Knowledge Representation Language *QUIXOTE*

ABSTRACT

QUIXOTE system provides important facilities required for knowledge information processing, such as knowledge representation and inferences. *QUIXOTE* also provides basic functions for constructing an integrated knowledge-base management system on top of Kappa-P, a nested relational DBMS.

KEY FEATURES

QUIXOTE is a language for deductive object-oriented databases (DOODs), and can be seen as an extended logic programming language with its object-orientation features, subsumption constraints, and hierarchical modules.

Object Identity: Using extended terms (object terms), representing intrinsic properties of objects, as object identifiers

Subsumption Constraint: Using subsumption relations among object terms as constraints for properties

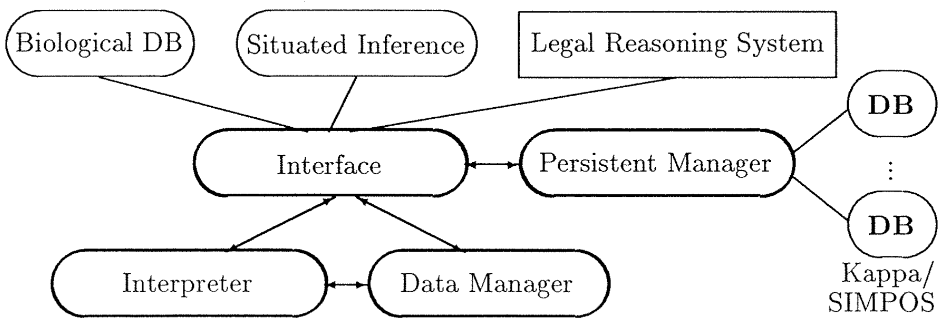
Property Inheritance: Using subsumption relations for property inheritance among objects including exceptions and multiple inheritance

Module: Introducing the modules having object terms as their identifiers in order to modularize knowledge bases, and the inter-module (submodule) relation for defining hierarchical structure of knowledge bases

Rule Inheritance: Importing and exporting rules among modules by submodule relations and set-theoretical operations including exception specified by modes of rules

Conditional Query: Introducing queries having additional assertions to a knowledge base, and answers with assumed constraints on properties

SYSTEM CONFIGURATION



Demonstration

QUIXOTE is not only a knowledge representation language for deductive object-oriented databases, but also a constraint logic programming language, database programming language, and situated programming language.

We show the basic features and the effectiveness of *QUIXOTE* through the following four demonstrations:

- Basic Features of *QUIXOTE*
- Application to Situated Inference
- Application to Molecular Biological Databases
- Legal Reasoning System TRIAL

Basic Features of *QUIXOTE*

Objects

The concept of an object plays a central role in *QUIXOTE*. An object consists of an object identifier (OID) and a set of attributes (properties). An OID is represented in the form of complex objects. For example,

$$apple, \quad apple[color = red], \text{ and } apple[color = red, area = nagano]$$

are OIDs. The first represents the concept of *apple*, while the latter two represent sub-concepts of *apple* with intrinsic attributes. Extrinsic attributes are written on the right hand side of "/" as follows:

$$apple/[color = red]$$
$$apple[color = red]/[area = \{nagano, aomori\}, weight = heavy]$$

Objects can be intensionally defined by rules as follows:

$$path[from{=}X, to{=}Y] \Leftarrow arc[from{=}X, to{=}Y]$$
$$path[from{=}X, to{=}Y] \Leftarrow arc[from{=}X, to{=}Z], path[from{=}Z, to{=}Y]$$

Subsumption Relation and Property Inheritance

Subsumption relation $\sqsubseteq$ is defined among OIDs.

$$apple \sqsubseteq fruit, \quad fruit \sqsubseteq plant$$

This is extended to the relation among complex objects: $apple[color = red] \sqsubseteq apple$. Properties are represented in the form of constraints based on the subsumption relation and Hoare's ordering relation $\sqsubseteq_H$ defined by $\sqsubseteq$:

$$apple/[family \rightarrow rose, area \leftarrow \{nagano, aomori\}]$$
$$\Longleftrightarrow\ apple|\{apple.family \sqsubseteq rose, apple.area \sqsupseteq_H \{nagano, aomori\}$$

Properties are inherited according to the ordering of the lattice constructed by $\sqsubseteq$:

$$apple/[color = red, family \rightarrow rose]$$
$$\Longrightarrow\ apple[color = green]/[family \rightarrow rose, color = green]$$

Exception is defined by intrinsic attributes and multiple inheritance is processed by merging constraints.

Module and Inheritance

A module is defined as a set of rules:

$$\begin{array}{l} west :: cider/[source = apple] \\ usa :: cider/[alcohol = no] \\ uk :: cider/[alcohol = yes] \end{array}$$

Submodule relation $\sqsubseteq_S$ is defined among modules and rules are inherited according to the relation.

$$\left.\begin{array}{l} usa \sqsupseteq_S west \\ uk \sqsupseteq_S west \end{array}\right\} \Longrightarrow \left\{\begin{array}{l} usa :: cider/[source = apple, alcohol = no] \\ uk :: cider/[source = apple, alcohol = yes] \\ japan :: cider/[source = soda\_pop] \end{array}\right.$$

Exception and multiple inheritance are also defined as for rule inheritance. Modules can be dynamically generated by parametric modules.

