Epistemic reasoning, logic programming, and the interpretation of questions

by

Manny Rayner and Sverker Janson

SICS R87002
Research Report
ISSN 0283-3638

Epistemic reasoning, logic programming, and the interpretation of questions

Manny Rayner   Sverker Janson

Abstract

Reasons are given to support the claim that strong connections exist between linguistic theories on question semantics and work within AI and logic programming. Various ways of assigning denotations to questions are compared, and it is argued that the earlier, "naive" version is to be prefered to more recent developments. It is shown that it is possible to use the chosen denotation to give a model-theoretic semantics for the concept of "knowing what", from which a relationship between "knowing what" and "knowing" can provably be derived. An application to logic programming is described, which allows formal reasoning about what a logic database "knows", this being in a sense a generalization of the Closed World Assumption. Finally, a "knows-what" meta-interpreter in Prolog is demonstrated, and proved to be sound and complete for a certain class of database programs.

Keywords: Natural Language, Questions, Denotational Semantics, Modal Logic, Epistemic Reasoning, Closed World Assumption, Databases

1. Introduction

The purpose of this paper is threefold. Firstly, and most generally, we want to suggest that work done by theoretical linguists on the interpretation of questions is of great relevance to many other disciplines, among them AI and logic programming. We think that this is an important thing to say since, as far as we can make out, many of the people active in these fields are unaware even of the existence of the body of research we will be refering to.

Secondly, we wish to express our doubts about what we think has been an unfortunate turning in the linguistic line of development. We will give reasons to support our claim that current approaches to question semantics inspired by Kartunnen's extremely influential 1977 paper are not an improvement on the earlier ideas; in fact, it appears that in many cases Kartunnen's analysis makes erroneous predictions, while the earlier and more naive one is correct. We also point out some interesting corrollaries following from the naive analysis, using a simple but powerful algebraic trick which appears so far to have gone unnoticed.

Thirdly, we demonstrate a concrete application of question semantics to logic programming. We use our framework to give a model-theoretic definition of "knowing what", and show that it is possible to reason formally about what a Prolog database "knows"; moreover, that given certain fairly natural assumptions about the nature of the database it is also possible to build an efficient interpreter which is capable of determining whether a database "knows" the answer to a query. This has interesting consequences, not least among them being the possibility of replacing the notorious "closed world assumption" by a more intelligent alternative.

The organization of the paper is as follows. In section 2, we try to set question semantics in context, and list our objections to the Kartunnen approach. Section 3 is devoted to the naive approach, and we give our definition of "knows what" together with examples of its use. In section 4, we look specifically at the use of the "knows what" concept in logic programming, and explain the theoretical principles behind the interpreter, which is presented in the appendix. Section 5 contains our conclusions, together with some ideas about further directions of research.

2. Questions and their semantics

We begin by noting that questions in most languages can occur in two ways. Firstly, there are "direct" questions, like *What is the time?* or *What is his mother's maiden name?* To many people, this is what the word "question" means. However, questions also occur in another context, which is at least as important, namely as the complements to so-called *question-embedding verbs* like "know", "say" and "find out". Thus the italicized portions of sentences 1) to 4) are reasonably regarded as questions.

1) John knows *where Mary lives.*
2) Mary told John *when the train was due.*
3) It says on the form *which boxes you are supposed to fill in.*
4) Mary wants to find out *what he bought from John.*

The problem of formalizing statements like these has been considered by linguists ([Egli 73], [Kartunnen 77], [Hauser 78], [Hirschbühler 79], [Bennett 79], [Engdahl 86]), AI researchers ([McCarthy 75], [Moore 80], [Maida 82], [Konolige 82], [Haas 86]), philosophers ([Hintikka 76]), computer scientists ([Main 83]) and even roboticists ([Rosenschein 85]). We will now consider their proposals and attempt to draw some general conclusions.

If one is working in some kind of epistemic logic which includes a "knows" operator, then the simplest solution is to try and ignore the interpretation of questions altogether by reformulating the statements so that the questions disappear. So for example 1) above will become something like 1a):

1a) There is a place X such that Mary lives at X and John knows that Mary lives at X.

This can then be rendered as something like 1b):

1b) $\exists x\textbf{: lives(m,x)}\wedge\textbf{K(j,lives(m,x))}$

An approach of this kind is in fact the one that most of the above authors use, in one form or another; however, it seems to us quite unreasonable to claim that the questions can be eliminated so simply. To illustrate why we believe this, we focus on [Rosenschein 85]'s *situated automaton* proposal and examine the consequences of his treatment.

Rosenschein, building on work in the situation semantics school [Barwise 83], defines his knowledge operator K as follows: if X is a component in a larger system, X has an internal state S, and P is a proposition, then *X knows P* (in symbolic form K(X,P)) iff P is true in all possible worlds which are compatible with the correct operation of the system, and where the state of X is S. Rosenschein shows that this implies that K satisfies the axioms of the modal logic S5, and goes on to give examples of how his definition can be used to reason about the epistemic properties of simple robots. In these examples it is frequently necessary to say that components "know where" and "know what", and this is done in the manner described above.

Unfortunately, a close examination of Rosenschein's examples shows that the knowledge operator is actually irrelevant to the arguments, all of which can be reformulated without it! This is easy to prove once we notice that K only occurs in subformulas of the type $*X=\alpha\rightarrow K(X,\phi)$, where α is a value, ϕ a formula, and $*X$ is the state of X. But since this is necessarily equivalent with $*X=\alpha\rightarrow\phi$ (this is more or less proved in the paper itself), we can always eliminate the K's.

When we consider that Rosenschein's robots are essentially dataflow machines, this is quite comprehensible. (I am indebted to Annika Waern for this observation). In everyday life, the concept "knowledge" is powerful because it can be used to reason about the acquisition of information; the statement that John knows where Mary lives is primarily interesting because it gives us the possibility of finding out where she lives by asking him. Put in a form more suitable to a robot, we know that if we connect ourselves to John and send him a "question" message we will get an "answer message" that gives us the required information. But in a dataflow machine the linkages are fixed, and no such procedure is possible. If, on the other hand, we envisage a machine powerful enough to carry out procedures of this kind, then we are back where we started; we need to be able to represent questions to send the question message. In other words, we want a formalism that will let us express the fact that a component that "knows what P is" will "tell us what P is" if we "ask it what P is". If we replace "what P is" by "Q", we can describe even more compactly what it is we want to be able to say: that if "X knows Q" and we "ask Q" then "X will tell us Q". Clearly, "Q" is the denotation of a question. As far as I can make out, [Haas 86] recognizes the importance of this point, though he never actually states it as such.

If we are now ready to accept that it would be useful to give questions denotations, the next step is clearly to investigate their form. The naive point of view here is that a question is in some sense a declarative statement where a "question element" has been "abstracted out"; this suggests that some suitable lambda-abstraction of the denotation of the associated declarative sentence is what is required. This idea was proposed as long ago as [Egli 73] and has since made periodical reappearences, most recently in [Main 83]. However, since the publication of [Kartunnen 77], this point of view has been nearly completely replaced by an alternate hypothesis, namely that *the denotation of a question is the set of propositions which together make up a complete answer to it.*

Kartunnen's interpretation has since won wide acceptance in the linguistics literature among authors working in frameworks based on or derived from Montague grammar. (This includes things like GPSG [Gazdar 85] and Situation Semantics). In the next section we are going to argue that that the naive intepretation is in fact to be prefered, but first we want to say what we think is wrong with Kartunnen's idea.

