Comparison in NLIs — Habitability and Database Reality

Wolfgang Heinz, Johannes Matiasek, Harald Trost, Ernst Buchberger
Austrian Research Institute for Artificial Intelligence
Schottengasse 3
A-1010 Wien, Austria
Email: john@ai.univie.ac.at

Abstract

This paper describes the treatment of comparison and measures in DATENBANK-DIALOG,[1] a German language interface to relational databases. Besides giving a short overview of the system architecture the paper shows how design strategies support the development of a habitable system taking as example comparison and measures both of which are important for many application domains of NLIs and non-trivial from a linguistic point of view. In contrast to some former work we pay attention not only to the purely linguistic part but equally to the mapping to the underlying database model. This way we arrive at a balanced treatment of syntax as well as semantics and pragmatics.

1 Introduction

Experiments with NLIs indicate that the overall linguistic coverage of state-of-the-art systems is adequate since savings in training time outweigh problems with queries the system cannot handle. People adapt very well to grammatical restrictions in the language they use (Hendler and Michaelis 1983). Very important though is that the NLI performs in a predictable way, i.e. that it is habitable (Krause 1982, p.15ff). Users should be able to learn very fast which types of queries are acceptable. Otherwise, they will either have to face a continuously high rejection rate or—more likely, because humans adapt much better than computers—they will formulate their queries in an unnecessarily simple and inefficient way (Tennant 1980).

[1]The development of DATENBANK-DIALOG was carried out at the Austrian Research Institute for Artificial Intelligence jointly with "Software Management GmbH", Vienna and has been sponsored by the Austrian Government within the "Mikroelektronik Förderungsprogramm, Schwerpunkt S7".

However, syntactic coverage must not be judged in isolation. Queries are accepted only if they are correctly interpreted syntactically, semantically and pragmatically. While syntactic coverage depends solely on the parser of the NLI, semantic and pragmatic coverage must be considered with respect to the contents of the database to which the NLI connects.

The treatment of comparison is a linguistically interesting problem and an important issue for NLIs. Accordingly, a number of papers on the subject both from a general linguistic point of view (Rayner and Banks 1990, Pulman 1991) and for NLIs in particular (Ballard 1988) have been published. We will describe our treatment of comparison in DATENBANK-DIALOG focussing on the interaction between syntactic analysis and semantic interpetation and we will show how our design strategies lead to a habitable NLI.

2 DATENBANK-DIALOG

DATENBANK-DIALOG (Trost *et al.* 1988) is a German language interface to relational databases. The system has been fully implemented in different computer environments and been tested with a number of application domains. Currently a large field test is taking place. DATENBANK-DIALOG has been interfaced to a database about Artificial Intelligence research in Austria. Questions (in German) to that system can be sent by electronic mail (Email-address: `aiforsch@ai.univie.ac.at`) and are answered automatically.

DATENBANK-DIALOG comprises four main components. The *scanner* breaks up the natural language query into tokens. By applying a pattern matching module it can handle input following special formatting conventions such as dates (*15. 1. 1991, 15-JAN-91, 8:30pm*), amounts (*20,000, 20.000,–*), numerical data along with a unit of measure (*2,54cm, 1.0in, $20*), and abbreviations. Words are morphologically analysed. The result is passed on to the *parser* which performs syntactic and shallow semantic analysis in parallel. The parse results in one or—in case of ambiguity—more instantiated caseframes representing the query at the domain level.

The query *interpretation* consists of a mapping from domain-level (caseframes) to database-level predicates (DB-caseframes), a linearization step producing the Logical Form, and, finally, a syntactic transformation to SQL. The *answer* is produced directly by the DBMS as the result of executing the SQL query.

Unlike some other NLIs, DATENBANK-DIALOG has a separate grammar component which is completely domain-independent. This grammar was designed to make the accepted sublanguage as consistent as possible. To formulate it in a clean and concise way much effort was undertaken to incorporate recent advances of linguistic theory in its development.

An example for this strategy is the use of Generalized Quantifiers Theory (Barwise and Cooper 1981, Keenan and Stavi 1986) for the representation of Logical Form and the translation into SQL. Generalized Quantifier Theory gives a general framework for the treatment of nonlogical determiners (*seven, between 2 and 10, many*) that includes the standard logical quantifiers as special cases.

