Question Answering using a large NLP System

David Elworthy
Microsoft Research Limited, St. George House, 1 Guildhall Street, Cambridge CB2 3NH, UK

1 Introduction

There is a separate report in this volume on the Microsoft Research Cambridge participation in the Filtering and Query tracks (Robertson and Walker 2001).

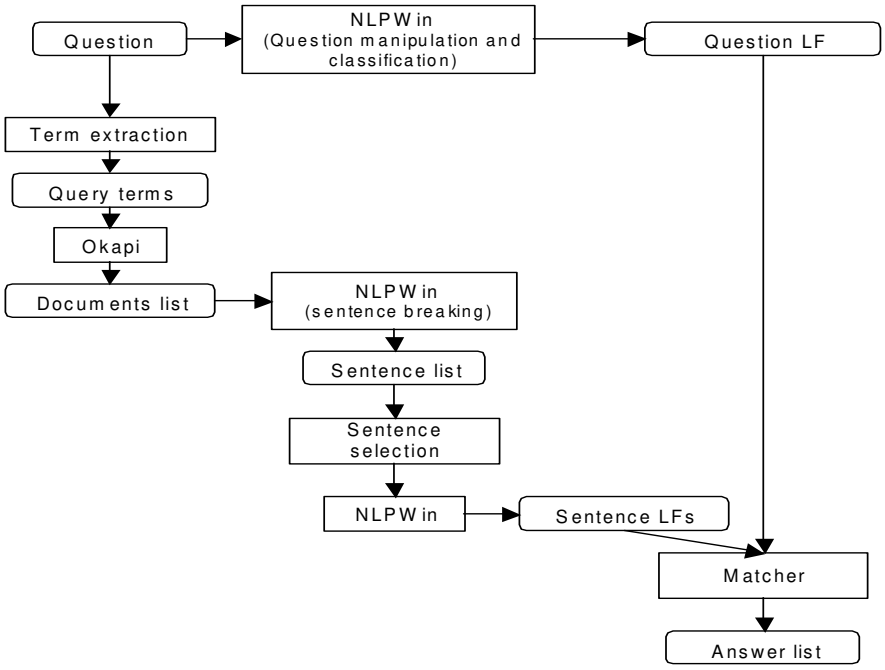
The Microsoft Research question-answering system for TREC-9 was based on a combination of the Okapi retrieval engine, Microsoft's natural language processing system (NLPWin), and a module for matching logical forms. There is no recent published account of NLPWin, although a description of its predecessor can be found in Jensen et al. (1993). NLPWin accepts sentences and delivers a detailed syntactic analysis, together with a logical form (LF) representing an abstraction of the meaning. The original goal was to construct a framework for complex inferencing between the logical forms for questions and sentences from documents. Many answers can be found with trivial inference schemas. For example, the TREC-8 question *What is the brightest star visible from Earth?* could be answered from a sentence containing ... *Sirius, the brightest star visible from Earth* ...by noting that all of the content words from the question are matched, and stand in the same relationships in the question and in the answer, and that the term *Sirius* is equivalent to the answer's counterpart of the head term in the question, *star*. The goal of using inferencing over logical forms was to allow for more complex cases, as in *Who wrote the play ``Hamlet''?* which should not be answered using ... *Zeferelli's film of ``Hamlet''* since a film is not a play. The idea of using inferencing for question-answering is not new. It can be found in systems from the 1970s for story understanding (Lehnert, 1978) and database querying (Bolc, 1980), and in more recent work for questions over computer system documentation (Aliod, 1998).

Time pressure forced this idea to be dropped (work on the system did not start until March 2000), and instead a simpler scheme was adopted, still using LFs from NLPWin. The main observation behind the actual system is that the answer often appears in close proximity to the content terms from the question within the LF, as in the Sirius example above. Consequently, we can try to find answers by identifying candidate nodes in the LF and then using a measure of the proximity. For some kinds of question, such as *when* questions, there is a clear way of identifying candidate answers; for others, such as *what*, it is much harder.

