REFERENT GRAMMAR (RG) IN COMPUTER COMPREHENSION, GENERATION AND TRANSLATION OF TEXT (SWETRA)

Bengt Sigurd

ABSTRACT AND INTRODUCTION

Theories of text and discourse have long assumed the existence of semantic objects called discourse referents, "things being talked about and referred to by pronouns and definite noun phrases"(cf. e.g. Karttunen (1976), Sigurd (1982), Sidner (1983)). Such referents are also often alluded to in sentence grammars, above all in theories of pronominalization and control - often in terms of antecedents - but the place of the referents in grammar is not clear, and referents are not to be found in textbook generative rules. The type of grammar to be described in this paper, named Referent Grammar (RG), presumes that referents are of crucial importance and places them directly into the syntactico-semantic representation of a sentence. The referents are introduced as variables in the noun phrase rules, carried into the sentence representation and identified on the basis of the syntactic patterns and markers of the language. The referents of sentence grammar can also be related to the referents of discourse, if rules for the text grammaticality of definite noun phrases and pronouns are included. We call that version of referent grammar Text Referent Grammar (TRG). The referents are assumed to have psychological reality.

Referent grammars have so far been written and implemented in Prolog for fragments of Swedish, English, French and Georgian (in decreasing order of detail; Georgian has been included in order to test referent grammar on a typologically very different language). The grammars have been used in analysing and generating sentences and text and generating answers to questions.

The grammars have also been used in a small-scale experiment translating between Swedish, English, French and Georgian (SWETRA).

The programs have been developed and run on a VAX 11/730 using a Prolog version, which the Department of Linguistics has obtained by courtesy of the Department of Cognitive Science, Sussex University (Sussex POPLOG Prolog, version 10,1985).

Referent grammar has been inspired by the potentials of the Definite Clause Grammar (DCG) formalism supported by many Prolog programs. It has also been influenced by ideas put forward in papers by Robin Cooper, e.g. Cooper(1985) and by comments from colleagues.

SOME REFERENT GRAMMAR RULES OF ENGLISH

A simple English declarative intransitive sentence is described by the following rule written in DCG.

```
sent(d,_,s(subj(X),pred(Y))) --> nps(X),vip(Y),[.],{agr(X,Y)}.
```

The predicate "sent" starts the built-in parser, "nps" is short for subject noun phrase, "vip" is short for intransitive verb phrase (not specified here) and the condition "agr" is short for agreement (not specified here). The first argument of the basic predicate "sent" specifies the mode of the sentence, typically ranging over d(eclarative), q(uestion), i(mperative). The second argument (_ in this case) will be a focused constituent in sentences where there is one - as in questions and topicalized sentences. Further "sent" arguments characterizing the sentence type can be included. It seems natural e.g. to have a further argument to distinguish aorist sentences in Georgian, which have a very different setup of cases. The representation s(subj(X),pred(Y)) is the expression which can be stored as a fact in the data base or used in the translation system. It can be shown as a tree in programs supporting this facility.

Subject and object noun phrases must be distinguished even in English (textbooks give an oversimplified view of the np problem), as pronouns have different forms in the two cases. Reflexive noun phrases must be distinguished in referent grammar, as they have a a very special referent meaning, namely that the referent of the subject is referred to in this constituent too. A referent is introduced into an indefinite noun phrase by a rule of the following type:

```
np(np(R1,nom(A,indef))) --> indart(X),n(Y),{fm(Y,A),gensym(rf,R1)}.
```

A phrase, such as "a girl" will be parsed, since "a" is an indefinite article (indart), specified later in the program (we disregard the difference between "a" and "an" here) and because "girl" is a noun (n). The predicate fm(Y,A) gives A as the translation (form-meaning relation) of Y. Translations of all words are to be given later in the program. The meaning of the noun is thus inserted directly into the representation, using a kind of "Semantic English" as a common semantic language for all languages - English is no doubt the most wide-spread language.The variable R1 will be given a successive number by the function "gensym", which will deliver rf1,rf2,rf3, etc. allowing us to count the number of referents introduced into the text. If we do not want to keep track of the number of referents, we may leave out "gensym". Prolog will then give successive numbers for the variables R1 in different noun phrases for each sentence. It is a characteristic feature of referent grammar that a referent is assumed to be one of the constituents of the np, namely the sister of the name of it, constituted by the predicate nom(inal) with its arguments, i.e. the meaning of the words in the noun phrase (derived by fm).

A sentence representation of "A boy ran" could then be: s(subj(np(rf17,nom(boy,sg,indef))),pred(v(run,past))), where the figure 17 indicates that this is the 17th referent introduced. In a transitive sentence, e.g. "A boy hit a girl", the rules would give a sentence representation where the referents of the boy and the girl would be represented by different numbers. A reflexive sentence, such as "The boy hit himself" would, however, be treated differently, and the subject and the object referents would get the same number.

