A DETERMINISTIC ANALYZER FOR THE INTERPRETATION OF NATURAL LANGUAGE COMMANDS

Leonardo LESMO, Daniela MAGNANI, Piero TORASSO

Istituto di Scienze dell'Informszione - University di Torino
C.so Massimo D'Aseglio, 42 - 10125 TORINO - ITALY

Abstract

This paper describes a system which translates a query in Italian language into a representation which can be immediately interpreted as a sequence of algebraic operations on a relational data base. The use of a lookshead buffer allows the system to operate deterministically. Different knowledge sources are used to cope with semantics (associated with the lexicon) and syntax (represented as pajt tern-action rules). These knowledge sources cooper ate during the query translation so that independent translation steps and intermediate representations of the command are avoided. Therefore the term "determinism" is used to mean that all the structures built during the process concur to build the final command representation.

Introduction

One of the most popular new ideas in the field of natural language processing is the "determinism" introduced by Marcus /1/. The system presented in this paper accepts Marcus' proposals for the syntaf tic analysis of the text, but extends that analysis to a real translation of an input query in Italian language by means of a more semantical structure of the actions associated to the pattern-action syntaf tic rules. These actions have the purpose of build ing representations of the constituents with which the syntactic rule is associated, in a form which allows the system to perform semantic checks by us ing the semantic information available from the lexicon.

One of the main features of the system is a co operation between declarative (lexicon and pattern part of the syntactic rules) and procedural (proce dures to perform semantic checks and the action part of the syntactic rules) knowledge sources such that the control flow is automatically transferred from one knowledge source to another without the inter vention of a central controller.

As a benchmark, the system has been specialized to act as an interface to a medical consultation program for the analysis of the liver functional assessment /2/.

Command representation

The interpretation process translates the in put command into a set of frame instantiations linlc ed together. By "frame" we mean a collection of slots, each of which has a name, may contain a val ue and may be associated with a procedure which checks the semantic correctness of the value insert ed into the slot by the grammar rules.

The system contains a set of prototype frames (currently 4: ACTIONFR, COMMANDFR, RESTRICTIONFR, CONCEPTFR) which are instantiated under request of the grammar rules. Notice that the checks associated with the slot of a given fraae F may operate on data not necessarily stored in F, but in other frames coil nee ted to F. The check procedure associated with a frame slot is executed when the slot is filled: it may happen that, at that time, some of the slots (in the same frame or not) needed to perform the checks are still empty. In this case the procedure is suspended and its suspension point is stored in a list together with information which allows the procedures associated with slot filling and frame linking to resume its execution as soon as possible.

Lexicon

The lexicon contains noise words, non-content words (e.g. prepositions) and content words. The latter are used to build the final representation of the command and the information associated with them concerns their syntactic category and (for some of them) semantic information stored as links to other content words /3/, so that the lexicon may be considered as a network which allows the system to verify the semantic correctness of the command. A spelling correction is performed to detect and correct possible typing errors.

A more detailed description of the lexical knowledge source is reported in /4/.

Syntax

The syntactic knowledge is represented by means of Syntactic Rule Packets (SRP) each of which coit tains a set of pattern-action rules which specify the different forms a particular constituent may as sume. The pattern part of the rules allows the syf

tem to deterministically identify the rule of the Packet which has to be applied to analyze the input. The pattern consists in a sequence of syntactic cat egories or constituents which have to be matched on the lookahead buffer, which is filled under request of the syntactic knowledge (CAT statements; see be low) with the exact portion of input string needed to discriminate among the rules of the Packet. For the sake of efficiency, the set of patterns concern ing the same SRP have been represented by means of a discriminant net which guides the filling of the buffer.

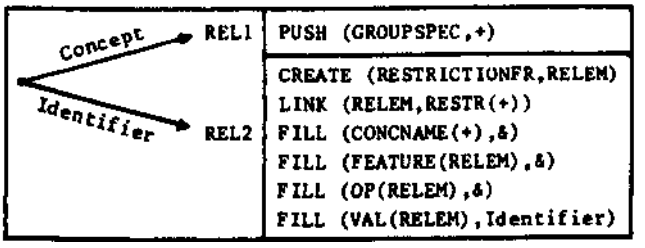
The action part of the rules is a procedure in volving imperative statements:

- FILL. It fills a slot of a frame and triggers the checks associated with that slot.
- CREATE. It builds a new instantiation of a frame.
- LINK. It connects aframe instantiation to a slot of another one.
- CAT. It checks the category of the current word of the lookahead buffer. If all the elements in the buffer have already been scanned, a new word is transferred into it from the input string. Ill cidentally, the categories used in the grammar (and the constituents as well) are not the classi cal categories (noun, adjective, etc.) but are more semanticslly biased (concept, property, etc).
- PUSH. It involves the triggering of the SRP asso ciated with the embedded constituent that has to be analyzed and the transfer of control to the SRP handler.
- DEL. It deletes a word from the buffer.

