], relied on the ability to transform natural sentences into some form of logic and then using a mechanical procedure to discover (or prove) an answer. More recent work has included [10,11] who used a logic framework in order to understand generic answers; [7], who used description logics in order to find both generic and specific answers to questions; and [8], who carried out an in-depth analysis of the use of resolution refutation for question answering. Nevertheless, all these attempts at using logical proof to find answers to questions have been limited to very specific domains or "toy" problems.

Recent research in open-domain QA systems (such as the ones presented in the various TREC conferences) has generally avoided deep logical approaches (early attempts at this, for example [22], performed worse than systems which didn't use any form of deep sentence analysis), adopting instead a combination of information retrieval and extraction techniques, with variations of simple rules such as the following: "*If a question starts with "when", and a sentence contains a time entity, then that sentence is an answer to that question*". More recently, researchers have returned to the idea of using a logical proof which connects question and answer as a fallback mechanism in order to solve some of the cases in which information retrieval/extraction and pattern matching techniques fail to find an answer (for a particularly successful example, see [20]; see [26] for a method for finding inference rules for such a system).

3. Logical relevance

3.1. Relevance

Applying the "logical" approach to question answering outlined above is problematical in "real" text for a number of practical problems such inconsistencies between documents ($A \,\&\, -A$, a typical case when we have multiple sources for our knowledge, for example the *Financial Times* and *Le Monde*), errors (A instead of B, typical where we seek answers in documents such as newspapers with an abundance of misspellings and typographical errors), omissions (A instead of $A \,\&\, B$, again a typical case when looking in documents such as newspapers which often contain only partial information). An alternative to using classical logic is to consider the notion of *relevance*, not the notion of truth, as fundamental to question answering (see [36] for an introduction to the notion of relevance in communication): the idea behind the concept of relevance is that answers are not simply true or false, but should be considered instead more or less useful/interesting to a question. Logical relevance will enable us to move beyond the notion of a "true" or "false" answer and instead present a ranking of answers which are to a greater or lesser degree logically related to a question.

3.2. Previous work: entailment and relevance logic

The importance of the concept of relevance has been recognised by a number of researchers in the field of logic. In logic, relevance is the idea that in "if ... then" propositions the antecedent must be in some way "relevant" to the consequent. In human logical thinking, the conditional between two propositions, as in "if X then Y", depends not only on the truth of X and Y, but also on a causal relationship between X and Y [9]. This is in opposition to the mathematical treatment of "if ... then" propositions, where the relevance of antecedent to consequent plays no part whatsoever and in fact propositions such as $(-X\_X) \rightarrow Y$ are considered "true", even though natural human logical thinking would immediately reject them. The idea of relevance in logic, although central in classical philosophers from Aristotle until the beginning of the nineteenth century, was abandoned by the more mathematically inclined tradition of Frege, Whitehead and Russell [2]. "Relevance logic", brought to the attention of mathematicians by Ackermann [1], tries to give a mathematically satisfactory way of grasping the idea of "relevance" in implication. One way in which relevance logic helps in overcoming the problems of inconsistent, missing or incorrect information is by substituting the two-valued semantics of "classical" logic with a four-valued semantics where the truth values are the powerset of $\{t, f\}$: $\{t\}$, indicating that the system has evidence to the effect that a proposition is true; $\{f\}$, indicating that the system believes that the proposition is false; $\{\}$, corresponding to lack of knowledge; $\{t, f\}$, indicating inconsistent knowledge. This allows for only a "mild" form of modus ponens, avoiding the "paradoxes" of logical implication, in particular: not allowing contradictory premises to entail anything (i.e. not allowing $A\_ - A \rightarrow B$).

A number of logic systems that attempt to avoid the "implicational paradoxes" of mathematical logic have been devised. One of the earliest mathematicians to argue that in order for the sentence $A \rightarrow B$ to be true there should be "some connection in meaning between A and B" was [32]; a number of relevance logic systems have been developed since then, notably, the "system R of Relevant Implication", first formulated (independently) by [30] and [12]; "system PI' of rigorous implication" presented in [1]; the "system E of entailment" in [2]; the logic "$E(fde)$ of tautological entailments" [1], based on a four-valued denotational semantic developed by [4]; the "systems Cm, Cn and Cnd of entailment" in [25]; the "system C of conditional" described by [9].

