DESCRIPTIONS AS CONSTRAINTS IN OBJECT-ORIENTED REPRESENTATION

Luc STEELS
Artificial Intelligence Laboratory
Vrije Universiteit Brussel
Pleinlaan 2 - 1050 Brussel (BELGIUM)

ABSTRACT

A motivation is given to introduce indefinite descriptions. Parts of a description language are presented. Mechanisms for the interpretation of indefinite descriptions are briefly discussed.
KEYWORDS : Knowledge representation, AI programming languages, object-oriented systems, description languages, reasoning, constrain! propagation.

1. INTRODUCTION

A form in LISP or a term in PROLOG can be viewed as a description, i.e. an expression which refers to an entity in a domain of discourse. Forms and terms are both definite descriptions. They are subject to two restrictions

(i) The uniqueness condition : A descriplion may onJy denote a unique referent, it cannot be ambiguous. For exampJe , (Fat her John) is allowed because John can only have one father, but (Brother John) is not allowed because John may have more than one brother and it is therefore not clear which one is intended.

(ii) The computability condition: when the referent of a description is needed during interpretation, it should be computable. For example, the expression (Father John) must be computable when it is needed otherwise an error condition oc cur s.

Variables can also be viewed as descriptions. Typically, the uniqueness and computability conditions hold for then too : a variable may have only one value (within a given environment of course) and the value should be retrievable when needed, otherwise an error condition (unbound variable) results.

Descriptions that may have more than one possible referent or whose referent is not computable at the time it is needed are called indefinite. We want to develop a system where definite as well as indefinite descriptions can be used.

For example, (Greaterthan 10), which denotes an unknown number greater than 10, or (Divisor 12), which denotes a divisor of 12, i.e. 1,2,3,^,6 or 12, are indefinite descriptions which would be allowed in the system. Note that the conjuction of (Greaterthan 10) and (Divisor 12) uniquely refers to 12. In other words, the conjunction of indefinite descriptions may be definite.

A definite description provides a deterministic method to compute the referent when needed. An indefinite description expresses a constraint on its referent. Computation with indefinite descriptions consists of operations over constraints. For example, (Greaterthan 10) expresses the constraint that the referent has to satisfy the predicate (lambda (X) (> 10 X)). (Divisor 12) introduces the constraint that the referent has to be a member of (1,2,3,)< , 6 , 1 2) . The predicate can be applied to filter members out of this set, so that tho conjunction of the two descriptions yields 12 .

In recent years there have been some proposals that are relevant to this researdi Concepts like lazy evaluation (Henderson and Morris, 1976), or futures (Hewi1 1 , 1 977) make it possible to delay evaluation until needed or until sufficient information is available.

The concept of a logical variable as used in PROLOG (Kowalski, 1977) can be viewed as a way to relax the computability condition, because a logical variable can be used even though its referent is not known or only partially known. Note however that a logical variable can be bound to only one object at the time, which would make it necessary to backtrack 5 times before 12 yields a successful match in the previous example (but see Kornfeld, 1983).

Because the evaluation process needed to deal with indefinite descriptions resembles techniques used to implement constraint propagation, there are some interesting relations to constraint languages as well (cf. Borning (198O), Steele and Sussman (1980)

There are four motivations for studying indefinite descriptions. First they can be incorporated in ordinary programming languages, particularly in object-oriented languages in which definite descriptions

already play an important role. Second they can be incorporated in knowledge representation languages, particularly in the recent generation of description languages (see e.g. Winograd (1982), Attardi and Simi (1982)) .

Third, because indefinite descriptions feature prominently in natural language, the relation between formal languages and natural languages promises to become more transparent. Finally, indefinite descriptions can be used as a component of query languages for databases. It would make it possible that a user supplies constraints in the form of indefinite descriptions to refer to an object in the database-.

In this short paper a full treatment of the description system is impossible. Instead we only introduce some basic linguistic constructs and mention some properties of the evaluation mechanism.

1. DOMAIN VS. REFERENT

The first key idea is that we make a distinction between the domain of a description and its referent. The referent is the element denoted by the description. The domain is the set of possible elements out of which the referent has to be chosen. For the description 'uneven' the domain is the set of uneven numbers and the referent is an element out of this set - although it is unknown which one. The domain of 'the brother of John' is equal to all brothers of John. The referent is one element out of this set.

The distinction between domain and referent allows the construction of a clear set-theoretic semantics for a description language. By making it possible to give partial descriptions of the domain, it raises the expressive Power of the language Also the explicit representation of constraints on the domain makes the deduction more powerful. Often we know a lot about the domain of a description but not its referent. By operating at the level of domains, we can sometimes reduce the domain until it is a singleton, i.e. until there is a unique referent left. These points will be illustrated in the rest of the paper. First we introduce description types

2. NAMES AND SPECIAL OBJECTS

The simplest form of a description is a name which uniquely indentifies an element in the domain of discourse. A name is an atom or a datastructure containing only names as elements. Sequences and sets are considered to he primitive datastructures. For example, JOHN, [JOHN MARY JAMES] and (JOHN MARY JAMES) are examples of names.

There are three special objects in the system. The undefined object or all-object, the null-thing or empty object, and the overdefined object. The names of these objects are T(top), NIL, and 1. (.bottom;. ?he domain of T is the universe of discourse. The domain of NIL is the empty set. ?he domain of 1 is overdefined. When at a :ertain point a subexpression refers to the)verdefined object, then the whole expres-;ion refers to the overdefined object.