The relative clause is typically marked by a relative pronoun or another relative marker in natural languages. The basic meaning of the relative marker is that a referent in the relative clause is identical to a referent in the matrix clause (the antecedent). Referent grammar can adopt this traditional idea, by inserting a copy of the "antecedent referent" into the relative clause by the np rules. One such rule is the following:

```
np(np(R1,nom(A,indef,rs(subj(np(R1,nom(A,indef)),pred(W))))))  -->
        indart(X),n(Y),rel(Z),vip(W)..,{fm(Y,A)}...
```

It states that the subject inside the relative clause (rs) is the same as the head of the np ("the antecedent"). This rule would handle cases such as "A boy,who ran,..". Similarly referents may be inserted into minor clauses, such as infinitival complements of the verb "promise". The identity of the referents in reflexive sentences, in complex noun phrases with relative clauses and verb phrases with complements can be secured directly by the phrase structure rules in referent grammar. It is also possible to scan the representations later in the derivation and identify referents according to the configurational patterns (indirect referent identification). Dahl(1986) includes some ideas of indirect referent identification.

REFERENT GRAMMAR IN TEXT COMPREHENSION

Definite and indefinite noun phrases are often treated by the same rule in generative grammars. There is, however, a basic difference between them, which appears clearly when one tries to relate the referents of sentence grammar to the referents of text. In a text grammar a definite NP should not be accepted (text grammatical) unless the referent can be identified. This can be done in the Prolog implementation by searching the data base for a previous mention of the referent. When trying to parse "the boy" the grammar looks for a fact in its data base (stored previously) where a "boy" is mentioned. If such a referent is found, the same referent number is assigned to the referent of "the boy". A referent grammar can thus assign the same referent number to the referent of "a boy", the referent of "the boy", the subject of the relative clause and the object represented by the reflexive pronoun "himself" in the following text:

"A boy(rf1) ran. The boy(rf1), who(rf1) ran, hit himself(rf1)."

Pronouns also occur with referents, which should be identified by looking in the text stored as semantic representations in the data base. Clearly a pronoun has to look for a referent identified by a noun with the proper gender features, as is well known.

A text referent grammar should be able to identify referents as indicated by numbers in the following short text:

"A boy(rf1) had a dog(rf2). He(rf1) liked a girl(rf3). He(rf1) gave it(rf2) her(rf3)."

The identification of referents with referents in the previous text or in the verbal and non-verbal context is a most complicated matter, however, which might, in fact, be out of reach for computers, as will be illustrated by the following little case study.

A REFERENT GRAMMATICAL ANALYSIS OF THE OPENING PARAGRAPH OF GUNNLAUG ORMSTUNGAS SAGA

An English translation of the first sentences of the famous Icelandic saga about Gunnlaug Ormstunga is given below. The numbers after noun phrases are the numbers given to the referents in the order they are introduced.

> Once upon a time there was a man(rf1), who(rf1) was called Torsten. His(rf1) father(rf2) was Egil(rf3), who(rf3) was son of Skalla-grim(rf4). His(rf4) father(rf5) was Night-Ulf(rf6), who(rf6) emigrated from Norway(rf7). His(rf1) mother(rf8) was called Asgerd, and was daughter of Bjorn(rf9). Torsten(rf1) lived at the farm Borg(rf10) in Borgarford(rf11). He(rf1) was a rich man and a great chief.

The phrase "once upon a time there was" is a well-known saga opener and the following phrase "a man, who was called X" is also common. The phrase "once upon a time there was" is a stylistic marker with equivalents in other languages, e.g. French "il-y-avait" Swedish "det var en gång", but it is empty from a semantic point of view. The sentence could have started by "there was a man", but the standard phrase has to be included in a grammar which is to understand sagas. The meaning of the first sentence is that a man called Torsten existed once and a referent grammar can give the following expression.

```
s(subj(r1,nom(man,indef,rs(subj(r1,nom(man,indef)),pred(v(name,past)),
        pf(Torsten)))),pred(v(exist,past)),tadv(once))
```

Note that the subject in the relative clause has the same referent as its head(rf1), given automatically by the grammar rules as discussed above. The analysis presumes that there are grammatical rules which identify "Once upon a time there was" as one chunk semantically represented as pred(v(exist,past)),tadv(once). The noun phrase rules

will identify "the man" and gensym will assign the number rf1 to this first referent. We note that "was" has the meaning "existed" here; there are other meanings of "be" as will be seen below and as is well-known. The phrase "was called" was rendered by "name,past" and we note that this predicate appears in different syntactic shapes in different languages, reflexive in French "qui s'appellait", active verb in Swedish "som hette", or passive construction "som kallades". The current referent grammar takes the name Torsten to be a predicative (predicate filler, abbreviated pf).

