A Quick Overview of QUARK

Massimo Poesio,
Department of Computer Science, University of Rochester,
Rochester, NY 14627, USA
poe@cs.rochester.edu

1989

Abstract

The development of a dialogue understanding system presents fairly
interesting representational problems, most of which are beyond the ca-
pabilities of the currently available term subsumption systems (i.e., those
developed in the KRYPTON tradition.). In this paper I will discuss some
of these problems (namely, the representation of the agents' beliefs and
goals, the representation of temporal information, and of complex dis-
course entities evoked by, e.g., generalized quantifiers and plurals), I will
present the assertional language developed for an ABox called QUARK,
and explain how the system has been implemented.

1 Motivations

The developers of term subsumption languages can be divided in extensionists
- those who claim that the currently available languages have to be extended:
[13, 9] - and reductionists, those who are more interested in keeping the computa-
tional complexity low, and therefore would prefer simpler but more tractable
languages. [22, 23, 14].

I am taking an extensionist position here. I will describe work done in
support of the dialogue understanding system WISBER [11], which gives advice
on financial investments. The development of such a system presents fairly
interesting representational problems, most of which are beyond the capabilities
of the currently existing hybrids developed in the tradition of KRYPTON [4,
22, 23]. In the rest of this section I will list these problems; in the next section
I will present the assertional language which has been developed to cope with
them, called IRS; finally, I will describe the implementation of QUARK, the
ABox we have developed, which uses IRS as an interface.
Representing Mental States

The purpose of a system like WISBER is to understand the beliefs and goals of the person seeking its advice and try to satisfy them. It does this by setting to itself goals to achieve: to make a very simple example, when the user asks information about a certain term, WISBER adds a new goal to its list of tasks, namely answering that question. Both the user and WISBER may have more goals to achieve; and, in the course of the conversation, both WISBER's and the user's beliefs and goals change - for example, a certain question is answered, and doesn't have to be answered again.

For all of this to work in a principled way, it is necessary that the discourse model include a representation of the beliefs and goals of both the user and WISBER. The idea of just taking one of the existing logics of belief and goal states (e.g., [7, 10], however, is undermined by the necessity for the representation to satisfy the following desiderata:

- more than one agent are involved;
- beliefs and goals have a time of validity - that is, they don't hold forever (goals are achieved, beliefs are corrected, etc.)
- belief and goal states have to be treated as any other state, like, e.g., OWN or LIKE: by which I mean that we want to have in our ontology concepts like BELIEF and WANT, whose instances are associated with the beliefs and goals of the agents. The reasons: a tighter interconnection of the terminological and assertional knowledge bases; since the ontology is used by the parser to check selectional constraints \(^1\), sentences like

\[(1) \quad \text{I believe bonds are safer than stocks}\]

...can be processed in the same way than sentences like \(I \text{ own bonds}\); and the same kind of role analysis can be used for \(\text{believe}\) and \(\text{want}\) that is used for the other stative verbs.

On the other hand, we can't just define the concepts BELIEF and WANT, and make beliefs and states instances of these concepts. The semantics of propositional attitudes raises a lot of issues which have been extensively explored in the literature, the more relevant of which have been for us how to represent the object of these states (e.g., \(\text{bonds are safer than stocks}\) in (1)), and what kind of axiomatic treatment of propositional attitudes to use. On the practical side, we have worried about how to organize the knowledge base of QUARK so that inferences like

\[\text{BEL}(A, t, P) \vdash \text{BEL}(B, t, P)\]

\(^1\)A discussion of whether this solution is optimal in handling e.g., metaphor is a matter completely outside the scope of this paper
are not licensed, at the same time granting the accessibility of the terms from one belief space to the other (what this means will be explained in section 4).