None of these methods have been implemented in actual Question Answering systems. Relevance logic has however been examined for the related task of information retrieval: see [29] who showed that the problem was decidable, but EXPTIME-hard; it is therefore doubtful that current implementations of relevance logic could be successfully applied to a more complex problem such as question answering. More importantly, however, there is no indication that this mathematical notion of relevance logic would be able to represent the complex linguistic structures that make up language and that enable humans to make relevance judgments about texts.

3.3. A novel approach: from relevance logic to logical relevance

As seen, there are a number of mathematical approaches to relevance logic, but no practical implementation. It is unrealistic, given the state of the art, to think that any of the proposed theories could be implemented in the near future. What is needed is a method for providing more than one provable answer, ranked in order of reliability. We do this by providing a new measure of *logical relevance* as opposed an implementation of the mathematical concept of *relevance logic*.

We shall build on the idea that eliminating constraints on the proof of an answer can be used not only to find answers where answers are malformed, contradictory or where there is insufficient knowledge, but also to provide a ranking of answers in order of relevance. For example, take the question:

Q: Where was the French composer Jean-Baptiste Lully born?

Consider the following answer sentences:

A_1: The French composer Jean-Baptiste Lully was born in Florence.
A_2: The Italian composer Jean-Baptiste Lully was born in Florence.
A_3: Jean-Babtiste Lully was born in Florence.
A_4: The composer Lully was born in Florence.
A_5: The French composer Jean-Philippe Rameau was born in Dijon.

Although to a human reader all the sentences $A_1 \ldots A_4$ contain a correct answer to Q (i.e. Florence), while A_5 is completely irrelevant, it is clear that we have inconsistencies (A_1 and A_2 imply that Lully was both Italian and French—a true but possibly confusing fact), errors (A_3 misspells "Baptiste") and omissions (which Lully does A_4 refer to?). A human would probably use any of $A_1 \ldots A_4$ to provide an answer to Q, "intuitively" understanding that incomplete or inconsistent sentences, possibly containing spelling mistakes, can still be used to give a useful answer to the question. But from a strict logical point of view, only A_1 satisfies all the constraints specified by the question and hence, strictly speaking, only A_1 is an answer to Q. What is needed is an alternative which recognises the relevance of $A_2 \ldots A_4$, while also acknowledging that we should be happier with A_1 and that A_5 does not provide any useful information to the question. The idea of *logical relevance* provides such an alternative.

4. Implementation of logical relevance

4.1. Relevance as constraint relaxation

We shall now show a possible implementation of logical relevance, moving from a logical *connection* between question and answer to the concept of logical *relevance* through a progressive relaxation on the constraints on which any logical "proof" of the answerhood of an answer depends. *Constraints* are the terms in the question which define the acceptable properties of an answer sentence: in the question "Who was Puccini's wife?", constraints are the terms "Who", indicating the answer should be a person, "wife", indicating that the answer should be a married female and "Puccini's", indicating that the answer should be in a marriage relationship with "Puccini". Our approach can be seen to have similarities with the idea of soft constrain satisfaction, or partially satisfied constraints [3]: question answering can be seen as an over-constrained problem; the idea of logical relevance indicates that even if all the constraints may not be satisfied, nevertheless they may be partially satisfied by certain answers, giving degrees of satisfaction.