Let us give an example of how Kartunnen's proposal works. If it happens to be the case that John's wife's name is "Mary", then the denotation of 5a) will be 5b), and the denotation of 6a) will be 6b). ("?" is the "question operator" - see e.g. [Engdahl 86], p. 155).

5a) What is John's wife's name?
5b) **"?" {^∃x: wife'(j,x)∧name-of'(x,"Mary")}**

6a) Bill knows what John's wife's name is.
6b) **know'(b, {^∃x: wife'(j,x)∧name-of'(x,"Mary")})**

It seems to us fair to say that intuitively speaking there is something strange about identifying a question with its answer; we now point out some concrete difficulties resulting from Kartunnen's analysis.

Kartunnen's approach works best with question-embedding verbs like "know" and "tell", where the identification of question and answer is natural. However, there are many other verbs where this identication appears to lead to difficulties: the worst are perhaps "ask" and "find out". Suppose that John's phone number is 123-4567, and that he is the only person who has this number. Then Kartunnen's analysis will make 7a) equivalent with 7b), and 8a) equivalent with 8b), which is clearly incorrect.

7a) Mary asked Bill what John's phone number was.
7b) Mary asked Bill whose phone number was 123-4567.

8a) Mary easily found out what John's phone number was.
8b) Mary easily found out whose phone number was 123-4567.

A second objection concerns question-answer pairs. It is reasonable to demand that a semantic analysis of a question should be such that it is possible to determine from it whether or not a response is a correct answer. According to Kartunnen's proposal, this will work if the answer is in the form of a complete sentence, but *not* if it is a "short" answer. So for example 9a) can be linked to 9b) but not 9c).

9a) Q: What is John's phone number?
9b) A: John's phone number is 123-4567.
9c) A: 123-4567

The denotation of 9a) can be linked to that of 9b), since 9b) is a member of the set of answers; but there is no good way to relate it to the denotation of 9c). It is possible to claim in defence of Kartunnen's proposal that short answers are essentially elliptical, but I for one find this unconvincing, though it is of course a viewpoint that cannot be refuted as such.

A third objection to Kartunnen is provided by certain multiple embeddings of questions. Thus for example 10a) and 10b) reasonably entail 10c); but of course it is quite possible that 10d) also holds, which by Kartunnen's proposal will invalidate the argument.

10a) John believes that Mary is married to Paul.
10b) John hears Bill tell Sue that Mary is married to Paul.
10c) John believes that Bill has told Sue who Mary is married to.
10d) Mary is not married to Paul.

Our conclusion with respect to Kartunnen's proposal is thus that it fails to give a satisfactory explanation of the facts. In the next section we reexamine the naive approach, and show that it appears to overcome the above-mentioned problems.

3. A defence of the naive approach to questions

We present the "naive" approach in the following slightly unusual form: a question is a function which takes an individual **x** and a possible world **w**, and returns **true** if and only if **x** is a correct "short" answer to the question in **w**; otherwise it returns **false**. So if we let **I** be the set of individuals, **T** the set of truth-values, and **W** the set of possible worlds, we have that a question is a function $\mathbf{q}: \mathbf{I}\times\mathbf{W} \to \mathbf{T}$.

To explain the reasoning behind this choice, we first recall the standard categorial identity

11) $\mathrm{Hom}(A\times B,C) \cong \mathrm{Hom}(A,\mathrm{Hom}(B,C)) \cong \mathrm{Hom}(B,\mathrm{Hom}(A,C))$

(See e.g. [Barr 85] p. 51). Applying this, we have natural transformations between the function **q** and two other functions $\mathbf{i(q)}\in \mathrm{Hom}(\mathbf{I},\mathrm{Hom}(\mathbf{W},\mathbf{T}))$ and $\mathbf{j(q)}\in \mathrm{Hom}(\mathbf{W},\mathrm{Hom}(\mathbf{I},\mathbf{T}))$. A little thought should be enough to convince oneself that **i(q)** is the function that takes an individual into the proposition that that individual satisfies the question; while **j(q)** is the function taking a possible world into the set of individuals that are answers to the question in that world, and is thus in a natural sense *the intention of the answer*. We give a couple of examples to make this clearer: in 12) and 13), a) is the question, b) is **i(q)** and c) is **j(q)**.

12a) Where does Mary live?
12b) $\lambda x{:}\ {}^\wedge lives(m,x)$
12c) ${}^\wedge\ \lambda x{:}\ lives(m,x)$

13a) Who has a car?
13b) $\lambda x{:}\ {}^\wedge\exists y\ car(y)\wedge has(x,y)$
13c) ${}^\wedge\ \lambda x{:}\ \exists y\ car(y)\wedge has(x,y)$

We now show that this definition allows us to solve the problems described in the last section. Firstly, we define when a short response is a correct answer to a question. If we let **Q** be the denotation of the question, and **x** the denotation of the response, we can formulate as follows. (We assume that **x** ranges over individuals).

14) $\forall \mathbf{Q,x}$: **answer(Q,x)**$\leftrightarrow {}^{\vee}\mathbf{i(Q)(x)}$

14) defines the simplest kind of short answer; the answer is just the name of an individual. Actually, short answers are in practice often noun-phrases, as for example in 15) and 16).

15) Q: Who was late?
A: Three of the boys.

16) Q: Who owns a silver Mercedes-Benz?
A: Nobody I know.

It is not hard to generalize the predicate **answer** to cover answers of this type. We assume (in line with generalized quantifier theory) that an NP denotation is a set of sets (See e.g. [Barwise 81], [Cooper 83]). If **N** ranges over NP denotations, we have:

17) $\forall \mathbf{Q,N}$: **answer(Q,N)**$\leftrightarrow {}^{\vee}\mathbf{j(Q)} \in \mathbf{N}$

Next, we proceed to an important point: the connection between direct and indirect questions. For example, we want 18a) to imply 18b).

18a) Pierre asked Mary: *"Quelle heur est-il?"*
18b) Pierre asked Mary in French what the time was.

To describe this connection, we use an axiom of the following form. (**asks-directly(x,y,s)** is to be read as "**x** asks **y** the sentence **s**"; **denotes(s,Q,L)** is "the sentence **s** denotes the question **Q** in the language **L**". **asks-indirectly(x,y,L,Q)** is "**x** asks **y** a question in **L** that means **Q**").

19) $\forall$**x,y,s,Q,L: asks-directly(x,y,s)**$\wedge$**denotes(s,Q,L)**$\rightarrow$**asks-indirectly(x,y,L,Q)**

The key point is that **denotes(s,Q,L)** can be defined without reference to what the *answer* to **Q** might be; since **denotes** actually *defines* the language **L**, this should be independant of which facts happen to obtain. In Kartunnen's framework this will not be true.

Having got through these preliminaries, we present our major example. We want to define "knows what", and explain its connection with "knows"; we give a model-theoretic definition, extending the Kripke semantics for ordinary "knows". Suppose as above that **W** is a set of possible worlds, and $\mathbf{A_x}$ is the accessibility relation for the agent **x**. Then if we call the "knows" operator **K**, the normal truth conditions will be described by 20): (Φ a proposition, **w** a possible world)

20) $[\![\mathbf{K(x,\Phi)}]\!]_{\mathbf{w}}$ iff $\forall \mathbf{w'} \in \mathbf{W}: \mathbf{A_x(w,w')} \rightarrow \mathbf{\Phi(w')=t}$

It is now possible to give a similar definition for "knows what", which we symbolize **KW**:

21) $[\![\mathbf{KW(x,Q)}]\!]_{\mathbf{w}}$ iff $\forall \mathbf{w'} \in \mathbf{W}: \mathbf{A_x(w,w')} \rightarrow \mathbf{j(Q)(w')=j(Q)(w)}$

That is, **x** "knows what" **Q** iff **Q** has the same answers in each world which **x** regards as possible - a very natural definition. This of course assumes that it is possible to compare objects in different possible worlds. The acceptability of such a step has been the object of much discussion in philosophical circles, and this is a debate that we would rather not become involved in; in the present paper, we will simply assume that "cross-world identification" is defined as part of the definition of the frame, in the same way as the accessibility relation.