Our solution, in a nutshell, has been that of endowing the assertional language with the two modal operators \( BEL \) and \( WANT \) (whose semantics will be defined in the usual way) and a ‘state-forming’ operator \( \alpha^* \) which takes a proposition \( \alpha \) and yields a predicate whose extension in each possible world is the unique individual associated to \( \alpha \) (similar operators have been suggested, among others, by Reichenbach, Schubert and Pelletier, and Chierchia and Turner [19, 21, 6]); and to express the relation between the concepts \( BELIEF \) and \( WANT \) and the corresponding modal operators as constraints on the model expressed via the state forming operator.

Incorporating Time into the Knowledge Base

\( BELIEF \) and \( WANT \) are not the only concepts whose extension varies with time. The chairman of bank, for example, won’t be a chairman forever; a certain state of owning existing between me and a certain group of stocks will only hold as long as I don’t sell the stocks, etc. In a highly dynamical context like financial transactions, it’s necessary to have a different semantics for concepts and roles than that provided by Schmolze and Israel [20].

Financial transactions have the additional characteristic of involving extremely complex temporal patterns, like, e.g., periodic events or complex events related in complicated ways:

... you first buy a certain amount of stocks of type A, then sell part of them after six months and get some bonds of type B and more stocks of type C

... and so forth.

Last but not least, an adequate treatment of tense and temporal adverbs is expected of any NLU system, which, again, presupposes a good representation of time.

Our treatment of time is based on Allen and Hayes’ Interval Calculus [1], which we have modified as described in [17] to handle periodic events. Our model includes therefore a temporal structure \( J \) organized by a \( meet \) relation and closed over a sequencing operation \( \triangleleft \).

Representation of Discourse Entities

Any reasonably sophisticated natural language has to handle phenomena like anaphora and to be able to understand (and use) NPs like some bonds or more stocks presented in the sentence quoted in the previous paragraph. No existing
hybrid can be used to build a discourse model of this kind (although some
proposals in this direction have been presented, e.g., in the context of SB-TWO).
We have followed in doing this the path of Link [12], which consists in using
as domain of discourse a complete boolean algebra hierarchically organized by
a part-of relation whose correspondent in the object language is the logical
predicate \( \preceq (x, y) \). This work is currently being extended and revised [18]; the
version presented here is the one implemented in QUARK.

2 The ABox Language

Syntax

The basic structure of the assertional language, called IRS, is very similar to that
of the assertional languages of the other hybrid systems: the atomic formulas
can be either one-place, concept predicates; or two-place, role predicates; or
three-place, modal predicates. A small set of logical two-place predicates is
also present, which includes equality (=) and group inclusion (\( \preceq \)). Non-atomic
formulas are built in the usual way using connectives and quantifiers. As in [2],
a quantifier is constituted by a determiner, a bound variable, and a range. The
following is the IRS formula which the parser will build as the representation of
the sentence Sue owns a saving account:

\[(\lambda x: \text{SAVINGS-ACCOUNT}(x)), (\lambda y: \text{OWN}(y)) \quad \text{HAS-OWNER}(y, \text{sue}) \land \text{HAS-POSSESSION}(y, x)\]

The basic non-logical expressions of IRS include an infinite denumerable set
of constants Con and an infinite denumerable set of variables Var; a set of
one-place predicates C (the concepts); and a set of two-place predicates R (the
roles).

The logical expressions of IRS include the logical predicates = (equality), \( \preceq \)
(group inclusion), and TT (temporal validity); the usual connectives; a set Det
of determiners (a, some, every, each, many, most, few, and the cardinals), which
denote functions from subset of the domain into subsets of the domain; the two
modal operators BEL and WANT, and a set of concept forming operators, among
which lambda abstraction \( \lambda \), the group abstraction operators \( * \) (which takes a
concept \( \chi \) and returns the concept whose denotation includes both instances of
\( \chi \) and groups whose members are instances of \( \chi \)) and \( \mathbb{G} \) (which takes a concept
\( \chi \) and returns the concept whose denotation includes only the groups whose
members are instances of \( \chi \)).