Using the results of Keenan and Stavi for natural language quantifiers (e.g. conservativ-

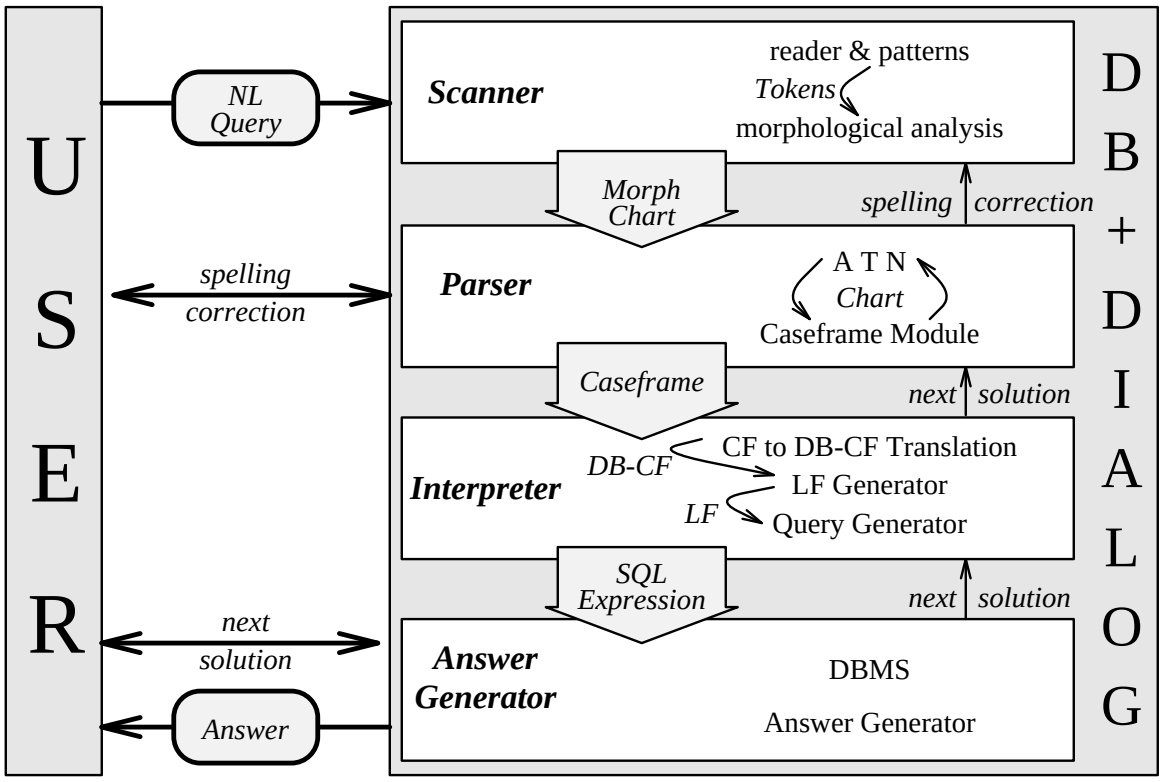

Figure 1: Architecture of DATENBANK-DIALOG

ity) a formal correspondence between GQ-formulas (our means of representing the logical form of a query) and SQL-statements (representing formulas over the relational calculus) was established and straightforwardly implemented. This gives us a sound theoretical basis for semantic interpretation and SQL generation. All extensional natural language determiners can be handled—matching the extensional nature of relational databases (for details cf. Heinz and Matiasek 1989).

3 Treatment of Comparison

We will now have a closer look at the treatment of comparison and measurements in DATENBANK-DIALOG to demonstrate our efforts to create a habitable system. Comparisons between various kinds of objects are of central concern in NLIs. They involve a relation between values associated with a dimension and units of measure. The values may be given explicitly or implicitly by derivation (thus including superlatives).

The means for expressing comparison vary widely. In linguistic terms comparison is mainly associated with gradable adjectives and adverbials. But there is not only variation of constructs on the linguistic level, but also on the underlying knowledge base—in our case a relational database system. The knowledge base is usually not designed with the use of an NLI in mind, especially since NLIs are often added later on to existing systems.

Unlike other work dealing with comparison we take into account problems that arise when coupling an NLI to a database that does not directly reflect the structure imposed by the use of natural language—be it the splitting of tables for performance reasons, or the encoding of information.

To advance habitability such properties of the realization must be hidden from the user. This is achieved in DATENBANK-DIALOG by separating the domain model from the database model and by providing an explicit translation step.