The second sentence talks about Torstens father, Egil and his father Skalla-grim. Egil Skalla-grimsson is a well-known Icelandic hero. The sentence is "His(rf1) father(rf2) was Egil(rf3), who(rf3) was son of Skalla-grim(rf4). The rules should give the following representation.

```
s(subj(np(rf2,nom(father,def,gen(rf1,nom(pro,ma)))),pred(v(ide,past)),
    pf(np(rf3,nom(egil,rs(subj(np(rf3,nom(egil))),pred(v(ide,past)),
    pf(n(son),prep(of)),np(rf4,nom(skalla-grim))))))
```

The representation of the expression "son of" is preliminary. The choice must be made considering how this meaning is expressed in the languages of the world. The important thing is the chain of referents in the representation. The representation above presumes certain rules of genitive noun phrases, where several referents are involved. It is assumed that the head of a genitive noun phrase is definite, as is suggested by the inacceptability of its occurence in a presentation construction such as "Once upon a time there was his father." The genitive noun phrase seems to introduce the head noun by indicating its relation to the determining (inflected) noun. The rule in the current grammar gives the head a new referent number. All the complexities of genitive noun phrases will not be outlined here. The cases in the text include a pronoun ("his"). We assume

that Torsten's father and Egil are taken to be different referents in this sentence; the purpose of the sentence is the identification of Torsten's father with Egil, who is a well-known person. While "his" in this sentence refers to Torsten, the next "his" refers to Skalla-grim. In pronunciation the last "his father" should rather be pronounced with stress on "his", but this cannot be rendered in writing. The referent of "his" can only be found by consulting the data base. We refrain from a detailed analysis of the next sentences which use very much the same vocabulary and syntax. The identification of referents poses many problems (cf. also Johnson & Klein,1986).

The referents discussed above are nominal, but there are others at least the size of a sentence. A typical case is given by such evaluative sentences as "That was fine", occurring after and clearly referring to a previous sentence, e.g. in the following sequence: "Sweden beat Britain. That was surprising." One may express this as follows in order to make the reference clear: "(The fact) that Sweden beat Britain was surprising" or "It was surprising that Sweden beat Britain." If each sentence is given a sentence number s1,s2, etc. we may keep track of these referents separately.

Referent number	Text
1 2 3 4 5 6 7 8 9 10	
+	Once upon a time there was a man(rf1),
+	who(rf1) was called Torsten.
+ + +	His(rf1) father(rf2) was Egil(rf3),
+ +	who(rf3) was the son of Skalla-grim(rf4).
+ + +	His(rf4) father(rf5) was Night-Ulf(rf6),
+ +	who(rf6) emigrated from from Norway(rf7).
+ +	His(rf1) mother(rf8) was called Asgerd,
+	and was the daughter of Bjorn(rf9).
+ +	Torsten(rf1) lived at the farm Borg(rf10).

Fig.1. Referent dynamic diagram of the first sentences of Gunnlaug Ormstungas saga. Each + marks a reference to that referent.

One way of presenting the referent dynamics in the opening paragraph is given in fig.1. It shows which referents are focused in each sentence and how the author returns to the hero Torsten after the presentation of his relatives. The text presents some problems of referent identification, which are, in fact, hard for both humans and computers. How do we know that "his" in "his mother" refers to Torsten and not to Night-Ulf, Skalla-grim or Egil? The reason seems to be the previous phrase "his father", which creates expectations of a parallel phrase "his mother". Note that there is another phrase "his father", but it seems probable that the first phrase "his father" gets its parallel phrase first. (A kind of push-down store strategy). The diagram also lists the typical predicates used in this opening.

COMMENTS ON THE SWEDISH MODULE OF THE PROGRAM

The figure (below) shows a fragment of a Prolog grammar which can analyse and generate some Swedish core sentences. Such a grammar has also been used in the translation between Swedish and English, French and Georgian. Similar grammars have been written for these languages. The first rules show the syntactic structures hinted at by the English translations with the rules. Note that Swedish has inverted word order whenever the subject does not occur first in the sentence. The grammar ("sent") has a special slot for mode, where the value d occurs when the sentence declarative, and q, when the sentence is a question. Declarative sentences and yes/no-questions are assumed to have identical representations, the only difference being in the mode variable. The next argument of "sent" is a slot for a focused constituent. An initial adverb is assumed to be a focused constituent in Swedish and so is the finite verb in yes/no questions and the question word in wh-questions. Swedish may also focus the verb in declarative sentences using an auxiliary verb corresponding to English "do". This construction is illustrated by the syntactic rule including [gjorde].