When a FILL or LINK operations is executed,the list of suspended checks is scanned to decide wheth er some of them may be resumed. After the completion of the action part of a rule, the control is returti ed to the action which activated the SRP containing the rule whose action has been completed.

As an example of Syntactic Rule Packet, consid er the constituent ELEMENT: it refers to a group of objects defined by a concept name (e.g. Patient) followed by the conditions selecting the indiviid uals that should belong to the group, or, in case the group is restricted to a single individual, by its name (Identifier). The structure of the SRP is as follows:

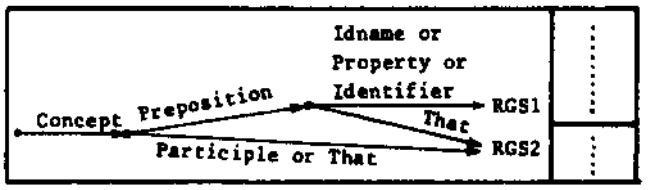
ELEMENT



The symbol + refers to the current frame instantia tion (in this case a copy of CONCEPTFR), whereas the symbol & means that the filler of the slot is not explicitly present in the input sentence; in this example all the slots containing & will be filled by the semantic check procedure associated with the slot VAL, by inspecting the semantic infor mation stored in the lexical entry corresponding to the current identifier. As regards RELEM, it is a variable which is filled by the CREATE operation with the actual name of the newly created instanti£ tion.

It may be observed that, in this case, the dis criminant net is very simple because only one word has to be examined in order to determine which rule has to be applied. In the GROUPSPEC discriminant net it is apparent that two or three lookahead words are needed to deterministically decide what rule has to be applied; for this reason the lookahead buffer size is not fixed, even if our grammar never re- quires more than four cells.

GROUPSPEC



Notice that the trvelling on the discriminant net does not imply that an input word is consumed (for example, the first word in the PUSHed GROUPSPEC is the same Concept tested by ELEMENT).

Examples and comments

One of the most important features of the sys tem is its ability to obtain the same representa tion for sentences which have the same meaning but completely different surface structures. Consider for example the sentences:

S1) Dammi il valore della bilirubina totale per i pazienti con ittero (Give me the value of total bilirubin for the patients with jaundice).

S2) Per i pazienti che hanno ittero dimmi quanto va le la bilirubina totale (For the patients who have jaundice tell me what the total bilirubin is).

S3) Quanto hanno di bilirubina totale i pazienti con ittero presente? (How much total bilirubin have the patients with jaundice present?).

For all of them the final representation is the same and is reported in fig.1.

It should be pointed out that information which is implicit in the input command has been made explicit (e.g. in RESTR3 the fact that "jaun dice" is the "name" of the symptom and in RESTR2 that the "value" of the jaundice used for patient selection should be "present", even if in the *sen* tences S1 and S2 it is understood).

The resulting representation may seem rather awkward. For example, it may not be apparent why RESTR2 is linked to an action (named SHOW) instead