Consider some types of linguistic variation given in (1):

(1) a) *Welche Ärzte haben ein höheres Gehalt als 20.000,– ?*
(Which doctors have a salary higher than 20.000,– ?)

b) *Welche Ärzte verdienen mehr als 20.000,– ?*
(Which doctors earn more than 20.000,– ?)

c) *Welche Ärzte haben ein Gehalt von mehr als 20.000,– ?*
(Which doctors have a salary of more than 20.000,– ?)

d) *Welche Ärzte haben mehr als 20.000,– Gehalt ?*
(Which doctors have more than 20.000,– salary?)

e) *Gib mir alle Ärzte mit einem höheren Gehalt als 20.000,– .*
(Show all doctors with a salary higher than 20.000,– .)

All the utterances in (1) should map onto the same database query, e.g., the SQL statement:

(2)
```
select ID, NAME
    from DOCTOR
   where SALARY > 20000;
```

Accordingly, the interpretation of sentences (1a-e) must be the same. DATENBANK-DIALOG uses a compositional semantics and separates the lexical item (word) from the underlying semantic relation, which may be used with more than one word. In (1) we have the underlying predicate `SALARY` with two arguments describing the thematic roles (deep cases) `RECIPIENT` and `VALUE`, both with their appropriate semantic restrictions:[2]

(3)
```
pred: SALARY
    a1: RECIPIENT (sem-restr PERSON)
    a2: VALUE     (dimension MONEY)
```

The predicate `SALARY` shown in (3) is associated with different words in the lexicon, among them the verb *verdienen* (earn) and the noun *Gehalt* (salary). Every lexical entry gives the syntactic restrictions on the fillers of the arguments. The syntactic information associated with lexical entries binds `RECIPIENT` to a nominative NP (in active contexts) in the case of the verb and to a genitive NP or a PP with preposition *von* in the case of the noun.

[2] domain level caseframes like (3) and subsequent examples are given in simplified versions

(4) a) *Wieviel verdient [$_{NP_{nom}}$ Dr. Haid] ?*
(How much does Doctor Haid earn?)

b) *Gib mir das Gehalt [$_{PP}$ von Dr. Haid] ?*
(Show me the salary of Doctor Haid?)

The picture gets even more complicated when the `VALUE` is specified explicitly in the query. Throughout (1t)he comparison consists of the comparative *höheres / mehr* and the object it is compared to (the value) *als 20.000,–*. In (1a) the comparative is realized as an adjective phrase embedded in a noun phrase, whereas in (1b-d) both comparative and value are adjacent, forming a common syntactic entity: an adverbial phrase in (1b), a postnominal PP modifier in (1c), and a complex determiner phrase in (1d).

All of these constructions map onto the same semantic representation, in our example a relation (>), a value (20.000) along with a dimension (money) and a unit (Austrian Schillings), and a compared object. This enables a uniform treatment from a semantic point of view. Note that because the unit forms part of the representation, comparisons with values given in different units (e.g. $ or DM) are possible. We will return to this aspect later when discussing the relation between domain and database models.

3.1 Derived Comparison

Up to now we have assumed a constant as value. But compare sentence (1b) with:

(5) Welche Ärzte verdienen mehr als Dr. Haid?
(Which doctors earn more than Doctor Haid?)

In (1b) the value *20.000* is given explicitly, resulting in the interpretation (2). In contrast, in the structurally similar query (5) the value is specified only implicitly by referring to the salary of Doctor Haid. The resulting SQL query should be[3]

(6)
```
select ID, NAME
  from DOCTOR
 where SALARY > (select MAX (GEHALT)
                   from DOCTOR
                  where NAME = 'Haid');
```

Although (2) and (6) have a different structure the user will hardly notice the fundamental difference between query (1b) and (5). For a habitable system it is therefore necessary to provide solutions to both types of comparisons.

The semantic type associated with the value of the phrase to be compared enables DATENBANK-DIALOG to arrive at different interpretations for (1b) and (5). If the value has the correct dimension, it may safely be inserted as an argument into the comparison relation (in our case ">"). Otherwise, DATENBANK-DIALOG tries to construct a subquery giving a value by using the dominating relation (in our example `SALARY`) and

[3]SQL requires the subquery to return an unique value for comparison with >, hence the `MAX`-function.

fitting the comparison object into the "subject" slot of the attribute. In (6) this results in a subquery equivalent to the interpretation of (4a). This substructure is processed in a manner analogous to a top-level query.