In the following section, we will look at the architecture of the system, and describe the main question types and how they are handled. The evaluation follows in section 3. The results turned out to be relatively poor. Interestingly, there is a very large difference between the results on the TREC-8 test set and the TREC-9 questions, and we will use a fine grained evaluation to examine why.

2 Method

The architecture of the system is shown below. The question is analysed by NLPWin to produce a logical form, and in addition a set of query terms is extracted from it. The query terms will normally contain all of the words of the question less the question word itself (*what*, *who*, *how*, etc.) and a few other stop words. The query terms are used by the Okapi IR engine with BM25 weighting to produce a list of documents. The documents are then segmented into sentences. This stage uses NLPWin, although without using its detailed linguistic analysis capabilities. The resulting list of sentences is ordered by the number of terms from the question they contained, and processed again by NLPWin, this time producing the full linguistic analysis. A cutoff on the number of sentences is used to control the processing time, since a full NLP analysis can be quite time-consuming. The resulting logical forms are compared with the question's logical form to produce a ranked list of answers with scores.



An example of a logical form appears below, for the question *What is the brightest star visible from Earth?*

```
be1 (+Pres +WhQ +L1)
|_Dsub---star1 (+Def +Pers3 +Sing +Conc +Count)
         |_Attrib+-bright1 (+Supr +PosSupr +A0)
                 +-visible1 (+PostNom +E0)
                   |_from---Earth1 (+Pers3 +Sing +PrprN +Conc +Count +Mass)
|_Dnom---what1 (+Wh +Pers3 +Sing)
```

The nodes of the graph (in bold) generally represent the content terms of the analysed sentence, although a few nodes (such as *be1* and *what1*) are more of a structural nature. The nodes are connected by directed relations such as *Dsub* and *Attrib*. Each node can have a number of binary properties, such as *+WhQ*. What we show here is a simplified version of the LF, and the full internal representation contains more information. There is a very large number of different relation types and properties, and we will not attempt to list them here.

2.1 Question manipulation and classification

The aim of the question manipulation stage is to simplify the logical form of questions in order to make it easier to classify them, and to label certain terms in the question as formal and hence not expected to match a term in a candidate answer.

The majority of the manipulations look for a specific question word, attached to a specific relation. For example, a question of the form *Who is X* receives a logical form in which *X* has an *Equiv* relation to a node for *who*. In such cases, we simply delete the relation and who and add an annotation to the top node of *X* which indicates that we are looking for an answer to a *Who* question over objects with the property of being *X*. Similar principles apply to many of the question types. The relation may be other than *Equiv*; for example in *where* and *when* questions, the relations *Locn* and *Time* are used. A second case which occurs frequently is logical forms in which the topmost node is *be*, usually with a single child, or with one child which is a Wh-word and one which is a content node. In such cases, we remove the *be* node, and in the latter case move the Wh-word's properties to the other child.

There are some common subjects for *what* questions, such as *what country...*, *what year...*. In these cases, we remove the whole what-phrase and mark the remaining top level node with a special property to indicate that the question should be answered with a restriction as to the answer type. This is only done when the subject corresponds to a property which NLPWin marks in the LF, such as *Cntry* for country. NLPWin derives this information from its lexical resources.

Tthere are a number of other question manipulations on broadly similar principles. After the manipulation, we then assign each question to a category, using the question word (often now discarded and encoded as a property) and the structural configuration. An example of a distinction made using the structure comes with *who* questions, where we distinguish questions asking about identity, as in *Who is the leader of India?* from questions about a role of a predicate, as in *Who invented the paper clip?*. A full list of the question types appears in the appendix. A few questions are left as having *Unknown* type, and questions with an incomplete parse are assigned the type *Bad*.