The syntax of IRS is defined as follows:

Terms

1. Every constant in Con is a term
2. Every variable in \textbf{Var} is a term
3. Nothing else is a term.

\textbf{Atomic Formulas}
1. If $\chi$ is a concept and $\alpha$ is a term, $\chi(\alpha)$ is an atomic formula.
2. If $\rho \in \mathbf{R}$ and $\alpha, \beta$ are terms, $\rho(\alpha, \beta)$ is an atomic formula.
3. If $\alpha$ and $\beta$ are terms, $\alpha = \beta$, $\alpha \preceq \beta$ and $\text{TT}(\alpha, \beta)$ are atomic formulas.
4. Nothing else is an atomic formula.

\textbf{Formulas}
1. Every atomic formula is a formula.
2. If $\alpha$ is a formula, $\neg \alpha$ is a formula
3. If $\alpha$ and $\beta$ are formulas, $\alpha \land \beta$, $\alpha \rightarrow \beta$ and $\alpha \lor \beta$ are formulas.
4. If $D \in \mathbf{Det}$, $u \in \textbf{Var}$, $\chi$ is a concept, and $\alpha$ is a formula,
\[ (D \ u \ \chi(u)) \alpha \]
is a formula.
5. If $\alpha$ is a formula, $a$ is a term, and $i$ is a term, $\text{BEL}(a, i, \alpha)$ and $\text{WNT}(a, i, \alpha)$
are formulas.
6. Nothing else is a formula.

\textbf{Concept Forming Operators}
1. If $\chi \in \mathbf{C}$ then $\chi$ is a concept.
2. If $\alpha$ is a formula and $u \in \textbf{Var}$, $\lambda u. \alpha$ is a concept.
3. If $\chi$ is a concept, $^* \chi$ and $^\circ \chi$ are concepts.
4. If $\alpha$ is a formula, $|\alpha|^*$ is a concept.
5. Nothing else is a concept.
Semantics

The semantics for the attitude reports in IRS is the classical one, based on possible worlds and accessibility relations [8]. The lattice-oriented semantics for plurals and quantifiers is an adaptation of work from Link [12]. The treatment of the \([\alpha]^*\) operator is mostly derived from [6].

A model \(M\) of IRS will be a tuple \(<E, U, I, P, S, Delta, T, \zeta, \varphi, V>\), where

- \(E = O \cup J \cup NF\) is a complete atomic Boolean algebra of entities, with join operation \(\cup_i\) and intrinsic order relation \(\leq_i\); \(O, J\) and \(NF\) are mutually disjoint. The set of atoms in \(E\) will be called \(A\). \(O\) is a non-empty set of objects which includes, among other things, a set \(I = \{0, 1\}\) of truth values. \(J\) is a set of time intervals on which is defined a meets relation \([1]\), and closed under a sequencing operator \(\varsigma\). \(NF\) is a set which contains one element for every function in \([E \to E]\) (the bijection is defined by the functions \(\gamma\) and \(\delta\), see below). The idea is that there will be an individual in \(NF\) for every formula \(\alpha\) expressible in IRS; this individual will be the extension of the predicate obtained by applying the \([\alpha]^*\) operator to \(\alpha\). The lattice structure imposed on the domain of individuals has the purpose of defining a denotation for plurals and quantified expressions.

- \(U\) is a model of \(\lambda\)-calculus \(<E, [E \to E], \gamma, \delta>\), where \([E \to E]\) is a set of functions defined over \(E\). \(\gamma : [E \to E] \to E\) and \(\delta : E \to [E \to E]\). Intuitively, \(\gamma\) associates an individual of \(E\) with every function (in particular, an individual of \(NF\) with every function from \(E\) to \(I\)), and \(\delta\) defines the inverse (partial) mapping from individuals to functions; it is required that \(\delta(\gamma(f)) = f\).

- \(I = <I, \cap_r, \sim, \cup_r, \equiv>\), where \(r = i, u, n, f\) or \(e\), is a boolean algebra of information units. \(\cap : I \times I \to I\), \(\sim : I \to I\), \(\equiv : E \times E \to I\), and \(\cup_r : [E_r \to I] \to I\). \(E_r = E, E_{nf} = NF = \{e \in E : \exists f \in [E \to E] \land \gamma(f) = e\}\). \(E_i = I\), and \(E_u = E \setminus F\). The values of the formulas of IRS will be elements of \(I\). \(\sim\) is used to interpret negation, and \(\cap\) to interpret conjunction.