3. DESCRIPTIONS BASED ON CONCEPTS

The second type of descriptions is of the form <concept> or (<concept> < arg > . . . ;arg >) for n > 1. A concept may either be L function, in which case a unique referent :an be computed given referents as argulents. For example, (+5 10) is a description with referent 15. But a concept need tot be a function. For example, (Divisor 2) is a description with possible refe-•ents 1,2,3,4,6 and 12. Divisor is a concept but not a function.

4. DESCRIPTIVE CONNECTIVES

The descriptive connectives dAnd, dOr, Not, dEither and dAncinot are used to combine descriptions. They should not be confused with the propositional connectives used to combine statements in predicate calculus .

The descriptive connectives reflect set-theoretic relations between the domains of the component descriptions. The referent of (dAnd d1 d 2) is an element out of the intersection of the domains of d1 and d 2 . The referent of(dOr d1 d 2) is an element 5ut of the union of the domains of d1 and d2. The referent of (dEither dl d2) is an element of the domain of dl or of the domain of d2 but not of both. The referent of (dAndnot dl d2) is an element out of the set-theoretic difference between the domains of d1 and d2. (dNOT d) is an abbreviation of (dAndnot T d), i.e. dNot indicates a set-theoretic difference with the domain of the all-description. Thus (dNot female) is equivalent to (dAndnot T female).

It is easy to prove that T and NIL act as the identity and zero-element for these connectives. Analogues exists also for the other propositional laws, such as De Morgan's.

5. DESCRIPTIONS OF THE DOMAIN

There are a variety of things that could be know about the domain of a description. We want to have constructs that are able to express this partial information. Here are some examples :

ENUMERATION OF THE POSSIBLE MEMBERS. An expression of the form (Element-of Xi....X expresses the constraint that the referent has to be either X1.... or X . For example, an alternative description for (Brother John) could be (Element-of George James).

PARTIAL ENUMERATION. A description of the form (INCLUDES X) expresses that the referent comes out of a domain that has X as a member. For example, if it is known that George is one of the brothers of John then (includes George) is an alternative description for (Brother John).

CARDINALITY OF DOMAIN. The description (Number-of X) with X an integer, indicates that the cardinality of the domain is equal to X. For example, if it is known that John has two brothers, then (Brother John) can be described as (Number-of 2).

The deduction rules for the descriptive connectives include rules for dealing with such domain descriptions. For example, the referent of (dAnd (Brother John) (Friend Frank)), where (Brother John) is (Element-of George James) and (Friend Frank) is (Element-of George Mary), is equal to George because George is the only element in the intersection of the domains of two descri pt ions.

6. VARIABLES

Variables are descriptions which start out as indefinite members of the universe and gradually assume their domain as constraints accumulate. Variables are preceded by the symbol : and are lexically scoped within the expression in which they occur, although they may occur anywhere in the description. For example, in (dAnd : Y 5), or its equivalent (dAnd 5 : Y), the referent of : Y will be equal to 5• Variables in the description system thus behave like logical variables.

7. CONVERSE DESCRIPTIONS

If (Father George) i-s a~~de sc r i pt ion for John, then (with Father John) is an alternative description for George. A description of the form (with <concept> <description> is called a converse description, because it denotes the converse of a concept.

8. EVALUATION

The goal of evaluation is to find the name of the referent of a description, i.e. a value. When a description is definite, its referent can be computed and evaluation proceeds as ordinary applicative evaluation When the description is indefinite, the result from evaluation is a collection of constraints on the referent, called a constraint cluster.

These constraints take the form of a predicate that the referent has to satisfy, generators which could start enumeration of the domain if needed, and constraints on the domain such as a list of its members a list of the elements not in the domain, a partial list of the members, the cardinality of the domain, etc.

The evaluator will attempt to proceed with the computation even though partial results are constraint clusters. Deduction rules for the descriptive connectives operate over constraint clusters.

For example, if the constraint clusters of two descriptions contains predicates, then the constraint-cluster of the conjunction of the two descriptions will contain the AND-conjunction of the two predicates. Thus (dAND (Greaterthan 10) (Lessthan 5)) results in ((Predicate (lambda (x)(and (> x 10)(< x 5))))).

The evaluation process will also attempt to apply functions to constraint clusters. For example, when an explicit domain is known, computations can be performed by mapping the function. For example, (+(element-of 1 2) (element-of 3 4) is equal to (element-of 4 5 6).

Note however that it is possible to specify descriptions whose referent will not be computable because it would require the introduction of much more knowledge. For example, the description (dAnd Even Prime) has only one referent, namely 2, but this cannot be determined from knowing predicates or generators on the component descriptions themselves.

9. CONCLUSION

We argued for the introduction of indefinite descriptions, sketched some linguistic constructs and briefly indicated a possible evaluation.

REFERENCES

[1] Attardi and Simi (1982) Semantics of inheritance and attribution in the description system OMEGA. MIT-AI lab. Memo. *642*

[2] Borning, A. (1979) THINGLAB, A. Constraint oriented simulation laboratory. Xerox Parc Report 55L~79_3

[3] Hewitt,Carl (1977) Viewing control structures as Patterns of Passing Messages. AI Journal 8, n° 3. PP. 323-364

[4] Henderson, P. and J. Morris (1976) A lazy evaluator. Proceedings of the 3d POPL Symposium, Atlanta George

[5] Kornfeld, B. (1983) Equality for PROLOG. IJCAI-83, Karslruhe

[6] Kowalski, R. (1978) Logic for problem Solving. North-Holland, Amsterdam

[7] Steele, G. and J. Sussman (1980) Constraints. MIT A.I. Las Memo 502

[8] Winograd, T. (1983) Language as a cognitive process. Prentice-Hall, Engle wood Cliff.