2.2 Matching

Matching proceeds by selecting and scoring possible answers guided by the question type, and then by extracting the phrase to return as the result. Answer selection is the most complex part of the matching process, and we return to it in a moment. The result of answer selection is a node in the logical form of the answer sentence and a score. To extract the answer, we look up the syntactic node associated with the LF node, and take the portion of the original sentence which led to it. This process is imperfect, and was intended as a quick way of recovering the answer. It tends to give phrases which span more words than necessary. For example, the LF node may describe an entity, but the corresponding syntactic node is a prepositional phrase, as the preposition is absorbed into the structure of the LF, resulting in an answer such as *by X* or *to X* rather than simply *X*. If the resulting phrase is longer than the maximum allowed width (50 bytes or 250 bytes), then words are removed from the ends of the phrase until it is short enough. By preference, words which appeared in the question are removed over ones which were not, and otherwise the process alternates removing words from the left and right hand ends of the phrase.

2.3 Answer selection

Answer selection is the heart of the matching algorithm. The rules used in the TREC-9 test are rather ad hoc; some of them are reasonably well principled, while others are hacks which seemed to work more often than any alternative. The principles we use to identify candidate answers nodes include the following:

Node properties
Node properties are used when answer nodes usually have clear LF properties, but where the relationship with the query terms can vary. *Who*, *HowMany* and *HowMuch* questions are good examples, although we will see later that there is a risk involved in treating *Who* questions this way. The node properties are flags assigned by NLPWin usually using information stored in the lexicon. Node properties are used in three stages: firstly, we look for nodes which have one or more of a set of required properties; then we remove any which have certain properties which might indicate we have made the wrong choice as a result of over-generalisation; and finally, we look for preference properties whose omission indicates that the score assigned to the answer should be lowered. For example, in the case of *Who* questions, the only required property is *PrprN* (proper name), nodes are removed if they have properties such as *Tme* (time), *Titl* (title) and *Cntry* (country), and the score is lowered if node does not have one of the properties *Anim* (animate), *Humn* (human) or *Nme* (name).

Relation targets
Some answers can be found be looking for nodes which are the target of a given relation type, using proximity to determine whether the node is likely to be related to the question terms. Examples are *Where* and *When* questions, answers to which are often found as the target of *Locn* and *Time* relation. For *When* questions in particular, the answer time expression may appear on a different argument of a verb to the question term itself, or on a modifier of the question term.

Node-to-node relations
Node-to-node relations come closest to really using the structure of the LF. The idea here is to look for a node which lies at one end of a relation, the other end of which is a question term. The case where this is used most extensively is in questions of the form *What is X*. Answers are typically found as standing in an *Equiv*, *Mod* or *Attrib* relation to *X* in the answer, as in the logical forms generated from phrases such as (the answer is highlighted):

Head Start is **a preschool program**
Berlin is the capital of Germany
Sirius, the brightest star visible from Earth

The first two of these illustrate *Equiv* (equivalent) relations, and show that the answer can be either the source or the target of the relation in this case. The third example is a *Mod* (modifier) relation. Some relations may signal the answer better than others; for example *Equiv* tends to indicate the answer more strongly than *Mod* or *Attrib* (attribute). The term which stands at the other end of the relation from the answer, i.e. the term from the question, may or may not be the head of the question. Thus if the question is *What is the capital of Germany?*, the head of the question is *capital*, but we are as likely to find the answer related to the term *Germany*. Simple examples like this could be handled by specialised rules, for example manipulations of the question's LF, but this cannot always be done reliably. One case where we definitely do want the relation to be to a specific question word is questions about a specific role of a predicate. Thus, in *Who won the SuperBowl in 1968?*, the answer should be in the subject role of the verb *win*.

Combinations
Some of the questions types use more than one of these techniques, and select the one which gave the best score. An example is *WhoRole* questions (which ask who performed a particular role of an action), which look for words with the same properties as *Who* questions, and also look for entities in a particular role of a verb, as for *WhRole* and *WhatRole* questions (node-to-node relation type of answers).