- \(P = <\varphi(W \times J), Delta, \triangle, \Delta_r, \equiv, 1, 0>\) is a boolean algebra of propositions.

- \(T : I \to P\) is an isomorphism defined as follows:

  \(T(i \cap i') = T(i) \Delta T(i')\)
  \(T(\sim i) = \neg T(i)\)
  \(e \equiv e' = e = e'\)
  \(T(\cup_r f) = \Delta_r \lambda \epsilon T(f(e))\)
  \(T(\Delta(e)) = S(e)\), where \(\Delta : E \to I\) and \(S : E \to P\) is the truth extension function.
- $W$ is a set of possible worlds
- $\zeta$ is the belief accessibility relation, $\varphi$ the want accessibility relation
- $V$ is the value function.

The use of $P$ and $T$ is discussed in more detail in [6]. The denotation of the expressions of IRS with respect to model $M$, $w \in W$, $i \in I$ and value assignment $g$, indicated by $\llbracket \alpha \rrbracket_{M,w,t,i,g}$ will be recursively defined as follows:

**Basic Expressions**

1. If $\alpha$ is a non-logical constant of the language (constant, predicate or determiner) $\llbracket \alpha \rrbracket_{M,w,t,i,g} = [V(\alpha)]_{w,t}$
2. If $\alpha \in \text{Var}$, $\llbracket \alpha \rrbracket_{M,w,t,i,g} = [g(\alpha)]$

**Atomic Formulas**

1. If $\chi$ is a concept and $\alpha$ a term, $\llbracket \chi(\alpha) \rrbracket_{M,w,t,i,g} = \llbracket \chi \rrbracket_{M,w,t,i,g}(\llbracket \alpha \rrbracket_{M,w,t,i,g})$
2. If $\rho$ is a role and $\alpha$, $\beta$ are terms, $\llbracket \rho(\alpha, \beta) \rrbracket_{M,w,t,i,g} = \llbracket \rho \rrbracket_{M,w,t,i,g}(\llbracket \alpha \rrbracket_{M,w,t,i,g}, \llbracket \beta \rrbracket_{M,w,t,i,g})$
3. If $\alpha$ and $\beta$ are terms, $\llbracket \alpha = \beta \rrbracket_{M,w,t,i,g} = 1$ iff $\llbracket \alpha \rrbracket_{M,w,t,i,g} = \llbracket \beta \rrbracket_{M,w,t,i,g}$
4. If $\alpha$ and $\beta$ are terms, $\llbracket \alpha < \beta \rrbracket_{M,w,t,i,g} = 1$ iff $\llbracket \alpha \rrbracket \neq 0$ and $\llbracket \alpha \rrbracket \leq_i \llbracket \beta \rrbracket$
5. $\llbracket TT(x,i) \rrbracket_{M,w,t,i,g} = 1$ iff $x = uw[\alpha](w)$ for some $\alpha$ and $\llbracket i \rrbracket_{M,w,t,i,g} = t'$ and $\llbracket \alpha \rrbracket_{M,w,t,i,g} = 1$