From 21) we can prove 22), an equivalence which defines **KW** in terms of **K**:

22) $\Box\forall \mathbf{x,Q}: \mathbf{KW(x,Q)} \leftrightarrow [\forall \mathbf{y}: {}^{\vee}\mathbf{i(Q)(y)} \rightarrow \mathbf{K(x,i(Q)(y))}] \wedge [\forall \mathbf{y}: \neg {}^{\vee}\mathbf{i(Q)(y)} \rightarrow \mathbf{K(x,\neg i(Q)(y))}]$

That is, **x** "knows what" **Q** iff he knows that each element in the answer-set is an answer, and that each element *not* in the answer set *isn't* an answer.

It is interesting to compare 22) with the postulate used by Kartunnen (footnote 11). Kartunnen relates **KW** and **K** (which he calls know'$_{IV/Q}$ and know'$_{t}$) with the formula 23) (I have taken the liberty of altering Kartunnen's notation to bring it into line with that used in the present paper).

23) $\Box\forall Q,x{:}KW(x,Q)\leftrightarrow[\forall p{:}\ p\in Q\rightarrow K(x,p)]\wedge[\neg\exists p{:}\ p\in Q\rightarrow K(x,{}^\wedge\neg\exists p{:}\ p\in Q)]$

It seems to me that 23) contains an essential flaw. It is by no means the case that "knowing all the answers" implies "knowing what"; for example, if Mary happens to be the only person who passed the exam, then even if John knows that Mary passed he will not necessarily know who passed, merely that Mary is one of them. 22) on the other hand captures the intuition that John must also know that no one else passed, something that is not expressible in Kartunnen's framework.

Another point worth noting is the way in which our proposal deals with the "Paradox of the Hooded Man". This is discussed in some detail in [Landman 86], who presents it in the following form:

We are at a party, and I say to you, pointing to a woman on the other side of the room: "Do you know who that is?" And you answer me: "No, I haven't the faintest idea." The problem now is that, given the justified faith you have in your perceptual abilities, in all your epistemic alternatives that woman is standing on the other side of the room, and so by definition you know who she is. But yet, you don't. Now I point to a man, next to her, and you say: "I know him alright, he's my brother." For the possible world analysis the situation is the same, but this time I have no problem in accepting that you indeed know who he is. To pinpoint with somewhat more precision what the problem is, let us make the following assumption: at the party there is also a man whom we both cannot tell apart from his twin brother, although we know them both. Again I ask you who that man is, and you say: "I don't know, I can't tell them apart." Again, I accept what you say is true: I, as well, cannot tell them apart.

Let us suppose that Landman points at the woman at time t_1, at my brother at time t_2, and at the twin at time t_3. Then as long as we assume an interpretation of the deictic pronoun "that" that makes it roughly equivalent with "the thing I am now pointing at" (this is an idea from Situation Semantics which can be unproblematically imported into our theory), we can describe his knowledge as

24a) $\neg KW(me,{}^\wedge\lambda x{:}(pointing\text{-}at(you,x,t_1)))$
24b) $KW(me,{}^\wedge\lambda x{:}(pointing\text{-}at(you,x,t_2)))$
24c) $\neg KW(me,{}^\wedge\lambda x{:}(pointing\text{-}at(you,x,t_3)))$

We can justify this using the equivalence 22): first we look at the case of his brother. We have that 24a) is true iff 25) holds:

25) $\forall x{:}\ pointing\text{-}at(you,x,t_2)\rightarrow K(me,{}^\wedge\ pointing\text{-}at(you,x,t_2))\wedge$
$\forall x{:}\ \neg pointing\text{-}at(you,x,t_2)\rightarrow K(me,{}^\wedge\ \neg pointing\text{-}at(you,x,t_2))$

We will need the following assumptions. Firstly, there exists some property **P** *which I recognize my brother by.* I know who has this property, and I know that there is only one person who has it. Moreover, I know that there exists a person whom you are pointing at, and who has this property; and I know that you are pointing at exactly one person. In symbolic form:

26) $KW(me,{}^\wedge\lambda x{:}P(x))$
27) $K(me,{}^\wedge\forall x,y{:}\ P(x)\wedge P(y)\rightarrow x=y)$
28) $K(me,{}^\wedge\exists x{:}\ pointing\text{-}at(you,x,t_2)\wedge P(x))$
29) $K(me,{}^\wedge\forall x,y{:}\ pointing\text{-}at(you,x,t_2)\wedge pointing\text{-}at(you,y,t_2)\rightarrow x=y)$

We now prove the first half of 25): the proof of the second half is very similar, and will be left as an exercise to the reader. Assume that for some arbitrary **x*** it is the case that 30) holds; we must prove that in this case 31) also will.

30) $pointing\text{-}at(you,x^*,t_2)$
31) $K(me,{}^\wedge\ pointing\text{-}at(you,x^*,t_2))$

We first use the fact that everything we know is true to deduce from 28) that there exists a **y*** such that 32) holds.

32) **pointing-at(you,y*,t_2)$\wedge$P(y*)**

From 30), 32) and 29) we have that **x*** is equal to **y***; so we have 33)

33) **P(x*)**

But from 26) and 22) we can deduce that I *know* this, i.e.

34) **K(me,^ P(x*))**

and using 28) again, together with 34) and 27), we have proved 31), which was the desired result. Note that the last step of the proof depends on our knowing that the person pointed at possesses the identifying property **P**; if he had a hood on, we might no longer be aware of this, and the argument would break down. This is just what happens with the twin; if we know that the person pointed at at time $\mathbf{t_3}$ has a property **P** then this property can't be one that we know identifies one of the twins uniquely. So we don't know who's being pointed at. Actually being able to *prove* that we didn't know who he was would of course require a somewhat different formulation, since we would have to quantify over properties to say that we didn't know any properties of the person pointed at outside of a certain set, those in the set being insufficient.

Without giving further examples, we hope that we have convinced the reader of our main claim: the new interpretation of questions makes it simple to express the facts we wish to state, and avoids the problems associated with Kartunnen's approach.

Before moving on to the next section, where we discuss the application of these ideas to concrete problems, we take a quick look at some linguistic issues associated with our suggestion. We divide these up under three headings: the relation between relative clauses and embedded questions, alternate questions, and relational answers.

Relative clauses and embedded questions. An interesting consequence of our treatment is that it gives embedded questions a very close relation to relative clauses. As the reader has perhaps already noticed, **i(Q)** is of a semantic type that can be associated with a relative clause, and thus it is quite reasonable to identify these two constructs semantically. It is of course well-known that there are many syntactic similarities between the two categories; this is investigated in detail in [Hirschbüler 78], who actually takes it as his starting point when attempting to give a semantics to WH-questions. In English they are particularly close, as witness the following pairs:

35a) John knows *who stole the jewels.*
35b) The girl *who stole the jewels* is still at large.

36a) John told me *which one they wanted.*
36b) The one *which they wanted* was not the best.

37a) I know *when they turned up.*
37a) The day *when they turned up* was pretty hectic.

38a) It says *what they bought.*
38b) The things *that they bought* were expensive.