We follow [36] in judging relevance depending on the amount of effort needed to "prove" that a particular answer is relevant to a question. The fundamental rule we propose is the following

Relevance Measure: *the less effort required to prove an answer, the more relevant that answer is*

The effort required could be measured in terms of amount of prior knowledge needed, inferences from the text or assumptions. In order to provide a more manageable measure we propose to simplify the problem by focussing on ways in which constraints may be removed from the question, or in other words, how the question may be simplified in order to prove an answer. Consequently we have a *measured* simplification rule:

Revised Relevance Measure: *the relevance of an answer is determined by how many constraints must be removed from the question for the answer to be proven*; *the less constraints must be removed, the more relevant the answer*

A more elaborate version of the relevance measure would give a weighting to each constraint, for example depending on factors such as question type or other syntactic or semantic properties; we shall however focus on the simple model for clarity.

Given a number of simplification rules which can be applied to the question to relax constraints on the answer we can construct the following algorithm:

```
Function LogicalRelevance(sentence, question)
Returns the relevance ranking of a sentence in relation to a question or -1 if it can-
not be proven to be relevant
     Attempt to prove that the sentence contains an answer to the question
     If a proof is found return 0
     Else use a breadth-first search (up to depth n) to find one or more simplification
     rules that will prove the answer
     If the sentence can be proven to be an answer through some combination of simpli-
     fication rules, return the number of rules needed
     Else return -1
End Function
```

What we now need is a set of simplification rules which make sense for a human questioner: we cannot simply rely on "pure" logical rules (e.g. "and" elimination starting from the rightmost argument) as this will not necessarily provide a result which a human would consider "relevant": take for example the question

Q: Which English writer wrote the book "Treasure Island"?

Clearly, a simplification such as the following would not be of any use, as it would not provide many sensible answers to Q:

Q_1: Which English writer wrote a book?

While simplifications such as the following could easily provide a useful answer:

Q_2: Which English writer wrote "Treasure Island"?
Q_3: Who wrote "Treasure Island"?

Of course, having removed some constraints from the Q, the new questions Q_2 and Q_3 may or may not provide the desired answer (they may for example provide the author of a play entitled "Treasure Island") but will nevertheless provide *relevant* (useful or interesting) information in relation to Q.

Note that what we propose is not equivalent to an extended knowledge base with rules for inferring answers: we cannot say that the phrase "the book 'Treasure Island"', this is equivalent to "Treasure Island" (i.e. the knowledge base rule "the book X" $\rightarrow$ "X" is patently false as "X" may well be a play or a film with the same title as the book). What we are saying is that if we cannot find a *demonstrably true* answer to a question through the inference rules provided by the knowledge base we can still find a *relevant* answer through the application of a number of simplification rules to the question.

In the following paragraph we provide a method for constructing such simplification rules which will implement the above relevance measure.

4.2. Method

We randomly took 400 questions from the TREC-9, TREC-10 and TREC-11 datasets. The TREC data was used as it is easily available and widely used in QA system research. Answers to the questions were sought within the TREC documents and within generic documents retrieved from the Internet through the Google search engine (we did not limit ourselves to the TREC documents as they did not always provide a sufficient variety of answers). Sentences which contained what was considered an answer by a human judge following the TREC guidelines, but did not necessarily meet all the constraints specified in the questions were kept.

Initial experiments attempted to automatically derive rules in the following manner: transform the question and answer sentence into logical form using a customised parser based on output of the link parser given by [23]; use and-reduction to simplify the question; finally attempt to prove the answer by refutation along the lines of [31] by using the Otter theorem prover [28]. It soon became clear however that the lack of an adequate amount of training data, as well as implementation problems such as parsing errors meant that this approach could only yield very simplistic or specific rules (e.g. removing certain adjectives) and could not be relied upon to produce rules requiring a greater linguistic sophistication or the type of abstraction which is "obvious" to human readers. We therefore sought a manual alternative which would produce a set more general and "intuitive" rules corresponding to what humans would judge as a correct method for judging relevance. It should be noted however that if we were to apply the same method in a restricted domain automation of the method would be much simplified (a parser built for a specific domain would be much more accurate), at least to the point of being able to generate a number of rules which could then be manually evaluated by a human judge.