**Formulas**

1. $\llbracket \neg \alpha \rrbracket_{M,w,t,i,g} = \neg \llbracket \alpha \rrbracket_{M,w,t,i,g} (\neg \llbracket \alpha \rrbracket_{M,w,t,i,g} is 1 iff \llbracket \alpha \rrbracket_{M,w,t,i,g} = 0 and 0 otherwise)$
2. $\llbracket \alpha \land \beta \rrbracket_{M,w,t,i,g} = \llbracket \alpha \rrbracket_{M,w,t,i,g} \cap \llbracket \beta \rrbracket_{M,w,t,i,g}$ (which is 1 iff both $\llbracket \alpha \rrbracket_{M,w,t,i,g}$ and $\llbracket \beta \rrbracket_{M,w,t,i,g}$ are 1, otherwise it is 0). (The definitions for the other connectives can be derived from the two just given in the usual way.)
3. $\llbracket (D u \chi(u)) \alpha \rrbracket_{M,w,t,i,g}$ is 1 iff there exists a group $x$ among those returned by applying $\llbracket D \rrbracket$ to $\llbracket \chi(u) \rrbracket$, such that $\llbracket \alpha \rrbracket_{M,w,t,i,g}^{\llbracket (D u \chi(u)) \alpha \rrbracket_{M,w,t,i,g}} = 1$ (i.e., $[\alpha]$ is true when evaluated with respect to the assignment which differs from the previous one only in the value associated to u)
4. $\llbracket BEL(i, a, i) \rrbracket_{M,w,t,i,g} = 1$ iff for every $w'$ in W such that $\zeta(w, a, i) = w'$, $\llbracket \alpha \rrbracket_{M,w',t,i,g} = 1$. Similarly, $\llbracket WNT(i, a, i) \rrbracket_{M,w,t,i,g} = 1$ iff for every $w'$ in W such that $\varphi(w, a, i) = w'$, $\llbracket \alpha \rrbracket_{M,w',t,i,g} = 1$. 

...
3 THE TBOX LANGUAGE

Concept Forming Operators

1. $\|\alpha\|_{M,w,t,G}^{\mathcal{E}}$ is the function $h$ with domain $\mathcal{E}$ such that for every object $x$ in the domain, $h(x) = \|\alpha\|_{M,w,t:G}^{\mathcal{E}}$, where $g_{x/u}$ is the assignment exactly like $g$ with the possible difference that $g_{x/u}(u)$ is the object $x$.

2. $\|\chi\|_{M,w,t,G}^{\mathcal{E}} = \{\|\chi\|_{M,w,t,G}^{\mathcal{E}}\}$ (the complete $\sqcup_i$ sub-semilattice generated by $\|\chi\|$). $\|\chi\|_{M,w,t,G}^{\mathcal{E}} = \|\chi\|_{M,w,t,G}^{\mathcal{E}} \setminus \emptyset$

3. $\|\alpha\|_{M,w,t,G}^{\mathcal{E}} = \gamma(\alpha)$

Constraints on the Model

For all concepts $\chi$ representing states and events, the normalization of the concept and the instance of the concept are the same:

$$\forall \chi, x \ (\forall x \chi(x) \supset (\text{STATE}(x) \lor \text{EVENT}(x))) \equiv x = \iota w. [\chi(x)]^* (w)$$

The relation between the denotation of the concept BELIEF and the modal operator $\text{BEL}$ is expressed by the following axiom schema:

$$\forall x, y, z, t \ (\text{BELIEF}(x) \land \text{EXPERIENCER}(x, y) \land TT(x, t) \land \text{THEME}(x, z)) \equiv \text{BEL}(y, t, \alpha) \land z = \iota w. [\alpha]^* (w) \land x = \iota w. [\text{BEL}(y, t, \alpha)]^*(w)$$

An axiom schema of the same form relates the denotation of WANT to the modal operator $\text{WNT}$.

3 The TBox language

In this paper I will only use the relevant subset of QUIRK, and modify its syntax to make it similar to that of standard references like [20]. More details on QUIRK can be found in [3].

3.1 Syntax

The set of terms of QUIRK is partitioned in two groups, concepts and roles. Let $C$ denote all of the concepts, and $R$ all of the roles. The valid concepts according to QUIRK are the following:

1. THING;

2. (CCONGENERIC $C_1 \ldots C_n$), where $C_1 \ldots C_n$ are concepts;

3. (RESTRICTION $R \ (\text{VR} \ C)$), where $R$ is a role;

4. (RESTRICTION $R \ (\text{MIN} n)$);
4 IMPLEMENTATION

5. (RESTRICTION R (MAX m));