2.4 Proximity scoring

To assign a score to the nodes identified in answer selection, we use a simple measure based on how close the candidate answer is to significant terms from the question. The proximity measure marks each term in an answer sentence which matches a term from the question, and then sees how far this term is from the candidate answer, measured as the number of relations that have to be traversed in the logical form. The idea of proximity is to provide an approximation to matching the LFs, in that if an answer were closely related to the matched question terms, then it would have a small proximity, whereas if it had an indirect relation, the proximity would be lower. There is little linguistic basis for this approach, and the idea was really to obtain a baseline for performance based on a simple and easily implemented technique within the timescales of the TREC-9 exercise. The overall proximity is calculated by summing these distances for each of the question terms, taking its reciprocal, and weighting it by the logarithm of the total number of the matched question terms plus one. The latter factor is simply a way of taking into account what proportion of the question terms were matched. The logarithm is used just to weaken the factor; although this is ad hoc, it seems to give a better performance that using just the proportion of the query terms or no factor at all. An obvious enhancement to this process might be to weight question terms by importance, for example giving lower weight to question terms which are more deeply buried in the logical form.

3 Evaluation

3.1 The TREC-8 test set

The system was developed and tested using the questions and assessments from the TREC-8 evaluation. For an initial stage of evaluation, the retrieval stage was run in isolation, and the documents were examined to see if a correct answer appeared anywhere in them. This provides an upper bound on performance, by finding the best score which could be achieved if a perfect answer identification and extraction component were available. The evaluation also allowed tuning of the number of documents returned by Okapi: too few, and a correct answer might be missed; too many, and the processing time of the later stages would get out of hand. A document cutoff of 100 was selected for on this basis, which resulted in 92% of the questions retrieving a document which contained a correct answer. Larger cutoffs produced only a small further increase in this score. A variant of the experiment was run in which the term list for the retrieval was derived from the logical form, rather than by just taking the question and using a stemmed and stopped wordlist. The idea was to see if the segmentation and morphological analysis provided by NLPWin would help the retrieval stage. The scores were in general very slightly less than those above, showing that there is no clear advantage to using NLPWin as a pre-processor to the retrieval stage.

The performance for the overall system was calculated using mean reciprocal rank. Three scores were calculated: for the 50-byte and 250-byte limited runs, and for a run in the answer could be of any length, provided it lay within a single sentence. The results were as follows:

Run	50-byte	250-byte	Sentence
MRR	0.357	0.425	0.446

The first observation is that the best score, for the unlimited run, is significantly less than the maximum that could potentially be achieved with perfect answer selection (0.92, from the retrieval stage experiment). Secondly, the score does decrease with the window size, indicating that there is also scope for improvement in answer extraction. Compared to the official TREC-8, the 50 byte run would have come roughly 3rd out of 20 (or 21 including this run), and the 250 byte run about 10th out of 25.

3.2 TREC-9 test

The TREC-9 test consisted of 682 questions, including variant forms. The official evaluation results were:

Run	50-byte, strict	50-byte lenient	250-byte, strict	250-byte, lenient
MRR	0.196	0.203	0.264	0.274

Clearly, these are well below what we saw on the TREC-8 data. So what went wrong? In order to try to understand why, we look at the results for the separate question types in greater detail. In the table below, we list, for each question type:

- the number of questions of each type in the TREC-8 and TREC-9 test sets
- the MRR on that type

- the relative contribution of the class to the overall results
- the changes in MRR and relative contribution.

The relative contribution of a question type is the MRR for the type multiplied by the proportion of the questions which have that type. For example, if a type had a MRR of 0.5, and one quarter of all the questions had that type, the relative contribution would 0.5x0.25 = 0.125. The difference in MRR gives an indication in the abstract of the how well a question types was handled. If there is a large change, it would suggest that the rules for the type are too sensitive to the particular questions seen in the TREC-8 data. The change in relative contribution gives an indication of how much this matters, and therefore where efforts should be focussed to alter the system's performance. There may be more benefit in correcting a small decrease in MRR on a class with many questions as opposed to a large decrease on a class with only one or two. Some question types are handled identically, and we therefore list them both as the separate types and combined. The table is ordered by the change in the relative contribution, and the TREC-9 results are based on the 250-byte lenient judgements.