A human judge determined a series of generalisable simplification rules necessary in order for the answer sentences to be proven. The simplification rules were then applied to the question and the sentences were transformed into logical form using the output of the link parser described above (the resulting logical form was then manually corrected as we were interested in developing a method for calculating logical relevance and were not interested in the accuracy of

the parser or of our logical transformation rules). Finally, the logical from was used to prove the answer by using the Otter theorem prover.

A number of relaxation rules were derived in this manner. Their usefulness was verified using a second subset of 400 randomly chosen TREC questions. As above, answers to the questions were sought within documents from the TREC collection and the Internet and answers which could not be proved by enforcing all the constraints specified in the question were kept; usefulness was then measured as the ability to use the rules to discard constraints given by the question in order for the logical form of the answer to be proven as an answer through Otter.

4.3. Relaxation rules

We shall now give an overview of the rules which were derived, showing how they were used to infer answers which could not otherwise have been proven.

The rules are presented in "plain" English, as opposed to logical notation, simply for a question of ease of understanding and readability. It should be clear, however, that they are trivially transformed into logical form (an obviously necessary step for implementation). For readability the rules have been divided into "syntactic" rules, which rely solely on information about sentence structure and "semantic" rules, which rely on an understanding of the meaning of specific parts of the sentence.

4.3.1. Syntactic rules

Relaxation rule:

```
Relax constraints in the question by removing adjectives and substantiated adjectives
```

Take for example the question:

What is the name of the volcano that destroyed the ancient city of Pompeii?

and the sentence containing the answer:

"... set off for the Italian resort city lying beside the Vesuve volcano which destroyed the city of Pompeii"

Given that the answer sentence does not contain the constraint that Pompeii is an "ancient" city, in order to prove the answer, we must introduce a rule which allows a simplification such as: "What is the name of the volcano that destroyed the *ancient* city of Pompeii?" → "What is the name of the volcano that destroyed the city of Pompeii?", i.e. a rule which allows us to simplify the question by removing the constraint given by the adjective. Examples of questions to which this process may be applied are:

- "What is Australia's *state* flower?", which becomes "What is Australia's flower?"
- "What was the last year that the *Chicago* Cubs won the World Series?" which becomes "What was the last year that the Cubs won the World Series?"
- "The sun is mostly made up of what *two* gasses?", which becomes "The sun is mostly made up of what gasses?"

Relaxation rule:

```
Relax constraints in the question by removing adverbs
```

As in the case of adjectives above, answer sentences often do not contain adverbs which constrain the question. We therefore need a rule to simplify questions by removing constraining adverbs. Examples of questions to which this process may be applied are:

- "What name is horror actor William Henry Pratt *better* known by?", which becomes "What name is horror actor William Henry Pratt known by?"
- "What card game uses *only* 48 cards?", which becomes "What card game uses 48 cards?"

Relaxation rule:

```
Relax constraints in the question by removing verbs in combinations such as "used
in", "employed in", "helps", "aids"
```

For example, given the question:

What is the currency used in China?

And the following answer sentence

"The currency for China is called Renminbi or RMB and is issued by the People's Bank of China"

in order to prove the answer we must remove the verb "use", i.e. we must have a rule which allows the transformation "What is the currency *used* in China" → "what is the currency in China". In other cases we must ensure that the main verb is put into the correct form, as in the following example, where "prevent" must become "prevents":

- "What mineral *helps* prevent osteoporosis?" which becomes "What mineral prevent*s* osteoporosis?"

Another similar rule is:

Relaxation rule:

```
Relax constraints in the question by removing modifier nouns, i.e. nouns which clar-
ify other nouns
```

Often noun phrases in questions contain clarifying information which is assumed as "obvious" in answer documents. As an example, we saw above the question

What company makes Bentley cars?

with the answer sentence:

"Volkswagen has made no secret of its plan to develop a more modest Bentley"

where the author of the answer assumed that the reader was aware that Bentley was a car, without having to spell this information out explicitly. We therefore need a rule which allows the transformation "What company makes Bentley *cars*?" → "What company makes Bentleys?".