As a consequence, anaphora resolution may be applied to it enabling DATENBANK-DIALOG to give the correct interpretation (7b)[4]for queries like (7a).

(7) *a)* *Wer$_i$ verdient mehr als sein$_i$ Vorgesetzter?*
(Who$_i$ earns more than his$_i$ superior?)

b)
```
select A1.ID, A1.NAME
  from DOCTOR A1
 where A1.SALARY > (select MAX(A2.SALARY)
                      from DOCTOR A2
                     where A1.SUPERIOR = A2.ID);
```

3.2 Domain Model vs. Database Model

Domain predicates like `SALARY` need not uniquely determine the relation and attributes of a corresponding predicate in the database. DATENBANK-DIALOG therefore splits the interpretation of an utterance into two stages: first, an interpretation in the domain model (i.e. a caseframe) is given. Secondly, this caseframe is interpreted to yield an interpretation in the database model (a DB-caseframe). The transformation step between the two structures is performed with the aid of a translation table. This approach enables DATENBANK-DIALOG to correctly produce quite different SQL queries from structurally similar questions. Compare (1b) with:

(8) *Welche Schwestern verdienen mehr als 20.000,–?*
(Which nurses earn more than 20.000,– ?)

On the domain level the resulting structures will still appear quite similar:

(9) *a)*
```
pred: SALARY
     a1: ?, (sem-restr DOCTOR)
     a2: >  20000
```
b)
```
pred: SALARY
     a1: ?, (sem-restr NURSE)
     a2: >  20000
```

Assume that our example database stores data about doctors and nurses in different tables leading to an ambiguity in the translation of the domain predicate `SALARY`. In that case the translation table will contain two entries for `SALARY` with different restrictions on argument `a1`. leading to SQL statements with different relations involved:

[4]The attribute `SUPERIOR` shall contain the ID of the superior of the doctor.

a)

Relation	Attributes	Comment
DOCTOR	ID	identification
	NAME	name
	NR_PATIENTS	number of patients

b)

PERSON	ID	identification
	NAME	name
	STATUS	doctor/nurse/patient
TREATMENT	ID_DOCTOR	id of doctor
	ID_PATIENT	id of patient

c)

DOCTOR	ID	id of doctor
	NAME	name of doctor
PATIENT	ID	id of patient
	NAME	name of patient
	ID_DOCTOR	patient's doctor

Table 1: Sample Database Models

(10) *a)*
```
select D.ID, D.NAME
  from DOCTOR D, DOCTOR_SALARY DS
 where D.ID = DS.ID and DS.SALARY > 20000;
```
b)
```
select N.ID, N.NAME
  from NURSE N, NURSE_SALARY NS
 where N.ID = NS.ID and NS.SALARY > 20000;
```

An argument of a domain predicate need not have a direct equivalent in the corresponding database table. This fact should be of no concern to the user. DATENBANK-DIALOG allows the translation table to contain an arbitrarily complex mapping of arguments—separately for each relation. Let us assume that the nurses' salary consists of a basic salary and a variable salary (contained in the attributes BASIC_SALARY and VAR_SALARY of NURSE_SALARY respectively). The interpretation for (8b) will then be

(11)
```
select N.ID, N.NAME
  from NURSE N, NURSE_SALARY NS
 where N.ID = NS.ID and (NS.BASIC_SALARY + NS.VAR_SALARY) > 20000;
```

This leads to another problem often encountered in realistic applications af natural language database interfaces. The user should not need to know about the actual encoding of information. Consider:

(12) Wieviele Patienten behandelt Dr. Haid?
(How many patients does Dr. Haid treat?)