39a) I saw *where he parked the car.*
39b) The street *where he parked the car* is round the corner.

The main objection to the identification of relative clauses and questions seems to be the claim that questions may contain several interrogative noun-phrases, but relative clauses only one. However, it is not obvious that this point is in itself of any great significance

Alternate questions. The only questions treated so far have been WH- (or *constituent*) questions. Kartunnen and other authors also deal with *alternate* questions, like those in examples 40) to 42).

40) Nobody knows *whether he got an A or a B on the exam.*
41) Can you find out *whether or not he wants one*?
42) Did he tell you *whether he's coming*?

It is possible to present a treatment of alternate questions along very similar lines, for example treating 40) as 40a)

40a) Nobody knows which of the following set of statements is true: {"He got an A in the exam" "He got a B in the exam"}

Relational answers. These are treated in great detail in [Engdahl 86]. Briefly, Engdahl is concerned about questions where the answer is in some way dependant on a variable bound in the question itself, as for example in 43).

43) Q: Who does every Englishman admire most?
A: His mother.

Engdahl's solution is based on Kartunnen's but involves quantifying over functions rather than individuals. As far as we can see, a similar extension is possible here. However, we are slightly dubious about the validity of questions like those Engdahl uses; my personal feeling on reading them is that most have a "rhetorical" ring, and are not questions in the normal sense of the word. In cases where the question is genuine, as for example in mathematical problems, the normal procedure is to phrase the question in terms of "arbitrary objects" [Fine 84]. 44) and 45) illustrate this.

44) Q: What is the area of a/*every circle?
A: Pi times the square of its radius.

45) Q: What is the gravitational force between two/*every two objects?
A: The product of *G* and their masses, divided by the square of the distance between them.

4. An application of the proposal to logic programming

In this section we are going to present a concrete application of the proposal made above. The situation we will consider is as follows: we have a Prolog database, which contains certain facts about the world; we will assume that all the facts in the database are true, but that the database's knowledge is not necessarily complete. The database is asked a question, and requested to list all the answers it can find. The reply is a set of individuals; under what circumstances are we justified in claiming that these individuals really are the only ones which answer the question?

Obviously, there can be no answer if we are not prepared to supply some extra information. We imagine this information as being of the form "if certain conditions obtain in the world, then the database will know what/whether certain facts are". For instance, if a robot is looking at a table from a suitable distance in an unobstructed well-lighted room, then it will know what is on it; if I have read the morning paper, then I will know whether or not the President of the United States has been assassinated the day before.

We will treat the database as an agent, which we call **db**. If a fact Φ is in the database, we write this as **K(db,Φ)**. Using our new framework, we can rephrase the problem in these terms: Given a question **Q**, we are interested in knowing the circumstances under which it is possible to prove that **KW(db,Q)**.

The first step is to use 22) to convert this into a form which contains only K's. We have

46) $\forall \mathbf{Q}: \mathbf{KW(db,Q)} \leftrightarrow [\forall \mathbf{y}: {}^{\vee}\mathbf{i(Q)(y)} \rightarrow \mathbf{K(db,i(Q)(y))}] \wedge [\forall \mathbf{y}: \neg {}^{\vee}\mathbf{i(Q)(y)} \rightarrow \mathbf{K(db,\neg i(Q)(y))}]$

We are assuming that **db** is a Prolog database, which is capable of producing a set of all answers to **Q**. So we know, given any **y**, that the inference process terminates either in success or finite failure for the proof of **i(Q)(y)**. We are also assuming that **db** contains no false information; so if **y** is not an answer to **Q**, i.e. $\neg {}^{\vee}$**i(Q)(y)**, then the proof of **i(Q)(y)** will end in finite failure. So the second half of the conjunction on the right-hand side of 46) will always be true, and given the assumptions we have

47) $\forall \mathbf{Q}: \mathbf{KW(db,Q)} \leftrightarrow [\forall \mathbf{y}: {}^{\vee}\mathbf{i(Q)(y)} \rightarrow \mathbf{K(db,i(Q)(y))}]$

That is: if **db** can infer that each answer to **Q** *is* an answer to **Q**, then it knows what the answers to **Q** are. We now give a detailed example to show how such proofs can be carried out in practice.

Our first example is taken from SNACK-85 [Rayner 86], a natural-language system which answers questions on Swedish history. The database contains among other things facts about births, deaths, and family relationships for Swedish kings and queens; of necessity, a database of this kind must be incomplete. (It would for instance be impossible to give the father of each person). However, it is possible to enter all facts which are deemed to be sufficiently important. Although the relation **son(x,y)** (x is y's son) is incomplete, we can decide that the sons of a king will be entered. In symbolic form:

48) $\forall y\text{: king}(y)\rightarrow KW(db,{}^\wedge\lambda x\text{:son}(x,y))$

Using 22), we derive from this the more useful consequence

48a) $\forall x,y\text{: son}(x,y)\wedge \text{king}(y)\rightarrow K(db,{}^\wedge\text{son}(x,y))$

Now suppose that K_1 is a king, and that he has only two sons, K_2 and K_3, both of whom became kings. We wish to determine whether **db** knows who K_1's grandsons are. (To simplify the problem, we neglect daughters). Our formalization is as follows:

49) $K(db,{}^\wedge\text{king}(K_1))$
50) $K(db,{}^\wedge\text{king}(K_2))$
51) $K(db,{}^\wedge\text{king}(K_3))$
52) $K(db,{}^\wedge\text{son}(K_2,K_1))$
53) $K(db,{}^\wedge\text{son}(K_3,K_1))$
54) $\forall x,y,z\text{: } K({}^\wedge\text{son}(x,y))\wedge K({}^\wedge\text{son}(y,z))\rightarrow K({}^\wedge\text{grandson}(x,z))$
55) $\forall x,z\text{: grandson}(x,z)\leftrightarrow\exists y\text{: son}(x,y)\wedge\text{son}(y,z)$

We wish to prove

56) $KW(db,\lambda x\text{:}{}^\wedge\text{grandson}(x, K_1))$

which by 47) is equivalent with

57) $\forall x\text{: grandson}(x,K_1)\rightarrow K(db,{}^\wedge\text{ grandson}(x,K_1))$

We assume that **x*** is such that **grandson(x*, K**$_1$**)** and attempt to prove **K(db,^ grandson(x*,K**$_1$**))**. From 55) we have that there exists a **y*** such that

58) $\text{son}(y^*,K_1)$
59) $\text{son}(x^*,y^*)$

So using 48a), 49) and 58) we have

60) $K(db,{}^\wedge\text{ son}(y^*,K_1))$

We now use "semantic attachment" [Weyhrauch 80]; we can run the query $\lambda y\text{: }{}^\wedge\text{ son}(y,K_1)$ against **db** and find all the values of **y** that satisfy it, to conclude that **y*** must be one of these. Thus we have

61) $y^*=K_2\vee y^*=K_3$

From this together with 50) and 51) we have

62) $K(db,{}^\wedge\text{king}(y^*))$

and since the database only knows true facts, we also have

63) $\text{king}(y^*)$

Using 48a), 59) and 63) we have

64) $K(db,{}^\wedge\text{ son}(x^*,y^*))$

and from this, 54) and 60) we finally establish

65) $\mathbf{K(db,{}^\wedge grandson(x^*,K_1))}$

which is the desired result.

