LESK: A LANGUAGE SYNTHETIZING NATURAL LANGUAGE, COMPUTER LANGUAGE AND LOGIC

Douglas Skuce

Dept. of Computer Science, University of Ottawa, Ottawa, Ontario, Canada

Introduction

LESK (Language for Exactly Stating Knowledge) is intended to bridge the considerable gap between natural language (NL), computer language (CL) and logic. It is desirable to do so for the following reason. To implement any non-trivial computer system for some problem domain, one must first have a very clear understanding of the domain concepts. Usually the implementors do not have this knowledge, and must struggle to obtain it from the domain experts, who are usually not computer experts. Hence a means of precise and efficient knowledge expression would be very useful. No such tool exists today and we therefore seek to develop one. LESK is a first approximation to such a medium for knowledge capture.

Anyone who has participated in the design of any system where knowledge transfer from experts was involved will appreciate the need for such a tool. The existing tools are the following: NL, mathematical concepts and notation, computer concepts and notation, plus assorted devices like diagrams, pictures, models, etc. By far the major tool is NL however, which is unfortunately very poorly used by most people, particularly recent university graduates. The most frequent errors include the use of undefined terminology, synonyms or homonyms (whose status as such must be guessed), unclear syntax, logical

errors (such as outright contradictions), undefined relations between concepts, and simple lack of good organization. LESK forces the user to express him/herself in ways designed to minimize such errors. A LESK system (an interactive program which "understands" LESK) would further reduce such errors.

Criteria for LESK

LESK should

a) be readable by most university graduates in science, medicine, law or business, for example;
b) be writable by most people with a basic knowledge of set theory, logic and computers;
c) have a simple semantics expressible in first-order logic;
d) be implementable using artificial intelligence techniques;
e) be sufficiently general to be usable in the subjects of interest to those listed in a).

The present design of LESK has been shown to meet all of these criteria.

Design of LESK

LESK should be viewed as a language for making assertions about sets, tuples, sequences, functions, relations, procedures and other simple mathematical concepts using an English-like syntax (many other NLs could be used; translation of LESK-based knowledge should be easier than translation of NL). The user declares words or phrases either explicitly or implicitly to be nouns (or noun-like), adjectives, prepositions, or verbs (other categories have not been necessary). Nouns and noun phrases denote sets, stative verbs denote relations, action verbs denote procedures, and prepositions denote case relations. The syntax is a compromise between NL, CL, and logic. All constructs have a simple first-order logic interpretation.

An example (whose spirit should warm the heart of those who despair of ever understanding the regulations of their

institution) follows. LESK primitives are in lower case. We first illustrate noun declarations. (N.B. In the final paper, this example will be greatly expanded and explained.)

```
every PERSON:
   - has 1 NAME
   - has 1 integer called their AGE
   - has 1 ADDRESS
end.

every STUDENT:
   kinds: FULL-TIME -, PART-TIME -;
   kinds: UNDERGRADUATE -, GRADUATE--;
   - is a PERSON;
   - is ENROLLED IN 1 DEGREE PROGRAM;
   - is a STUDENT IN 1 FACULTY, SCHOOL or DEPARTMENT;
   - has 1 ACADEMIC RECORD
end.

every ACADEMIC RECORD of a STUDENT X:
   - has 1 FACULTY, SCHOOL or DEPARTMENT;
   - has 0 or more COURSE CREDITS;
   - has 0 or more COURSE SECTIONS called the COURSES-BEING-
     TAKEN BY X
end.

every COURSE CREDIT:
   - is a pair <X, Y> where X is a COURSE,
                            Y is a LETTER GRADE
end.
```

We illustrate next two stative verb declarations:

```
a STUDENT X is ENROLLED IN a COURSE SECTION Y iff:
   X is on the ENROLLMENT LIST of Y.

a STUDENT X is a STUDENT IN a FACULTY, SCHOOL or DEPARTMENT Y
   iff:
```

Y = the FACULTY, SCHOOL or DEPARTMENT of the ACADEMIC RECORD of X.

Finally, we illustrate an action verb declaration:

```
To ENROLL a STUDENT X IN a COURSE SECTION Y:
   precond: for each PREREQUISITE Z of Y there is a COURSE
            CREDIT <Z,M> on the ACADEMIC RECORD of X such
            that M is a PASSING GRADE;
            X has PAID the FEE of Y;
    add:
    negate:
    delete:
    procedure: add X to the ENROLLMENT LIST of Y;
               add Y to the COURSES_BEING_TAKEN BY X
end.
```

Concluding Remarks

LESK has been used to describe neurophysiological knowledge, electrical circuits, PASCAL, the Canadian census database, and university regulations. A partial implementation has been developed in DEC 10 PROLOG. A LESK system should be capable of answering any question (expressed in LESK) which a person could answer from the same LESK knowledge base. This clearly requires both deductive and database komponents. Systems now exist which combine these two; it remains therefore to add the LESK component, which is our eventual goal.