Query and Answer

A $\mathcal{QUIXOTE}$ program (database) consists of control information, definitions of the subsumption relation, definitions of the submodule relation, and definitions of the rules. A query is processed as follows:

- Additional rules and relations can be added to a database with a query.

  if $west :: cider/[process = ferment]$ then ?-$uk : cider/[process = X]$

- An answer is returned with assumption related to attributes.

  ?-$japan : cider/[alcohol = yes]$ $\Rightarrow$ if $japan : cider.alcohol = yes$ then yes

- The derivation process of an answer can be returned as the explanation.

Updating to a database is also specified in the program and is processed by an nested transaction logic. All objects in a $\mathcal{QUIXOTE}$ program are persistent and stored in a Kappa database or a file.

Situated Inference

Objectives

In *natural language understanding*, we often need to represent knowledge and information which depend on *situations*. The required mechanisms for situated inferences are summarized as follows:

- Environmental parameters
- Circumstantial rules of inference
- Making implicit parameters explicit

Advantages of *Quixote*

The requirements for situated inference are naturally realized with the following basic mechanisms in *Quixote*:

- Situation as module
  $s \models \sigma \iff s :: o$
- Parameter set as an object term with attributes (feature structures)
  $\ll rel, agent, \cdots \gg \iff o[rel/\mathrm{xxx}, agent/\mathrm{yyy}, \cdots]$
- Circumstantial inference rule as a rule with modules
  $s_0 \models \sigma_0 \leftarrow s_1 \models \sigma_1, s_2 \models \sigma_2, \cdots \iff s_0 :: o_0 \Leftarrow s_1 : o_1, s_2 : o_2, \cdots$
- Explication of hidden parameters by constraints
  $s :: o \Leftarrow \cdots \| C \iff s' \models \ll \sigma, C \gg$
- Offer of circumstantial information by a conditional query

Outline of Demonstration

- **Situated inference with perspectives**

  Variations in tense and aspects are considered differences in perspective toward the same situation. The inference system explicates the ambiguity of a phrase, integrating such perspectives for each lexical item that may still be temporally ambiguous by itself. We show the example of Japanese '*-teiru*', which can have three different meanings (progressive, resultant, or experience), depending on the context.

- **Inference with knowledge and belief**

  The same rule may infer different results, depending on hidden knowledge and beliefs. We analyze the following examples, where the result reflects each speaker's different knowledge.

  If Bizet and Verdi are compatriots, then Bizet is Italian.
  If Bizet and Verdi are compatriots, then Verdi is French.

Application to Molecular Biology

Objectives

As the amount of molecular biological data is explosively increasing, it is necessary to develop automatic analysis and to represent various knowledge as rules that can be automatically utilized. The knowledge in protein databases that needs to be represented is as follows.

- Protein Function Representation
- Integrated Representation of Existing Public Databases
- Tentative Representation of Unclassified Data

Advantages of *QUIXOTE*

The advantages of *QUIXOTE* are shown through our demonstration, except for a couple of the advantages of using modules as follows:

- Intrinsic knowledge of substance/species, and every public database
- Partitioning of inconsistent data

Outline of Demonstration

- **ProSite**

  Show how the *zinc finger* (a kind of functional motif) is represented and used in *QUIXOTE*, in comparison with the relational model.

  - Flexibility of Schema
  - Query Processing Facility

- **Functions of Cytochromes**

  Show how the *electron transfer of cytochromes* (metabolic reaction) is represented in *QUIXOTE*, by using the module and subsumption relation.

  - Precision Customization of Answer by Module Facility
  - Thesaurus Representation by Object Hierarchy

- **Citric Acid Cycle**

  Show how the *metabolic pathways* are represented in *QUIXOTE*, by showing the differences between the *citric acid cycle* and the *glyoxylate cycle*, both of which are metabolic reactions.

  - Representation of Path with Loops

TRIAL: A Legal Reasoning Experimental System

Objectives

We deal with a legal reasoning problem as an application of knowledge base and knowledge representation technologies to a legal field. Legal reasoning consists of three components, that is, fact recognition, statutory interpretation, and statute application. Since it is hard to treat fact recognition at the present level technology, TRIAL in *QUIXOTE* aims to perform statutory interpretation as well as statute application, especially analogical interpretation which uses relevant existing precedents.

Therefore, given a new case, TRIAL gives possible judgments by using existing precedents which are similar to the new case, and constructs arguments that lead to a judgment.

Advantages of *QUIXOTE*

For the purpose of analogical interpretation, the following *QUIXOTE*-supported features are useful:

- By type hierarchies, the taxonomy hierarchies of legal concepts can be dealt with easily in TRIAL.
- By conclusions with assumptions, TRIAL can give judgments with assumptions. This also means that missing information can be supplied.

Furthermore, the following features are helpful in offering an environment for thought experiments with legal reasoning:

- By queries with hypotheses and module hierarchies, hypothetical queries can be asked experimentally in TRIAL.
- By a solution explanation feature, TRIAL can construct arguments for verification of judgments.

Outline of Demonstration

We demonstrate a process in which experts or semi-experts construct the desired argument by application of TRIAL to the *karōshi* (or death from overwork) problem.

- They gain a goal judgment by asking TRIAL a hypothetical query.
- They verify whether the arguments constructed by TRIAL are adequate.