Descending now to still more practical matters, our next question is: to what extent is it possible to perform such proofs automatically? The most obvious method is to use a mechanical theorem-prover for quantified modal logic, as described for example in [Geissler 86]. This is a line of inquiry which we certainly do not discount, and which we are also conducting experiments with; however, our feeling at the moment is that systems of this kind are not yet well enough understood to be practically usable. As usual, the price one pays for working in an extremely general framework is a corresponding loss of efficiency, and in such cases it is often necessary to seek a compromise; a limiting of expressive power in exchange for a gain in tractability. We now present some ideas in this direction.

To illustrate the basic intuition behind our attempt, we return to the previous proof. Recall that our initial goal is to prove 56), which we reproduce here:

56) $\mathbf{KW(db,\lambda x{:}{}^\wedge grandson(x, K_1))}$

Thinking now in terms of logic programming, it is natural to associate the question $\mathbf{\lambda x{:}{}^\wedge grandson(x, K_1)}$ with the Prolog query $\mathbf{grandson(?x, K_1)}$ (we write Prolog variables with an initial question-mark to distinguish them from constants). If we examine the Prolog proof procedure used in answering this question, an interesting parallel emerges. We start by using 55) to expand our initial goal, to get 66):

66) $\mathbf{son(?y,K_1), son(?x,?y)}$

We can use 48) to conclude that **db** knows all the answers to the first conjunct, $\mathbf{son(?y,K_1)}$, if we have that $\mathbf{king(K_1)}$; and we know this already, 49). We cannot do the same on the second half; however, we can expand the first conjunct in every possible way, and try a case analysis. There are two cases, both essentially the same. If we expand using 52), our goal list reduces to the single literal $\mathbf{son(?x,K_2)}$, and we can once again apply 48); if we use 53), the literal is $\mathbf{son(?x,K_3)}$, which for the same reason we also know the answers to. Hence we can conclude that 56) is true. The interesting thing is that we have really performed the same reasoning steps as in the first proof, but this time everything has been done with the normal Prolog backwards-chaining search strategy.

Let us look at another example. I would like to justify the claim that I know which Swedes have won the Nobel prize: I will assume that all Nobel prize-winners are world-famous, that if someone is world-famous I will know if they are Swedish, and that if someone is a Swede I will know whether he has won the Nobel prize. In symbolic form,we assume

67) $\mathbf{\forall x{:}\ nobel\text{-}winner(x) \rightarrow famous(x)}$

68) $\mathbf{\forall x{:}famous(x) \rightarrow}$
$\mathbf{[swedish(x) \rightarrow K(me,{}^\wedge swedish(x))] \wedge}$
$\mathbf{[\neg swedish(x) \rightarrow K(me,{}^\wedge \neg swedish(x))]}$

69) $\mathbf{\forall x{:}\ swedish(x) \rightarrow}$
$\mathbf{[nobel\text{-}winner(x) \rightarrow K(me,{}^\wedge nobel\text{-}winner(x))] \wedge}$
$\mathbf{[\neg nobel\text{-}winner(x) \rightarrow K(me,{}^\wedge \neg nobel\text{-}winner(x))]}$

and we want to prove

70) $\mathbf{KW(me,{}^\wedge \lambda x{:}\ swedish(x) \wedge nobel\text{-}winner(x))}$

Before proceeding to the proof, we make two points. Firstly, note that 69) is not enough on its own; it could conceivably be the case that I knew that each Swedish Nobel prize-winner had won the Nobel prize, without being aware that some of the people in question were Swedes. Given the *reasons* why the statement is true this is of course most improbable, but that is another matter. Secondly, and more importantly, the example suggests a notational improvement which generalizes the meaning of the KW operator; instead of 68) and 69) we would prefer to be able to write 68a) and 69a), thus:

68a) $\mathbf{\forall x{:}\ famous(x) \rightarrow KW(me,{}^\wedge swedish(x))}$

69a) $\mathbf{\forall x{:}\ swedish(x) \rightarrow KW(me,{}^\wedge nobel\text{-}winner(x))}$

To make this work, we extend the model-theoretic definition of **KW** in a natural way to cover the case where the argument is a proposition, as follows (symbols as in 20) and 21))

71) $[\![\mathbf{KW}(x,\Phi)]\!]_W$ iff $\forall w'\in W: A_x(w,w')\rightarrow\Phi(w')=\Phi(w)$

In words: $\mathbf{KW}(x,\Phi)$ iff Φ has the same truth-value in each world that **x** considers possible. From this we can derive a formula analogous to 22):

22a) $\Box\forall x,\Phi: \mathbf{KW}(x,\Phi)\leftrightarrow[{}^{\vee}\Phi\rightarrow \mathbf{K}(x,\Phi)]\wedge[\neg{}^{\vee}\Phi\rightarrow \mathbf{K}(x,\mathbf{not}(\Phi))]$

(**not** is the operator that takes a *proposition* into its negative, as opposed to $\neg$ which operates on truth-values). In words: $\mathbf{KW}(x,\Phi)$ iff Φ true implies that **x** knows that Φ, and Φ false implies that **x** knows that Φ is false. This implies the following interesting equivalence between the two **KW**'s:

72) $\Box\forall x,Q: \mathbf{KW}(x,Q)\leftrightarrow\forall y: \mathbf{KW}(x,i(Q)(y))$

that is, that "knowing what" of a question is equivalent to "knowing what" of each individual propositional instance.

We now proceed to the proof itself, which we present first in a natural-deduction style form. Our goal is to prove 70): we assume for an arbitrary **x*** that 73) holds, and try to derive 74).

73) $\mathbf{swedish(x^*)\wedge nobel\text{-}winner(x^*)}$
74) $\mathbf{K(me,{}^\wedge swedish(x^*)\wedge nobel\text{-}winner(x^*))}$

Arguing in the same way as above, we use 73) and 69) to deduce

75) $\mathbf{K(me,{}^\wedge nobel\text{-}winner(x^*))}$

From 73) and 67) we have

76) $\mathbf{famous(x^*)}$

and from 76), 68) and 73) we get

77) $\mathbf{K(me,{}^\wedge swedish(x^*))}$

Putting 75) and 77) together, we arrive at the desired conclusion.

We now rearrange the steps to fit a backward-chaining format. Our initial "goal" is

78) $\mathbf{swedish(?x)\wedge nobel\text{-}winner(?x)}$

Since we are really trying to prove a universally quantified implication, we can locally add **swedish(?x)** and **nobel-winner(?x)** to our assumptions. Looking at the first conjunct, we use 68a) to conclude that we know all the answers to this if we have **famous(?x)**; by 67) this can be reduced to **nobel-winner(?x)**, and this we have assumed. The second conjunct is similar: we know the answers to **nobel-winner(?x)** if we have **swedish(?x)**, and this is again a local assumption.

We give a third example before making an explicit statement of the principles used. Returning to our historical database, we assume that we know the sons and daughters of kings, and the dates of death for people who were sons or daughters of kings; given that K is a king, we wish to show that we know when his children died.

Our assumptions are as follows:

79) $\mathbf{king(K)}$

80a) $\mathbf{child(x,y)\leftrightarrow son(x,y)\vee daughter(x,y)}$
80b) $\mathbf{\forall x: K(db,{}^\wedge son(x,y))\rightarrow K(db,{}^\wedge child(x,y))}$
80c) $\mathbf{\forall x: K(db,{}^\wedge daughter(x,y))\rightarrow K(db,{}^\wedge child(x,y))}$

81a) $\forall x$: **king(x)$\rightarrow$KW(db,^λy: son(x,y))**
81b) $\forall x$: **king(x)$\rightarrow$KW(db,^λy: daughter(x,y))**

82a) $\forall x,y$: **king(x)$\wedge$son(y,x)$\rightarrow$KW(db,^λt: died(y,t))**
82b) $\forall x,y$: **king(x)$\wedge$daughter(y,x)$\rightarrow$KW(db,^λt: died(y,t))**

We wish to prove

83) **KW(db,^λt: $\exists$x: child(x,K)$\wedge$died(x,t))**

(It may appear that the formulation is somewhat clumsy; the reason will soon become apparent, since it has been expressly chosen so that certain problems will emerge).