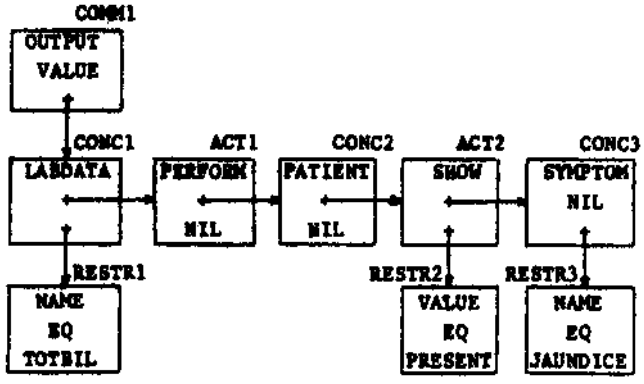


Fig.1 - Final representation of S1, S2 and S3

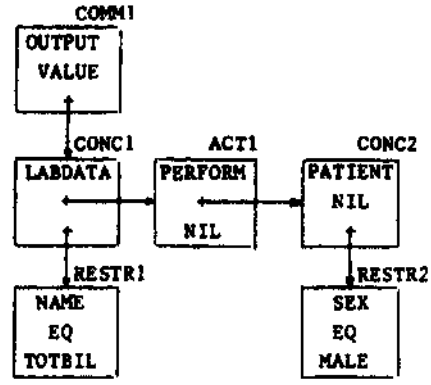


Fig.2 - Final representation of the sentence S4

of the concept it qualifies (jaundice). The advan cage of this representation is the one-to-one na£ ping between the pointers which link the frames and the algebraic operators of the relational approach to database construction. For example, each pointer to a restriction implies a SELECT operation on the relation to which the concept (or the action) is associated, each pointer connecting an action and a concept implies a JOIN operation on the two involv ed relations and so on.

It is worth noticing the difference between the representation of the sentence SI (fig.l) and the representation (fig.2) of the sentence:

S4) Dammi il valore della bilirubina totale per i pa sienti di sesso meschile (Give me the value of total bilirubin for the patients of male sex).

The difference is due to the fact that "sex" is a property of a patient and for this reason it is stored in the data base relation PATIENT, whereas "jaundice" is an identifier of the concept "symptom" (i.e. a key of the data base relation SYMPTOM), so that the value of jaundice for a given patient is stored in the connection relation named SHOW.

In fig.l and 2 the frame structure was heavily simplified for the sake of readibility; let us con sider as an example the actual structure of RESTRIC TIONFR:

RESTRICTIONFR (FEATURE, OP, VAL, FEATURE!, OP!, VAX!, BACK, RESTRCHAIN)

The slots composing the frame are of three kinds: External, Internal, Link; the External slots (e.g. FEATURE, OP, VAL) contain the actual lexical entries appearing in the input sentence; the Internal ones (e.g. FEATURE!, OP!, VAL!) concern the correspond ing internal representation, which will be used to" access the data atored in the data base relations; the Link slots (e.g. RACK, RESTRCHAIN) contain the pointers to other instantiations.

In RESTRICTIONFR the slot FEATURE contains the name of the attribute used to perform the selection, whereas the slots OP and VAL contain the operator and the linguistic or numeric value involved in the SELECT operation. When the string "di sesso raaschi le" in the sentence SA is analysed, the slots FEA TURE and VAL are filled with "sesso" and "meschile" respectively, whereas the slot OP is filled with & (undefined). The semantic checks associated with FEATURE end VAL provide not only the filling of FEA TURE! end VAL! with the internal representation of "sesso" and "meschile" (i.e. SEX and MALE),but also the filling of OP! with EQ (the choice of EQ depends on the information associated with "maschile") thus leading to the situation of RESTR2 shown in fig.2.

The slot BACK contains a backward pointer to the CONCEPTFR (or ACTIONFR) instantiation for which the RESTRICTIONFR instantiation acts as a condition. Finally, the slot RESTRCHAIN may contain a pointer to another instantiation of RESTRICTIONFR which acts as a further condition for the selection (i.e. the two - or more - conditions are ANDed).

The system has been implemented in LISP on a DEC-10; the lexical and syntactic processes have been tested fully, wherees the procedures associat ed with the semantic checks are currently under debug.

References

/1/ Marcus, M.P.: A Theory of Synctactic Recognition for Natural Language. MIT Press, Cambridge, Mass., 1980.

/II/ Lesmo,L.; Saitta, L.; Torasso, P.: Computer-Aided Evaluation of Liver Functional Assessment, Proc. 4th Symposium on Computer Applications in Medical Care, Washington, D.C., 1980, pp.l81rl89.

/3/ Heidorn, G.E.: Augmented Phrase Structure Gram mars, in Schank, R.C. & Nash-Webber, B. (eds.): Proc. Theoretical Issues in Natural Language Processing, Cambridge, Mass., 1975, pp.1-5.

/4/ Lesmo, L.; Magnani, D.; Torasso, P.: Lexical and Pragmatic Knowledge for Natural Language Analysis, 1981 Int. Conf. on Cybernetics and Society, Atlanta, Georgia, Oct.1981.