Relaxation rule:

```
Relax constraints in the question by simplifying possessive expressions such as "X's
Y" to "Y"
```

As an example, take the question:

What part of the eye continues to grow throughout *a person's* life?

with the answer sentence:

"the lens . . . , a protein-filled disk. . . continues to grow throughout life"

in order for this answer sentence to be proven to be an answer, we must apply a transformation such as the following: "What part of the eye continues to grow throughout *a person's* life?" → "What part of the eye continues to grow throughout life?"

Relaxation rule:

```
Relax constraints in the question by removing relative clauses
```

Relative clauses in questions often specify additional information which is either taken for granted or ignored in answer sentences; a rule therefore needs to be applied to remove these clauses. For example:

Which disciple received 30 pieces of silver *for betraying Jesus*?

There are a number of documents which refer to the disciple being given 30 pieces of silver, without specifying the reason; in order to infer the answer the question must therefore be rephrased as "Which disciple received 30 pieces of silver?"

Relaxation rule:

```
Relax constraints in the question by removing parenthetical remarks
```

Often, constraints which are either obvious or unimportant are found within brackets in a question. For example, given the question:

How fast does an Iguana travel (mph)?

there are a number of documents which contain an answer in kilometres per hour. While the questioner would obviously prefer an answer given in miles, an answer in kilometres would essentially contain the same information: it could not be considered *as relevant* as an answer in miles, but it should nevertheless be considered *relevant*. A rule which allowed a transformation such as the following would provide for this: "How fast does an Iguana travel (*mph*)?" → "How fast does an Iguana travel?"

Relaxation rule:

```
Relax constraints in the question by removing appositive phrases, i.e. sub-sentences
of the form "noun, noun phrase, ..." which describe in more detail a noun, for exam-
ple providing information such as job titles
```

Documents often take for granted that the reader knows enough about the subject not to need "obvious" information to be spelled out: "everyone" knows that Frank Sinatra was a singer and therefore documents often omit to mention this fact. Examples of questions to which this rule applies are:

- "Where did Roger Williams, *pianist*, grow up?" which becomes "Where did Roger Williams grow up?"
- "What year was Ebbets Field, *home of Brooklyn Dodgers*, built?", which becomes "What year was Ebbets Field built?"

4.3.2. Rules based on semantic information

Relaxation rule:

```
Relax constraints in the question by removing time and place constraints, i.e. expres-
sions such as "in Time", "in Place", "at Time", "at Place", "on Time", "on Place"
```

For example, given the question:

Which long Lewis Carroll poem was turned into a musical on the London stage?

We would only be able to prove an answer in the form "Lewis Carroll's poem... was made into a musical" by removing the constraint "on the London stage", i.e. having a rule which allowed the transformation: "Which long Lewis Carroll poem was turned into a musical *on the London stage*"" → "Which long Lewis Carroll poem was turned into a musical?" and then applied the rule above on adjectives to also remove "long", leaving us with "Which Lewis Carroll poem was turned into a musical?" Another example is:

- *In the late 1700s* British convicts were used to populate which colony?", which becomes: "British convicts were used to populate which colony?"

Relaxation rule:

```
Relax constraints in the question by removing nouns indicating works of art such as
"film", "movie", "story", "poem", "novel", "book", "bestseller" when they are fol-
lowed by the title of the work of art, i.e. in expressions such as "the book Title"
```

Often questions specify that titles are titles of books, films, TV series etc., while answer sentences usually omit such information. Take for example the question:

What was the name of the dog in the Thin Man movies?

and the answer sentence:

"Asta in the "Thin Man" (1934): Asta, the wire haired terrier ... was ... fashionably art deco..."

In order to prove the answer we need a rule which allows the transformation "What was the name of the dog in the Thin Man *movie*" → "What was the name of the dog in the Thin Man?" Other examples are:

- "Who directed *the film* "Fail Safe"?", which becomes "Who directed "Fail Safe"?"
- "When did *the story of* Romeo and Juliet take place?", which becomes "When did "Romeo and Juliet" take place?"