As an answer the user expects the number of patients which were treated by Doctor Haid. In the domain level the query is represented unambiguously as:

(13)
```
pred: TREATMENT
  a1: x, (sem-restr DOCTOR)
         (modifier (pred: NAME
                     a1: x
                     a2: "Haid"))
  a2: (? ; COUNT)
```

The necessary information can be realized in quite different database models. Some possibilities are given in Table 1. Depending on the actual database model one gets of course quite different SQL queries:

(14) *a)*
```
select NR_PATIENTS
  from DOCTOR
 where NAME = 'Haid';
```
b)
```
select COUNT (UNIQUE T.ID_PATIENT)
  from PERSON P, TREATMENT T
 where P.NAME = 'Haid' and
       T.ID_DOCTOR = P.ID and
       P.STATUS = 'DR';
```
c)
```
select COUNT (UNIQUE C.ID)
  from DOCTOR D, PATIENT C
 where D.NAME = 'Haid' and
       C.ID_DOCTOR = D.ID;
```

There is a fundamental difference between (14a) and (14b,c). Whereas in the database model Tab. 1 a) the attribute `NR_PATIENTS` is contained in the database *explicitly* and can be treated analogously to `SALARY` above, the other two database models contain this "attribute" only *implicitly*. This means that the number of patients has to be computed (i.e. counted) by the SQL query. To obtain these quite different interpretations for (13), DATENBANK-DIALOG requires only a different mapping of the (contents of the) `a2`-slot of `TREATMENT` in the translation step between domain and database level.

A special case, where implicit attributes have to be made explicit in the database, occurs with queries such as

(15) Wer behandelt mehr Patienten als Dr. Haid?
Who is treating more patients than Dr. Haid?

Since comparison of two subqueries is impossible within a single SQL query, the query must be split up into two parts. A temporary table has to be created containing the relevant count-attribute together with information on the object bearing that attribute. Then the actual comparison can be made with the now explicit attribute. This results in[5]

[5] for the database model given in Tab. 1 c)

(16)
```
create table TEMP as
  (select COUNT(UNIQUE P.ID) "ANZAHL", D.ID "ID", D.NAME "NAME"
     from DOCTOR D, PATIENT P
    where P.ID_DOCTOR = D.ID
 group by D,ID, D.NAME) ;
select *
  from TEMP T
 where T.ANZAHL > (select MAX(COUNT(UNIQUE P.ID))
                     from DOCTOR D, PATIENT P
                    where D.NAME = 'Haid' and
                          P.ID_DOCTOR = D.ID);
```

3.3 Superlatives

Most of the problems encountered with comparatives also occur with superlatives. They are dealt with in an analogous way. An interesting phenomenon which has no direct parallel in comparative structures is shown in (17):

(17) *Welcher Arzt, der in der Unfallambulanz arbeitet, verdient am meisten?*
(Which doctor, who works in the casualty department, earns the most?)

There are at least two interpretations of utterance (17), i.e:

(18) *a) Who has the highest salary among the doctors working in the casualty department?*

b) Who has the highest salary of all persons and is, by the way, a doctor working in the casualty department?

Although interpretation (18b) can easily be derived by DATENBANK-DIALOG using the same formalism as for comparatives (copying the dominating relation) as in[6]

(19)
```
select P.ID, P.NAME
  from PERSON P
 where P.DEPT = 'CASUALTY' and
       P.STATUS = 'DR' and
       P.SALARY = (select MAX(P1.SALARY)
                     from PERSON P1);
```

in most circumstances (18a) is the most plausible interpretation and should be preferred. To produce this reading, another kind of copying has to be performed: not only must the dominating relation be copied but also the restrictions on the subject slot (i.e. on the bearer of the attribute) have to be inherited. This results in[7]

[6] assuming an extended version of database model Tab. 1 b)

[7] Note that restrictions on the selected `PERSON P` (in the first part of the query) must not be omitted, since persons from other departments or with other `STATUS` may accidentally have the same salary!

(20)
```
select P.ID, P.NAME
  from PERSON P
 where P.DEPT = 'CASUALTY' and
       P.STATUS = 'DR' and
       P.SALARY = (select MAX(P1.SALARY)
                     from PERSON P1
                    where P1.DEPT = 'CASUALTY' and
                          P1.STATUS = 'DR');
```

This copying in DATENBANK-DIALOG works on the caseframe representation, and thus is able to handle restrictions resulting from different syntactic constructions, such as:

(21) Welcher x verdient am meisten?
where restrictions on x may result from

a) the lexicon: *Unfallambulanzarzt*
"casualty department doctor"

b) adjective phrases: *in der Unfallambulanz arbeitende Arzt*
"in the casualty department working doctor"

c) prepositional phrases: *Arzt aus der Unfallambulanz*
"doctor from the casualty department"

d) noun phrases: *Arzt der Unfallambulanz*
"doctor of the *(gen)* casualty department"

e) relative clauses: *Arzt, der in der Unfallambulanz arbeitet*
"doctor, who works in the casualty department"

All these constructions end up as modifications in the caseframe due to the compositional nature of our approach. Thus a unified solution for inheritance of modifiers in their various forms is achieved.