The initial "goal" is **child(?x,K)$\wedge$died(?x,?t)** and we locally assume **child(?x,K)** and **died(?x,?t)**. Looking at the first conjunct, we have no direct way to prove **KW**; so we use 80a) to do a case analysis, that is to reduce the "goal" to two "subgoals", namely **son(?x,K)$\wedge$died(?x,?t)** and **daughter(?x,K)$\wedge$died(?x,?t)**. Since the proofs of both of these are exactly the same, we only give the first one. We can make the additional local assumption **son(?x,K)**; now taking the first conjunct, by 81a) we know the answers to **son(?x,K)** if we have **king(K)**, and this is 79). By 82a), we also know the answers to the second conjunct if we have **king(K)$\wedge$son(?x,K)**. We have **king(K)** as already stated; and **son(?x,K)** is our latest local assumption, so we are done.

We now attempt to generalize the picture that we have presented in the above examples. We have a database, and a set of meta-facts; we assume that all the meta-facts are of one of the forms $\Phi \leftrightarrow \wedge\Phi_i$, $\Phi \leftrightarrow \vee\Phi_i$ or $\wedge\Phi_i \rightarrow$**KW**(Φ). where Φ, Φ_i are positive literals. (In the interests of conciseness, we omit the first argument in **K** and **KW**). If the facts are of one of the first two forms, then we also assume that the "Horn-clause half" is in the database, so for example if $\Phi \leftrightarrow \wedge\Phi_i$ is in the set then so is $K(\Phi) \leftarrow \wedge K(\Phi_i)$.
We will refer to formulas of the first type as *conjunctive meta-facts*, those of the second type as *disjunctive meta-facts*, and those of the third as *knowledge meta-facts*.

Given these assumptions, we claim that the following four operations, which we refer to as *reduction*, *expansion*, *case-analysis*, and *semantic attachment* are sound; they are the basis of the interpreter presented in the appendix.

The definitions and proofs follow. In each one, Φ, Ψ, Φ_i are single formulas, and Γ a set of formulas; we write $\Gamma\vdash\Phi$ for "Φcan be derived from Γ". Note that in the interpreter each rule is to be read "backwards", that is in the form "one way to establish the conclusion is to establish the premises". It will be helpful to think of Φ as the current goal, Ψ as the "continuation", and Γ as the set of local assumptions.

Operation 1: Reduction
If $\Gamma\cup\{\Psi\}\vdash \mathbf{KW}(\Phi)$ and $\Gamma\cup\{\Phi\}\vdash \mathbf{KW}(\Psi)$, then $\Gamma\vdash \mathbf{KW}(\Phi\wedge\Psi)$.

Proof: We have $\Gamma\cup\{\Psi\}\vdash\Phi\rightarrow K(\Phi)$ and $\Gamma\cup\{\Phi\}\vdash\Psi\rightarrow K(\Psi)$. So $\Gamma\cup\{\Phi,\Psi\}\vdash K(\Phi)\wedge K(\Psi)$ and the result follows.

Operation 2: Expansion
If $\Phi\leftrightarrow\Phi_1\wedge\Phi_2$ is a formula in Γ and $\Gamma\vdash \mathbf{KW}(\Phi_1\wedge\Phi_2\wedge\Psi)$, then $\Gamma\vdash \mathbf{KW}(\Phi\wedge\Psi)$.

Proof: We must prove $\Gamma\cup\{\Phi\wedge\Psi\}\vdash K(\Phi\wedge\Psi)$. Assume $\Gamma\cup\{\Phi\wedge\Psi\}$. Then we have Φ_1 and Φ_2 since $\Phi\rightarrow\Phi_1\wedge\Phi_2$, thus we have $\Phi_1\wedge\Phi_2\wedge\Psi$, and since $\Gamma\vdash KW(\Phi_1\wedge\Phi_2\wedge\Psi)$ we get $K(\Phi_1\wedge\Phi_2\wedge\Psi)$. Since we are assuming that the Horn-clause half of each equivalence is in the database, we have $K(\Phi_1)\wedge K(\Phi_2)\rightarrow K(\Phi)$, which gives us $K(\Phi\wedge\Psi)$ as required.

Operation 3: Case-analysis
If $\Phi\leftrightarrow\Phi_1\vee\Phi_2$ is a formula in Γ, $\Gamma\vdash \mathbf{KW}(\Phi_1\wedge\Psi)$, and $\Gamma\vdash \mathbf{KW}(\Phi_2\wedge\Psi)$, then $\Gamma\vdash \mathbf{KW}(\Phi\wedge\Psi)$.

Proof: As above, assume $\Gamma\cup\{\Phi\wedge\Psi\}$. Then either $\Gamma\cup\{\Phi_1\wedge\Psi\}$ or $\Gamma\cup\{\Phi_2\wedge\Psi\}$ holds. In the first case, from $\Gamma\vdash KW(\Phi_1\wedge\Psi)$ we have $K(\Phi_1\wedge\Psi)$, and since $K(\Phi_1)\rightarrow K(\Phi)$ is in the Horn-clause half of $\Phi\leftrightarrow\Phi_1\vee\Phi_2$ we have $K(\Phi\wedge\Psi)$. The other case is exactly the same, proving the result.

Operation 4: Semantic attachment
If $\forall x$: $\Gamma\cup\{\Psi(x)\}\vdash \mathbf{KW}(\Phi(x))$, and for every **a**, $\Gamma\vdash \mathbf{K}(\Phi(a))$ implies $\Gamma\vdash \mathbf{KW}(\Psi(a))$, then
$\forall x$: $\Gamma\vdash \mathbf{KW}(\Phi(x)\wedge\Psi(x))$.

Proof: Given any a, we have $\Gamma\cup\{\Psi(a)\}\vdash KW(\Phi(a))$. To prove that $\Gamma\cup\{\Phi(a)\}\vdash KW(\Psi(a))$, assume $\Gamma\cup\{\Phi(a),\Psi(a)\}$. Since $KW(\Phi(a)$ holds, we have $K(\Phi(a))$ and thus by assumption $KW(\Psi(a))$. Now we can use Operation 1 to get the desired result.

This is enough to prove that the interpreter is sound; we would now like to prove a partial converse, namely that for some suitable additional conditions it also was complete. The following represents a first move in this direction; it is of course possible that the conditions given here can be improved on, in the sense of being made less restrictive.

Converse

If Γ fulfils the conditions given above, and also the following:

i) The set of database rules (i.e. non-unit clause) is exactly equal to the Horn-clause halves of the conjunctive and disjunctive meta-facts.
ii) No predicate symbol occurs on the left-hand side of two different meta-facts.
iii) The disjuncts on the right-hand side of a disjunctive meta-fact are mutually exclusive.

then there is a proof from $\Gamma$of a formula of type $\forall x$: $KW(\Phi(x))$ iff the proof can be derived by use of the four operations, together with normal proof operations for non-modal formulas.

Comment: The point of the conditions is to restrict us to situations where there is a logic database composed of "database predicates" (consisting of unit clauses) and "definition predicates" , which are complete definitions in terms of mutually exclusive cases. Moreover, we are only allowed to give "know what" facts about the database predicates. It is to be noted that this still covers a lot of useful programs.