Relaxation rule:

```
Relax constraints in the question by removing specifications of place names in combi-
nations such as "District, Region"
```

Often questions specify the exact location of towns, cities, regions etc. by adding the country, region, state, etc. in which they are found, e.g. by stating "Rome, Italy", or "Stanford, California". Most documents, however, do not use such specifications (containing, in the examples above, simply a reference to "Rome" or "Stanford") and in order for the answer to be proven rules must allow the removal of these specifications. For example,

What is the current population of Bombay, *India*?

There are numerous documents which answer the question by giving details of Bombay without explicitly mentioning that it is in India: in order to be answered, the question becomes "What is the current population of Bombay?"

Relaxation rule:

```
Relax constraints in the question by removing first names, middle names, surnames or
titles (both preceding and following names) where these are in conjunction with other
proper names, as in the combination "Title Name Surname"
```

At times questions contain both surname and name of a given person, but documents contain only a surname (and vice versa); the same problem often occurs with titles (President Bush being referred to as Bush, for example). Examples of questions where such a rule would be useful are:

- "What was *Frank* Sinatra's nickname?", which becomes "What was Sinatra's nickname?"
- "What year was *president* Kennedy killed?", which becomes "What year was Kennedy killed?"

A related rule is:

Relaxation rule:

```
Relax constraints in the question by simplifying specifications such as "Y of X" to
"X" where Y is a more generic term for X, i.e. a hypernym of X
```

Often noun phrases containing specifications (e.g. town of X, science of Y) can be summarised without loss of information by the noun which is being specified (X and Y in the examples above). As an example, take question:

What name is given to *the science of* map making?

There are a number of documents about map making which do not mention explicitly that map making is a science; by applying the rule above, the question can be related to the answer sentences by being simplified to "What name is given to map making?"

Relaxation rule:

```
Relax constraints in the question by removing platitudes
```

Questions are often formulated as commands/requests, of the form "Tell me…" or "Please could you let me know…". An example is:

- "*Tell me* where the DuPont company is located", which should become "Where is the DuPont company located?"

Relaxation rule:

```
Relax constraints in the question by transforming complex idioms into simple senten-
ces, specifically transforming sentences of the form "X is home to Y" into "Y is in
X"; "the name of X is Y" into "X is Y"; "X is the person who did Y" into "X did Y";
"Who is the man who ..." into "Who ..."; "What was the man's name who..." into "Who";
"What was the place where..." into "where".
```

Often questions are formulated in a manner which is more complex than necessary, with answer sentences being expressed with much simpler language. It is therefore necessary to rephrase these questions to successfully prove the answer sentences. Examples are:

- "Which Italian city *is home to* the cathedral of Santa Maria del Fiore", which should become "In which Italian city is the cathedral of Santa Maria del Fiore?"
- "*What was the name of* the first Russian astronaut to do a spacewalk", which should be rephrased as "Who was the first Russian astronaut to do a spacewalk?"

- "*What was the man's name* who was killed in a duel with Aaron Burr?", which becomes "Who was killed in a duel with Aaron Burr?"

4.4. Implementation

We implemented the `LogicalRelevance` algorithm and rules above in order to prove their applicability in practice; what follows is therefore more the description of a proof of concept than a full implementation (we were not interested in questions of efficiency and speed as the engineering aspect of implementation was not the focus of our efforts). Implementation of the algorithm made use of both word-sequence matching and more complex rules using, for example, part-of-speech tagging (e.g. to recognise adjectives) and phrase structure information. A subset of 400 of the TREC-11, TREC-10 and TREC-9 questions (equally distributed), different from the questions used for the analysis, was then employed to evaluate the usefulness of the relaxation rules. Usefulness was taken to be the ability to use the rules to discard constraints given by the question and "prove" answers which could not be proved otherwise. In order to test this ability, answer sentences which provided an answer to the question were sought within the TREC document collection and on documents retrieved through the Google search engine. Answer sentences were then manually examined and only sentences which contained the answer but which could not be proven immediately were kept; furthermore the necessary background information for proofs was provided manually (we were testing the relaxation rules, not the quality of our knowledge base!). The use of the Internet to find answer sentences ensured we had a good number of sentences which met all the necessary criteria. The relaxation rules were then applied to the questions in order to determine whether the application of these rules would enable the theorem prover to prove the answerhood of the remaining answer sentences. Coreference resolution and background knowledge was provided manually in order to standardise the evaluation. While, as shown above, the use of background knowledge should contribute to the evaluation of the relevance of an answer, for simplicity we ignored this problem, focusing our attention on the constraint relaxation rules.