3.4 Dimensions and Units

A correct comparison is only possible if the values compared are of the same dimension and share a unit of measure. Differences and incompatibilities may arise in different places:

- From special formatting conventions (e.g. 20000, 20.000, 20.000,–) possibly combined with a unit of measure (e.g. 0,3m, 30cm, 12,34in, $20)
- In the NL expression, when the user specifies a dimension and unit of measure verbatim (e.g. "10 Meter", "vor 3 Jahren")
- In the encoding of the database, where comparable columns (e.g. `DOCTOR.SALARY` and `NURSE.SALARY`) may have different associated units of measure.

DATENBANK-DIALOG solves this problem by defining a normalized form with a value associated with a unit and associated transformation rules between measures of different units. The transformation rules operate on different levels:

- at the *scanner* level: Patterns are defined that transform different formats of numbers to the corresponding numeric values and normalize abbreviations of units. At this level "compound" numbers like dates are also normalized.
- at the *parser* level: linguistic information is used to fill the slots in the normalized value frame.
- at the *interpretation* level: associated procedures are used to transform constant values from one unit to another.
- at the *database* level: transformation functions of the query language are used to perform conversions on variable data.

This enables DATENBANK-DIALOG to give the correct interpretation for (22) assuming a database encoding where `DOCTOR.SALARY` is stored in dollars and `NURSE.SALARY` in cents.

(22) Welche Krankenschwester verdient genau so viel wie Dr. Haid ?
Which nurse has the same salary as Dr. Haid?

4 Summary

Habitability is a crucial feature for natural language interfaces. It depends to a large extent on a smooth and principled interaction between syntax and semantics/pragmatics. In DATENBANK-DIALOG we have implemented a treatment of comparison and measures adhering to that strategy. As a result, our system

- gives a uniform interpretation to user queries of different syntactic and morphological appearance (equivalent, but syntactically different queries get the same semantic representation)
- relieves the users from having to know about the database representations of the concepts they use (domain concepts vs. database relations and attributes, implicit functions, unit conversion)
- makes ambiguities explicit and incorporates presuppositions (relation and restriction copying).
- enables users to enter data in the format and unit most convenient to them (formatting, unit conversion)

Acknowledgements

Financial support for the Austrian Research Institute for Artificial Intelligence is provided by the Austrian Federal Ministry for Science and Research. We would like to thank Professor Robert Trappl for his continuing support and Elizabeth Garner for proofreading and stylistic improvements.

References

Ballard, B.W. (1988) 'A General Computational Treatment of Comparatives for Natural Language Question Answering', in *Proceedings of 26th ACL*, Buffalo, NY, 41-47

Barwise, J. and R. Cooper (1981) 'Generalized Quantifiers and Natural Language', *Linguistics and Philosophy* **4**, 159-219

Heinz, W. and J. Matiasek (1989) 'Die Anwendung Generalisierter Quantoren in einem natürlichsprachigen Datenbank-Interface', in J. Retti and K. Leidlmair (eds.) *5. Österreichische Artificial Intelligence Tagung*, Springer, Berlin

Hendler, J.A. and P.R. Michaelis (1983) 'The Effects of Limited Grammar on Interactive Natural Language', *Proceedings of CHI'83: Human Factors in Comput. Systems*, 190-192, ACM, New York

Keenan, E.L. and J. Stavi (1986) 'A Semantic Characterization of Natural Language Determiners', *Linguistics and Philosophy* **9**, 253-326

Krause, J. (1982) *Mensch-Maschine-Interaktion in natürlicher Sprache*, Niemeyer, Tübingen

Pulman, S.G. (1991) 'Comparatives and Ellipsis', in *Proceedings of 5th European ACL*, Berlin, 2-7.

Rayner, M. and Banks, A. (1990) 'An Implementable Semantics for Comparative Constructions', *Computational Linguistics* 16(2)86-112

Tennant, H.R. (1980) 'Evaluation of Natural Language Processors', University of Illinois, Report T-103

Trost, H., E. Buchberger and W. Heinz (1988) 'On the Interaction of Syntax and Semantics in a Syntactically Guided Caseframe Parser', in *Proceedings of the 12th COLING*, Budapest, 677-682