The valid roles are the following:
1. ASPECT;
2. (RCONGENERIC R₁ ... Rₙ), where R₁ ... Rₙ are roles;
3. (DOMAIN C), where C is a concept;
4. (RANGE C), where C is a concept;

For example, the definition of OWN in QUIRK looks like the following

(DEFCONCEPT OWN
 (CCONGENERIC STATE
  (RESTRICTION HAS-OWNER (VR HUMAN BEING) (MIN 1))
  (RESTRICTION HAS-POSSESSION (MIN 1)))))

3.2 Semantics

I will list here the interpretation of the concept-forming operators; the interpretation of the role forming ones is similar. ∀ₙₑyₙ is short for ∃₁ᵣ₁ ... ∃ₙᵣₙ.

1. ∀ₓTHING(x)
2. ∀ₓ(CCONGENERIC C₁ ... Cₙ)(x) ≡ C₁(x) ... Cₙ(x)
3. ∀ₓ(RESTRICTION R(VRC))(x) ≡ ∀ᵦ'R(x, y) ⊆ C(y)
4. ∀ₓ(RESTRICTION R(MINn))(x) ≡ ∃ᵦ'R(x, y)
5. ∀ₓ(RESTRICTION R(MAXm))(x) ≡ ∀ᵦm > m ⊆ ∃ᵦ'R(x, y)

The concepts are hierarchically organized by a specialization relation defined in the usual way.

4 Implementation

QUARK has been implemented in Xerox Common Lisp on the Siemens 5872 (equivalent to the Xerox 1186). It is constituted by two distinct modules. The Horne clause interpreter SN-HCPRVR is used to store and retrieve assertions in the form of Horn clauses from the knowledge base. The Supervision module has a twofold task: to translate IRS expressions into formulas that SN-HCPRVR can understand, and to return to the caller a more useful form of answer than the one returned by SN-HCPRVR. (More on this below.) Both modules use the terminological reasoner QUIRK.
4.1 SN-HCPRVR

SN-HCPRVR is based on HCPRVR, a program developed by Chester [5]. The control structure of HCPRVR has not been modified; neither have the binding mechanism. The modifications we have introduced are briefly listed below, and described in more length in the following subsections.

1. The clauses recognizes as special the clauses used to represent concepts and roles.

2. The data base has been organized as a context forest[2].

3. Within each context, the set of assertions has been organized in such a way that the facts relative to a specific individual can be accessed using that individual as an index.

4. The unification algorithm has been modified to take into account the type of the individuals.

5. QUIRK is used whenever ‘taxonomic’ inferences are required.

Epistemological Clauses

SN-HCPRVR does not impose any restriction on the format of the clauses. It does, however, handle in a special way those clauses the functor of whose head is one of role, isa and slink - that is, those clauses which are used to translate roles, concepts and logical predicates of IRS. These clauses are called epistemological clauses (mostly for historical reasons). ‘Handle in a special way’ means that the epistemological clauses are individual-indexed (see below) and a specialized unification algorithm is used.

The predicates whose functor is one of those listed above have a fixed structure which for predicates with functor role, for example, is

$$\text{role}(R, I, F, E, T, O),$$

where R is the role, I is the individual about which the predication expressed by the role is made, F is the filler of the role, E is the individual of the domain such that $[R(I, F)] * (E)$, T is the temporal interval of validity of the fact, and O is the opinion about the fact. (See [15] for more details.) The structure of the predicates with functor isa is the same, but F is a ‘dummy’ placeholder, and E is the same as I for all the instances of states and events.

[2]The newest versions of HCPRVR do have a context structure, albeit simpler than that of SN-HCPRVR.

4 IMPLEMENTATION
The Contexts

A context is a collection of assertions. Contexts are used to partition the knowledge base, to support alternative reasoning, and to represent the content of the mental states of the agents. Both evaluation and assertion always take place with respect to a context.