Using the relaxation rules we found additional relevant answer sentences to 28% of TREC-11 questions, 11% of TREC-10 questions and 19% of TREC-9 questions (the rest of the questions were already in minimal form, for example definition questions such as "Who was Galilei?" and could not be reduced further without serious loss of information). The examined questions did not contain any "surprises" in the form of constraints which required additional rules in order for the answer sentences to be proven.

5. Conclusion

We have proposed a method of determining the appropriateness of an answer to a question through a proof of logical relevance rather than a more traditional logical proof of truth. We defined logical *relevance* as the idea that answers should not be considered as absolutely true or false in relation to a question, but should be considered true in a sliding scale of aptness. We showed how the concept of logical relevance differs from both traditional logic approaches to question answering and from related concepts such as relevance logics, enabling us to reason rigorously about the appropriateness of an answer even in cases where the sources we are getting answers from are incomplete or inconsistent or contain errors. Logical relevance can be applied in practice to improve open-domain QA systems by moving beyond techniques based on information retrieval and extraction and traditional logical proof: by introducing the notion of logical *relevance*, we overcome the limitations of a "either right or wrong" approach which has limited applicability in practical settings where documents may lack important information, or contain contradictions and errors such as misspellings or factual inaccuracies. A method was presented for the calculation of logical relevance for question answering through the use of relaxation rules that gradually drop the logical constraints on the answer given in the question: relevance is defined in terms of the effort required to prove an answer by relaxing constraints. Finally we showed an example of how the method could be applied in practice by deriving a number of relaxation rules which are applicable to realistic (as they are derived from logs of questions which users have actually asked on sites such as Google) questions such as those given in the TREC evaluation framework and we then showed how these rules could be implemented to find answers which would otherwise have been ignored.