Proof: (sketch) We are asked to prove $\forall x$: $KW(\Phi(x))$, and as described above this is equivalent to proving $\forall x$: $\Phi(x)\rightarrow K(\Phi(x))$. We take Φ to be a conjunction of literals, and proceed recursively.

Let Ψ be a conjunct in Φ, say $\Phi\leftrightarrow\Psi\wedge\Psi'$. Now by hypothesis Ψ is either a "database" predicate or a "definition" predicate; if the latter is true, then the definition is either a conjunctive meta-fact or a disjunctive meta-fact.

If it is a conjunctive meta-fact, say $\Psi\leftrightarrow\wedge\Psi_i$, then we have $\Phi\leftrightarrow\wedge\Psi_i\wedge\Psi'$ and $K(\Phi)\leftrightarrow K(\wedge\Psi_i\wedge\Psi')$ since by hypothesis ii) above we can only derive $K(\Psi)$ by using Ψ's definition. Thus the result is equivalent with $\wedge\Psi_i\wedge\Psi'\rightarrow K(\wedge\Psi_i\wedge\Psi')$, which is equivalent with $KW(\wedge\Psi_i\wedge\Psi')$, or in general equivalent with the result of performing an expansion.

If on the other hand the definition of Ψ is a disjunctive meta-fact, say $\Psi\leftrightarrow\vee\Psi_i$, then we have as above that $\Phi\leftrightarrow\vee(\Psi_i\wedge\Psi')$ and $K(\Phi)\leftrightarrow K((\vee\Psi_i)\wedge\Psi')$. Since Γ is a database, the only way to derive a disjunction is to derive one of the disjuncts, and thus we have $K(\Phi)\leftrightarrow\vee K(\Psi_i\wedge\Psi')$. So we have reduced the initial formula to $\vee(\Psi_i\wedge\Psi')\rightarrow\vee K(\Psi_i\wedge\Psi')$. We now use hypothesis iii) to conclude that, since Ψ_i are all mutually exclusive, this is equivalent to $\wedge KW(\Psi_i\wedge\Psi')$, or in general equivalent with the result of performing a case-analysis.

Remember that we are assuming that "bag of" the program always terminates either with sucess or finite failure. So by performing a finite number of expansion and case analysis steps, we can show that the initial formula is equivalent to $KW(\wedge\Psi_i)$, where Ψ_i are all database predicates. This is then (by reduction steps) equivalent with $\wedge(\Psi_i'\rightarrow KW(\Psi_i))$, where Ψ_i' is the conjunction of all the conjuncts *excluding* Ψ_i. But the only ways to prove K of a database predicate are either to use a KW clause for it (which reduces the formula to a non-modal one), or to look it up directly, if it is equivalent to a ground formula. However, the only way in which this last case can occur is if one of the assumptions implies a disjunction of identities, since database predicates do not occur on the left-hand side of definitions. This last case occurs precisely when it is possible to perform an application of the semantic attachment rule, which completes the proof.

5. Conclusions and further directions

We have argued that work on the semantics of questions is relevant to research in AI and logic programming, and compared what we view as the two main approaches. Our conclusion is that the earlier, "naive", approach is superior to Kartunnen's 1977 suggestion, and we have supported this claim by giving examples which are handled correctly by the naive method, and incorrectly by Kartunnen's approach. We have then considered in detail the special case of "know" treated as a question-embedding verb, which we have referred to as "know what". We have used our question semantics to give "know what" a model-theoretic definition, and from this we have derived a relation between "know what" and "know". Finally, we have demonstrated that the "know what" concept can be used practically in logic programming, to reason formally about the completeness of database programs. We have first given examples of natural-deduction style proofs in a general context, and then shown that it is possible to carry out proofs of this kind in an efficient goal-directed manner if further assumptions are made about the nature of the database. Given these assumptions, we have shown that our proof procedure is both sound and complete, and we have implemented it in the form of a small Prolog meta-interpreter.

Finally, we sketch very briefly some developments which we are currently investigating, aimed towards a generalization from "know what" to "can find out what". We assume that it is possible to perform "epistemic acts" which extend the database; these could for example be perceptual acts, looking up information from a reference source, or accessing another database. Given a query, we are now interested in determining whether there exists a sequence of such acts (or more generally a plan using such acts), the execution of which would result in the database "knowing what" the answers are.

It seems feasible to extend a goal-directed proof procedure like that described in the previous section to cover reasoning of this kind. We have knowledge meta-facts which say that the database knows the answers to questions, if certain conditions obtain; in addition, we give "action" meta-facts which say that conditions *would* obtain if appropriate pre-conditions are fulfilled, and a suitable action is carried out. For example, a meta-fact might be that the agent knows what an individual's telephone number is if he has looked it up; and as a related action meta-fact, we can say that if the agent is holding the telephone directory, and knows X's name, then he can perform the action "look up X's number". In general, it does not seem to us that the problems that arise here are harder than those involved in other planning problems; in fact, there are some positive aspects resulting from the fact that actions cannot *destroy* knowledge that already has been acquired. This constrasts favorably with many cases reported in the planning literature, where the necessity to "protect" conditions during the execution of an operation is a major problem. We hope that a more detailed exposition of these ideas will form the subject of a future paper.

6. Acknowledgements

We would like to thank Per Kreuger, Annika Waern, and Professor Dag Prawitz for many helpful comments on previous drafts of this paper.

Appendix: A "Knows-What" Interpreter in Prolog

The interpreter is (conceptually) divided into two parts. Firstly, there is the KW-prover (provekw), that performs the operations *reduction* (clause 2 and 3), *expansion* (clause 6), *case-analysis* (clause 4), and *semantic attachment* (clause 5). Secondly, there is the collector of bindings for the semantic attachment (cases, and known_if). Knowledge- rules are expressed with the K-relation, conjunctive and disjunctive meta-facts are expressed with the <->-relation, and knowledge meta-facts are expressed with the KW-relation.

The KW-prover uses a special representation for the KW-goal being proved. It is divided into two parts—the current goal, and the continuation. The idea is that the continuation always shall be proved by a case analysis, or a semantic attachment, of the current goal. Clauses 2 and 3 make it possible to interchange the roles of current goal and continuation, and are thus symmetric variants of the reduction rule. The expansion rule is trivial. The conjunctive and disjunctive meta-facts are not separated at this stage, but later in the proving of a disjunctive KW-goal. It should be noted that <->-rules are specially prepared to work in the ->-direction, as this is the form in which they are used (the <- -direction is only a necessary prerequisite for the rule to be applicable). In other words, the skolem-functions that appear are conceived as coming from quantifiers in the ->-direction.