The contexts are related by two kinds of relations: extension and accessibility. A context $C'$ is an extension of the context $C$ if all the assertions of $C'$ are inherited from $C$. A context $C$ is accessible from the context $C'$ if $C$ contains the assertions describing the mental state at a certain time interval $i$ of an agent $a$ in $C$. $C'$ in this case does not inherit assertions from $C$.

Indexing of the Data Base of Assertions

The epistemological clauses are stored within each context in an individual-oriented way, in the sense that when

1. the individual (filler) argument of an epistemologic predicate is a constant $c$ or has been bound to a constant $c$, and
2. the predicate argument $p$ has also been bound,

the constant $c$ is used instead of the functor as the index in the data base associated with the current context. The clauses which are returned to attempt an unification with the goal include then

1. all the epistemological clauses in the current context with predicate $p$ and with individual (filler) argument equal to $c$
2. all the epistemological clauses with a variable or $p$ as a predicate argument and with a variable in individual (filler) position.

The traditional indexing methods are used when the conditions mentioned above do not apply.

This indexing greatly reduces the number of clauses which the matcher has to unify with the goal, and prevents unnecessary unification attempts. The generation of natural language descriptions (like the bond issued by Dresdner Bank) is another task for which it is essential to be able to reach all the facts about an individual at once.

The Unification Algorithm

The records associated with each variable in the environment include a type field, which is set when a constraint is posted on the argument (QUARK has simple constraint-posting capabilities, which include the capability of constraining the type of variables.) and used to restrict the unification of variables as follows:
1. A variable $x$ can be bound to a constant $c$ only if either of their types is a specialization of the other. The more restrictive type is then used as type of $x$.

2. Two variables $x$ and $y$ unify only if either of their types is a specialization of the other. The more restrictive type is then used as type of both.

Two epistemological predicates match iff their time intervals of validity intersect; the more restrictive time interval is used. The opinions are also combined on unification, using the tables presented in [16].

**Use of the Taxonomic Reasoner**

QUIRK is used

- To evaluate ground clauses corresponding to concepts, like, e.g., MAMMAL(Fido). In this case, the immediate classification $C$ of Fido is retrieved, and QUIRK is called to check whether $C$ specializes MAMMAL.

- To verify the consistency of the assertions before storing them. For example, before storing an assertion like HAS-OWNER($x$, $y$) the value and number restrictions on HAS-OWNER are checked.

**4.2 The Supervision Module**

The supervision module controls the input-output behavior of QUARK. The interface to QUARK is represented by the two functions TELL and ASK (there is of course a host of auxiliary functions, who need not be mentioned here). Both TELL and ASK are calls to the supervision module, which translates the IRS formulas into clausal form and then calls SN-HCPRVR. The result is then translated back to IRS. If the set of clauses cannot be proved, the supervision module tries to determine why (e.g., whether some restriction has been violated) and to return the most informative answer to the caller.

**Translation**

The translation of an IRS formula into clausal form is a recursive process which goes from the outside in. The result is a conjunction of clauses which is then asserted, or which SN-HCPRVR attempts to prove.

For example,

$$(\forall x: \text{SAVINGS-ACCOUNT}(x)), (\forall y: \text{OWN}(y))$$

$$\text{HAS-OWNER}(y, \text{sue}) \land \text{HAS-POSSESSION}(y, x)$$

is translated in the following conjunction of clauses:
isa(SAVINGS-ACCOUNT, x, x, E1, T1, O1),
isa(OWN, y, y, y, T2, O2),
drole(HAS-OWNER, y, sue, E2, T2, O2),
drole(HAS-POSSESSION, y, x, E3, T2, O2).

Producing Informative Responses

The most important task of the Supervision Module is to return to the caller of TELL or ASK an informative answer, by which I mean an answer which tells the caller whether the query/assertion is inconsistent with the current state of the knowledge base and why.

The answers returned by TELL and ASK are \(< opinion, justification >\) pairs, where the opinion is one of TRUE, FALSE, UNKNOWN and CONTRADICTORY, and the justification is a list of the assertional and/or terminological knowledge used to determine the value of OPINION. The use of these pairs is discussed in [15].
References