Question type	**TREC-8**			**TREC-9**			**Change**	
	#	**MRR**	**Rel.Cont.**	**#**	**MRR**	**Rel.Cont**	**MRR**	**Rel.Cont**
Unhandled	11	0.26	0.014	84	0.36	0.044	0.10	0.030
Unknown	2	1.0	0.010	38	0.37	0.020	-0.63	0.010
Bad	5	0.10	0.0025	6	0.50	0.0044	0.40	0.0019
WhPrep	3	0.11	0.0017	35	0.35	0.018	0.24	0.016
HowDo	1	0	0	5	0.20	0.0015	0.20	0.0015
WhoRole	20	0.29	0.029	62	0.36	0.032	0.073	0.0029
HowLong	1	0	0	1	0	0	0	0
HowManyTimes	1	0	0	0	0	0	0	0
WhatMeas	1	1.0	0.0050	8	0.29	0.0034	-0.71	-0.0016
When	18	0.38	0.034	47	0.45	0.031	0.070	-0.0032
Why	2	0.50	0.0050	2	0.5	0.0015	0	-0.0035
HowFar	1	1.0	0.0050	1	0	0	-1.0	-0.0050
WhatTime	6	0.29	0.0089	13	0.12	0.0022	-0.18	-0.0066
Where	21	0.44	0.046	71	0.37	0.038	-0.071	-0.0078
HowMuch	3	0.67	0.010	4	0.31	0.0018	-0.35	-0.0082
HowMany	15	0.28	0.022	26	0.29	0.011	0.0032	-0.010
HowProp	5	0.60	0.015	10	0.10	0.0015	-0.50	-0.014
What+	39	0.52	0.10	211	0.20	0.063	-0.22	-0.031
What	38	0.53	0.10	195	0.21	0.060	-0.32	-0.041
WhEquiv	1	0	0	16	0.15	0.0034	0.15	0.0034
WhRole+	28	0.38	0.053	92	0.16	0.021	-0.31	-0.038
WhRole	22	0.34	0.038	56	0.12	0.0095	-0.22	-0.028
WhatRole	6	0.50	0.015	36	0.23	0.012	-0.27	-0.0031
Who	28	0.55	0.078	52	0.30	0.023	-0.26	-0.055

It follows to look in more detail at what is going on in some of the more significant changes. Three classes in particular appear worth investigating on the basis of the change in relative contribution: *Who*, *WhRole+* and *What+*.

In the case of *Who* questions, the problem appears to be that some of the questions aim to identify an entity, while others aim to elicit a description of an individual. The two types are illustrated by

Who is the richest person in the world? (entity)
Who is Desmond Tutu? (description)

The TREC-8 test set included only entity questions, and the rules for answering *Who* questions did not allow for the description case. This could be corrected by adding a test to see if the question term already has the properties we look for in the entity case (*PrprN*, etc.), and if so using the same approach as *What* questions such as looking at *Equiv* and *Mod* relations.

The problem with *What* questions appears to be that many more of the questions have the form *What is the X of Y?* than the original set, for example,

What was the name of the first Russian astronaut to do a spacewalk?
What is the population of the Bahamas?

These are only handled well for a small number of predefined cases for the category condition *X*, such as *city*, *name*, and *kind*. To improve this class, we would need to have a set of additional rules which encode information about the category condition, for example that a population is usually expressed as a number.

A similar remark applies to the *WhRole* questions, many of which have the form *Which X does Y?*, such as

What sport do the Cleveland Cavaliers play?

Again, a few special cases are handled already, but the inclusion of some additional ones would help to select correct answers more reliably. One issue to be considered here is what conditions should have special rules and what should not. It is (perhaps) reasonable to have a list of sports for the above case, but what about

What soft drink would provide me with the biggest intake of caffeine?