The semantic attachment (know_if) works as an ordinary K-prover, but the unification is allowed to 'bind' the skolem-functions by entering an equivalence on the list of hypothesises (assuming it). These bindings are kept consistent by 'dereferencing'. All possible bindings that makes the goal provable are gathered by the cases-prover (cases). It then proves the continuation for each of these hypothetical binding-environments. In other words, the semantic attachment operation is reduced to the case-analysis one, by using the fact that a predication is under these circumstances equivalent to a disjunction of identities.

```
:-  op(1200,xfx,'<->').

provekw(true,C,_) :-!, (C=true; C=cont(P,C1,H), provekw(P,C1,H)).
provekw((P,Q),C,H) :- assume(Q,H,H1), provekw(P,cont(Q,C,H),H1).
provekw((P,Q),C,H) :-!, assume(P,H,H1), provekw(Q,cont(P,C,H),H1).
provekw((P;Q),C,H) :-!, provekw(P,C,H), provekw(Q,C,H).
provekw(G,C,H) :- kw_clause(G,B,H), prove(B,H), cases(G,C).
provekw(G,C,H) :- eq_clause(G,B,H), provekw(B,C,H).
```

```
cases(_,true).
cases(G,cont(P,C,H)) :-
    (bagof(H1,known_if(G,H,H1),L) ; L=[]),
    \+ (member(H1,L), \+ provekw(P,C,H1)).

prove(true,_)  :-!.
prove((P,Q),H)  :-!,  prove(P,H),  prove(Q,H).
prove((P;Q),H)  :-!,  prove(P,H);  prove(Q,H).
prove(kw(P),H)  :-!,  provekw(P,true,H).
prove((X=Y),H)  :-  unify(X,Y,H).
prove(G,H)  :-  mem(G,H,H).
prove(G,H)  :-  k_clause(G,B,H),  prove(B,H).

known_if(true,H,H)  :-!.
known_if((P,Q),H0,H)  :-!,  known_if(P,H0,H1),  known_if(Q,H1,H).
known_if(G,H0,H)  :-  k_clause_if(G,B,H0,H1),  known_if(B,H1,H).

k_clause_if(G,B,H0,H)  :-  k(G1,B),  unify_if(G,G1,H0,H).
k_clause(G,B,H)  :-  k(G1,B),  unify(G,G1,H).
kw_clause(G,B,H)  :-  kw(G1,B),  unify(G,G1,H).
eq_clause(G,B,H)  :-  (G1  <->  B),  unify(G,G1,H).

unify_if(X,Y,H,H1)  :-  deref(X,X1,H),  deref(Y,Y1,H),  !,  unify_if_1(X1,Y1,H,H1).

unify_if_1(X,X,H,H)  :-!.
unify_if_1(sk(I,D),C,H,[(sk(I,D)=C)|H])  :-!.
unify_if_1(C,sk(I,D),H,[(sk(I,D)=C)|H])  :-!.
unify_if_1([X|R],[Y|S],H0,H)  :-!,  unify_if(X,Y,H0,H1),  unify_if(R,S,H1,H).
unify_if_1(X,Y,H0,H)  :-  X=..[F|A1],  Y=..[F|A2],  unify_if(A1,A2,H0,H).

unify(X,Y,H)  :-  deref(X,X1,H),  deref(Y,Y1,H),  !,  unify_1(X1,Y1,H).

unify_1(X,X,_)  :-!.
unify_1([X|R],[Y|S],H)  :-!,  unify(X,Y,H),  unify(R,S,H).
unify_1(X,Y,H)  :-  X=..[F|A1],  Y=..[F|A2],  unify(A1,A2,H).

deref(X,V,H)  :-  nonvar(X),  X=sk(_,_),  member((X=Y),H),  !,  deref(Y,V,H).
deref(X,X,_).

assume((P,Q),L0,L)  :-!,  assume(P,L0,L1),  assume(Q,L1,L).
assume(P,L,[P|L]).

mem(E,[E1|_],H)  :-  unify(E,E1,H).
mem(E,[_|R],H)  :-  mem(E,R,H).

%  the examples

k(king(k1),true).
k(king(k2),true).
k(king(k3),true).
k(son(k1,k2),true).
k(son(k1,k3),true).

kw(nobelp(X),swede(X)).
kw(swede(X),famous(X)).
kw(son(X,Y),king(X)).
kw(daughter(X,Y),king(X)).
kw(died(Y,T),(king(X),son(X,Y))).
kw(died(Y,T),(king(X),daughter(X,Y))).
```

```
grandson(X,Z)  <->  son(X,sk(2,[X,Z])),son(sk(2,[X,Z]),Z).
child(X,Y)  <->  daughter(X,Y);son(X,Y).

famous(X)  :-  nobelp(X).

%   prove(kw(grandson(k1,sk(1,[]))),[]).
%   prove(kw((nobelp(sk(3,[])),swede(sk(3,[])))),[]).
%   prove(kw((child(k1,sk(4,[])),died(sk(4,[]),sk(5,[])))),[]).
```

References

[Barr 85]	Barr & Wells *Toposes, Triples and Theories* Springer, 1985
[Barwise 83]	Barwise, J. & Perry, J. *Situations and Attitudes* MIT Press, 1983
[Bennett 79]	Bennett, M. *Questions in Montague Grammar.* University of Indiana Linguistics Club, 1979
[Brachman 85]	R. Brachman & H. Levesque (eds.) *Readings in Knowledge Representation* , Morgan Kaufmann 1985
[Cooper 83]	Cooper, R. *Quantification and Syntactic Theory,* D. Reidel 1983
[Egli 73]	Egli, U. *Semantische Repraesentation der Frage,* Dialectica **27**, p. 363-370.
[Engdahl 86]	Engdahl, E. *Constituent Questions,* D. Reidel, 1986
[Fine 84]	Fine, K. *A Defence of Arbitrary Objects* in GRASS 3, Foris, Dordrecht, 1984
[Gazdar 85]	Gazdar, G., Klein, E., Pullum, G. & Sag, I. *Generalized Phrase Structure Grammar,* Basil Blackwell 1985
[Geissler 86]	Geissler, C. & Konolige, K. *A Resolution Method for Quantified Modal Logics of Knowledge and Belief,* in [Halpern 86]
[Haas 86]	Haas, R. *A Syntactic Theory of Knowledge and Belief* Artifical Intelligence, 1986
[Halpern 86]	Halpern, J. (ed.) *Theoretical Aspects of Reasoning about Knowledge* Morgan Kaufmann 1986
[Hausser 79]	Hausser, R. and Zaefferer, D. *Questions and Answers in a Context-dependant Montague Grammar* , in Guenthner, F. and Schmidt, S. (eds.) *Formal Semantics and Pragmatics for Natural Language,* D. Reidel, 1979
[Hintikka 76]	Hintikka, J. *The Semantics of Questions and the Questions of Semantics,* D. Reidel 1976
[Hirschbüler 78]	Hirschbüler, P. *The Syntax and Semantics of WH-Constructions.* PhD thesis, University of Massachusets at Amherst, 1978
[Kartunnen 77]	Kartunnen, L. *Syntax and Semantics of Questions* Linguistics and Philosophy **1**, 1977
[Konolige 82]	Konolige, K. *Knowledge and Planning in a Multi-agent System* in *Machine Intelligence* **10**, ed. D. Michie and Y. Pao, Edinburgh 1982
[Landman 86]	Landman, F. *Pegs and Alecs,* in [Halpern 86]
[Maida 83]	Maida & Shapiro, S. *Intentional Concepts in Semantic Nets,* in [Brachman 85].
[Main 83]	Main, M.G. & Benson, D.B. *Denotational Semantics for "Natural Language" Question-Answering Programs* American Journal of Computational Linguistics, Vol **9** p. 11-21
[McCarthy 75]	McCarthy, J. *Some Philosophical Problems from the Standpoint of Artifical Intelligence* in [Brachman 85].
[Moore 80]	Moore, R. *A First-order Theory of Knowledge and Action,* SRI 1980
[Rayner 86]	Rayner, M. & Banks, A. *Temporal Relations and Logic Grammars,* Proc. ECAI 1986
[Rosenschein 85]	Rosenschein, S. & Kaebling, L. *The Synthesis of Machines with Provable Epistemic Properties,* in [Halpern 86]
[Weyhrauch 80]	Weyhrauch, R. *Prolegomena to a Theory of Mechanized Formal Reasoning* Artificial Intelligence **13**, no 1-2, 1980