References

[1] W. Ackermann, Begruendung einer strengen Implikation, J. Symbolic Logic 21 (1956) 113–128.
[2] A.R. Anderson, N.D. Belnap Jr., Entailment—The Logic of Relevance and Necessity, Princeton University Press, 1975.
[3] K.R. Apt, Principles of Constraint Programming, Cambridge University Press, 2003.
[4] N.D. Belnap, How a computer should think, in: Contemporary Aspects of Philosophy: Proceedings of the Oxford International Symposium, Oxford, 1975, pp. 30–56.
[5] N.D. Belnap, T.B. Steel, The Logic of Questions and Answers, Yale University Press, 1976.
[6] Bobrow, et al., GUS, a frame driven dialog system, Artificial Intelligence 8 (1977) 155–173.
[7] A. Borgida, D.L. McGuinness, Asking queries about frames, in: Proceedings of KR-96, Cambridge, MA, 1996.
[8] D.T. Burhans, A question answering interpretation of resolution refutation, PhD Dissertation, State University of New York at Buffalo, Department of Computer Science and Engineering, 2002.
[9] J. Cheng, Logical tool of knowledge engineering: using entailment logic rather than mathematical logic, in: Proceedings of the 19th Annual Conference on Computer Science, San Antonio, TX, 1991.
[10] L. Cholvy, R. Demolombe, Querying a rule base, in: Proceedings of the First International Workshop on Expert Database Systems, 1986.
[11] L. Cholvy, Answering queries addressed to a rule base, Revue d'Intelligence Artificielle 4 (1) (1990).
[12] A. Church, The weak theory of implication, in: Menne-Wilhelmy-Angsil (Ed.), Kontrolliertes Denken, Untersuchungen zum Logikkalkuel und der Logik der Einzelwissenschaften, Munich, 1951, pp. 22–37.
[13] M.G. Dyer, In-Depth Understanding, MIT Press, 1983.
[14] B.F. Green, et al., BASEBALL: an automatic question answerer, in: Proceedings of the Western Joint Computer Conference, 1961.
[15] C. Green, B. Raphael, The use of theorem proving techniques in question answering systems, in: Blue (Ed.), Proceedings of the 23rd National Conference of the Association for Computing Machinery, Princeton, NJ, 1968.
[16] S.J. Green, Automatically generating hypertext by computing semantic similarity, Technical Report n. 366, University of Toronto, 1997.
[17] J. Groenendijk, M. Stokhof, Questions, in: J. van Benthem, A. Meulen, Handbook of Logic and Language, Amsterdam, 1996, pp. 1055–1124.
[18] J. Groenendijk, Questions and answers: Semantics and logic, in: Questions and Answers: Theoretical and Applied Perspectives, 2nd CoLogNET-ElsNET Symposium, 2003.
[19] C. Hamblin, Questions in montague English, Foundations of Language 10 (1973).
[20] S. Harabagiu, et al., Answer mining by combining extraction techniques with abductive reasoning, in: Proceedings of TREC-12, NIST, 2004.
[21] D. Harrah, The logic of questions, in: D. Gabbay, F. Guenthner (Eds.), Handbook of Philosophical Logic, vol. II, Reidel, 1984, pp. 715–764.
[22] K. Humphreys, et al., University of Sheffield TREC-8 Q & A system, in: Proceedings of TREC-8, NIST, 2000.
[23] J. Lafferty, The link parser API, http://www.link.cs.cmu.edu/link/api/index.html, 2000, last modified August 25 2000 (last accessed 17 October 2003).
[24] W.G. Lehnert, The Process of Question Answering, New Jersey, 1978.
[25] B. Lin, The motivation of constructing entailment logic, in: Proceedings of the 8th International Conference of Logic, Methodology and Philosophy of Science, vol. 5, Moscow, 1987.
[26] D. Lin, P. Pantel, Discovery of inference rules for question answering, Natural Language Engineering 7 (4) (2001) 343–360.
[27] D. Luckham, N. Nilsson, Extracting information from resolution proof trees, Artificial Intelligence 2 (1971) 27–54.
[28] W.W. McCune, Otter 3.0 Reference Manual and Guide, Argonne National Laboratories, 1984.
[29] C. Meghini, U. Straccia, A relevance terminological logic for information retrieval, in: Proceedings of the 15th Annual ACM SIGIR Conference on Research and Development in Information Retrieval, 1996.
[30] S.-K. Moh, The deduction theorems and two new logical systems, in: Methods, vol. 2, 1950, pp. 56–75.
[31] D. Moldovan, et al., LCC tools for question answering, in: Proceedings of TREC-11, NIST, 2003.
[32] E.J. Nelson, On three logical principles in intension, The Monist 43 (1933).
[33] M.L. Prior, A.N. Prior, Erotetic logic, The Philosophical Review 64 (1) (1955).
[34] R. Reiter, Deductive question answering in relational databases, in: H. Gallaire, J. Minker (Eds.), Logic and Databases, New York, 1978, pp. 149–177.
[35] R.C. Schank, R.P. Abelson, Scripts, Plans, Goals and Understanding, New Jersey, 1977.
[36] D. Sperber, D. Wilson, Relevance: Communication and Cognition, Blackwell, Oxford and Harvard University Press, Cambridge, MA, 1986.
[37] E. Voorhees, D.M. Tice, The TREC-8 question answering track evaluation, in: Proceedings of the 8th Text Retrieval Conference, NIST, 2000.
[38] E. Voorhees, Overview of the TREC 2003 question answering track, in: Proceedings of TREC-12, NIST, 2004.
[39] T. Winograd, Understanding Natural Language, Academic Press, New York, 1972.