The answer here appears to be some wider encoding of world knowledge. An interesting point emerges. If we are encoding world knowledge, should we try to encode all knowledge in the documents into some knowledge representation structure, and answer questions directly against it? This appears to be the thought process behind using MindNet (Richardson et al., 1998) in which dictionaries and encyclopedias are analysed and their logical forms merged into a single large structure, and it was also the approach used in the question-answering systems of the 1970s (Lehnert (1978), for example). The difficulty arises when the sources of the knowledge become more diverse and less coherent than those behind MindNet, or the Unix man pages used in ExtrAns (Aliod, 1998). There may be opinions, interpretations, inconsistencies, and simple errors in the document collection. An important challenge for future work may therefore be looking at how to build a system which merges definitive, pre-encoded knowledge, and ad-hoc documents of unknown reliability.

Appendix: Question types

These are the different types of questions which were used, with their frequencies in the TREC-8 and TREC-9 test sets.

HowDo (TREC-8: 1 TREC-9: 5) *How did Bob Marley die?*

HowFar (TREC-8: 1 TREC-9: 1) *How far away is the moon?*

HowLong (TREC-8: 1 TREC-9: 1) *How long do hermit crabs live?*

HowMany (TREC-8: 15 TREC-9: 27) *How many dogs pull a sled in the Iditarod?*

HowManyTimes (TREC-8: 1 TREC-9: 0)

How many times was pitcher, Warren Spahn, a 20-game winner in his 21 major league seasons?

HowMuch (TREC-8: 3 TREC-9: 4) *How much folic acid should an expectant mother get daily?*

HowProp (TREC-8: 5 TREC-9: 10) *How tall is the giraffe?*

WhEquiv (TREC-8: 1 TREC-9: 16) *What language is mostly spoken in Brazil?*

WhPrep (TREC-8: 3 TREC-9: 35) *What is Francis Scott Key best known for?*

WhRole (TREC-8: 22 TREC-9: 56) *What state has the most Indians?*

What (TREC-8: 38 TREC-9: 200)

What was the name of the first Russian astronaut to do a spacewalk?

What is Head Start?

Name a flying mammal.

WhatMeas (TREC-8: 1 TREC-9: 8)

What type of bridge is the Golden Gate Bridge?

What kind of animal was Winnie the Pooh?

WhatRole (TREC-8: 6 TREC-9: 36) *What does laser stand for?*

WhatTime (TREC-8: 6 TREC-9: 13) *What year did Montana become a state?*

When (TREC-8: 18 TREC-9: 48) *When did Vesuvius last erupt?*

Where (TREC-8: 21 TREC-9: 71) *Where is Belize located?*

Who (TREC-8: 28 TREC-9: 53) *Who is the leader of India?*

WhoRole (TREC-8: 20 TREC-9: 62) *Who invented the electric guitar?*

Why (TREC-8: 2 TREC-9: 2) *Why can't ostriches fly?*

Bad (TREC-8: 5 TREC-9: 6)

Questions which received no analysis from NLPWin, or a fragmentary one.

Unknown (TREC-8: 2 TREC-9: 39)

Other questions, not covered by any of the above classes.

Bibliography

Robertson, S. and Walker, S. (2001). *Microsoft Cambridge at TREC-9: Filtering track.* In these proceedings.

Bolc, Leonard (ed.) (1980). *Natural language question answering systems*. Macmillan.

Lehnert, Wendy G. (1978). *The process of question-answering: a computer simulation of cognition*. Erlbaum.

Molla Aliod, Diego, Jawad Berri and Michael Hess (2000). A real-world implementation of answer extraction. *Proc. of 9th International Conference and Workshop on Database and Expert Systems. Workshop "Natural Language and Information Systems" (NLIS'98).*

Stephen D. Richardson, William B. Dolan and Lucy Vanderwende (1998). MindNet: Acquiring and Structuring Semantic Information from Text. In *Proceedings of COLING*.