INSYST: An Automatic Inserter System for Hierarchical Lexica

Marc Light Sabine Reinhard Marie Boyle-Hinrichs
Universität Tubingen, Seminar für Sprachwissenschaft
Kleine Wilhelmstr. 113, D-7400 Tübingen
{light,reinhard,meb}@arbuckle.sns.neuphilologie.uni-tuebingen.de

1. Introduction

When using hierarchical formalisms for lexical information, the need arises to insert (*i.e.* classify) lexical items into these hierarchies. This includes at least the following two situations: (1) testing generalizations when designing a lexical hierarchy; (2) transferring large numbers of lexical items from raw data files to a finished lexical hierarchy when using it to build a large lexicon. Up until now, no automated system for these insertion tasks existed. INSYST (INserter SYSTem), we describe here, can efficiently insert lexical items under the appropriate nodes in hierarchies. It currently handles hierarchies specified in the DATR formalism (Evans and Gazdar 1989, 1990). The system uses a classification algorithm that maximizes the number of inherited features for each entry.

2. The INSYST-Architecture

The following information is required by the INSYST-Classifier module: i) the features that can be inherited from each node of the hierarchy, and ii) the features of the item to be inserted. Since the answer to i) is not explicitly stated in the DATR specification of a node, three modules preprocess the input DATR theory: the INSYST-Compiler and the INSYST-Inheritance Closure modules. The INSYST-Interface to the database answers question (ii). The modules are implemented in C. Figure 1 presents a pictoral view of the interactions between INSYST modules.

2.1 The INSYST-Compiler and Inheritance Closure modules

The INSYST-Compiler reads the input DATR theory from a file, creates nodes and inserts the path-value pairs into them as they are encountered.

The Inheritance Closure module loops through the node list provided by the Compiler, calling a recursive function that "expands" path-value pairs, for each path-value pair in each node. This "expansion" is necessary because of the complex DATR inheritance mechanisms: default inheritance (a node inherits all the values for paths that start with a certain prefix from a parent node), global inheritance, embedded paths, lists, etc. In a first pass (Inheritance Closure I), all inheritances are resolved and listed, except for the global (quoted) paths. These are resolved on a second pass (Inheritance Closure II), when a node is being inserted, because the values for the global paths are taken from that node currently being inserted.

2.2 The INSYST-Classifier

The INSYST-Classifier algorithm (s. Light, forthc.) strives to maximize the number of path-value pairs a new entry node inherits while minimizing the number of parents. It uses the following heuristic: choose the parent from which the node being inserted can inherit the most path-value pairs while counting clashes between a potential parent node path-value pair and a new entry path-value pair. The algorithm is computationally tractable and always produces a reasonable solution. However, a solution involving fewer parents may exist.

3. Conclusion

By building an inserter system for DATR with its particulary complex inheritance features (default inheritance, embedded paths, etc.), we have shown the plausibility of our design. We feel that INSYST or systems like it will become a standard tool for researchers using or designing lexical hierarchies.

References

[Evans and Gazdar, 1989, 1990] Evans, Roger and Gerald Gazdar (eds.). "The DATR Papers", Cognitive Science Research Papers, U Sussex, 1989 and 1990.

[Light, forthc.] Light, Marc. "A Classifier Algorithm for Default Hierarchies", SfS-Report, U Tübingen, forthc.

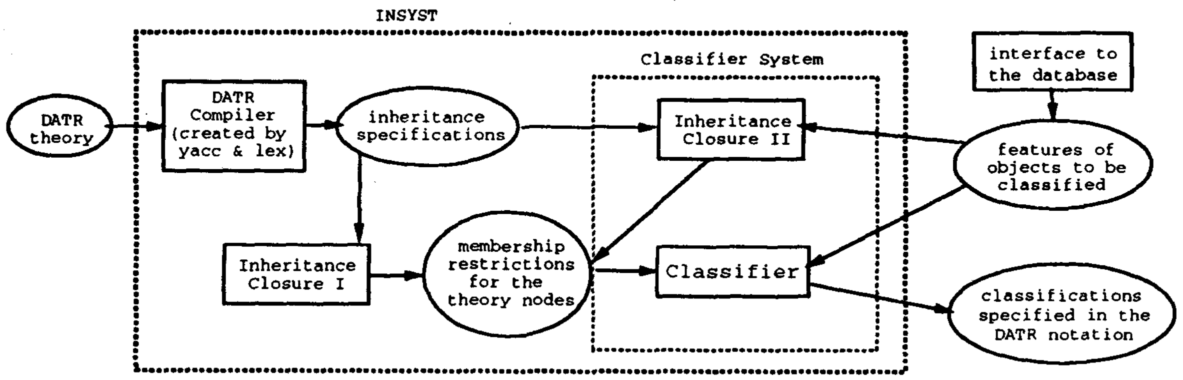
INSYST
Classifier System
DATR theory
DATR Compiler (created by yacc & lex)
inheritance specifications
Inheritance Closure II
interface to the database
features of objects to be classified
Inheritance Closure I
membership restrictions for the theory nodes
Classifier
classifications specified in the DATR notation

Figure 1: Internal Structure of INSYST