Some later rules are also shown indicating how definite and indefinite nps with relative clauses are handled. Swedish has very complex agreement within the noun phrase. All nouns are divided into neuter and non-neuter nouns (which is a well-known stumbling-block for foreigners). Furthermore, the number and the definiteness of the noun phrase has to be distinguished. The agreement rules suggested look at one pair of words at a time, e.g. first the article and the noun, then the adjective and the noun in order to make sure that e.g. "det lilla barnet" (the little child) and "en liten flicka" (a little girl) come out right.

```
1  71 80  (EDITING: subtrans.pl)
SWEDISH */
sent(d,_,s(subj(X),pred(Y))) --> nps(X),vip(Y),[.].   /* x ran */
sent(d,Z,s(subj(X),pred(Y),advl(Z))) --> adv(Z),vip(Y),nps(X),[.].  /* fast ra|
sent(d,Z,s(subj(X),pred(Y),advl(Z),advl(W))) --> adv(Z),vip(Y),nps(X),adv(W),[|
sent(d,_,s(subj(X),pred(Y),advl(Z))) --> nps(X),vip(Y),adv(Z),[.].  /* ran fas|
sent(d,_,s(subj(X),pred(Y),advl(Z),advl(W))) --> nps(X),vip(Y),adv(Z),adv(W),[|
sent(q,Y,s(subj(X),pred(Y))) --> vip(Y),nps(X),[?]. /* ran x? */
sent(q,Y,s(subj(X),pred(Y),advl(Z))) --> vip(Y),nps(X),adv(Z),[?].
sent(q,X,s(subj(X),pred(Y))) --> npq(X),vip(Y),[?]. /* who ran */
sent(q,X,s(subj(X),pred(Y),advl(Z))) --> npq(X),vip(Y),adv(Z),[?].
sent(q,X,s(subj(X),pred(Y),advl(Z),advl(W))) --> npq(X),vip(Y),adv(Z),adv(W),[|
sent(d,Y,s(subj(X),pred(Y))) --> vip(Y),[gjorde],nps(X),[.]. /* ran did x */
sent(d,_,s(subj(X),pred(Y),aobj(Z))) --> nps(X),vtp(Y),npo(Z),[.]. /* x hit y |
sent(d,_,s(subj(X),pred(Y),aobj(Z),advl(W))) --> nps(X),vtp(Y),npo(Z),adv(W),[|
sent(d,_,s(subj(X),pred(Y),aobj(X))) --> nps(X),vtp(Y),npr(Z),[.]. /* x hit hi|
sent(q,Y,s(subj(X),pred(Y),aobj(Z))) --> vtp(Y),nps(X),npo(Z),[?]. /* hit x y |
sent(q,Y,s(subj(X),pred(Y),aobj(Z),advl(W))) --> vtp(Y),nps(X),npo(Z),adv(W),[|
sent(q,X,s(subj(X),pred(Y),aobj(Z))) --> npq(X),vtp(Y),npo(Z),[?]. /* who hit |
sent(d,Z,s(subj(X),pred(Y),aobj(Z))) --> npq(Z),vtp(Y),nps(X),[?]. /* whom hit|
sent(d,_,s(subj(X),pred(Y),dobj(Z),aobj(W))) --> nps(X),vttp(Y),npo(Z),npo(W),|
sent(q,_,s(subj(X),pred(Y),dobj(X),aobj(W))) --> vttp(Y),nps(X),npo(Z),npo(W),|
sent(d,_,s(subj(X),pred(Z,Y))) --> nps(X),cop(Y),pf(Z),[.],{pagr(X,Z)}. /* x i|
sent(q,Z,s(subj(X),pred(Z,Y))) --> cop(Y),nps(X),pf(Z),[?],{pagr(X,Z)}. /* is |
```

Some syntactic rules used in an implementation of referent grammar (SWETRA)

REFERENT GRAMMAR IN AUTOMATIC TRANSLATION (SWETRA)

Referent grammar is used in the Swedish translation system SWETRA. Representations with "Semantic English" are used as an intermediate level. They have to be modified in some ways when going into another language and some interlanguage rules have been worked out. A number of basic sentences can be translated. Case marking and agreement are seen as "local" problems in each language. Such markings disappear when sentences are translated into the normalized Semantic English representations making translation possible. The following are some examples of sentences which can be translated. The predicate setrans means translate from Swedish into English, sftrans means translate from Swedish into French, etc. There is also a predicate which translates from Swedish into English, into French and back to Swedish. The vocabulary is very restricted as yet and the system is only a prototype intended to indicate the potentials of referent grammar in automatic translation. (The figure shows further examples)

setrans(en liten flicka,	a little girl,
som beundrade en pojke,	who admired a boy,
som en hund bet, sprang.)	whom a dog bit, ran.
sftrans(den lilla flickan sprang.)	la petite fille courut.
sgtrans(en hund bet flickan.)	jagli gogos kbenda.
sgtrans(den flickan gav ett barn en hund.)	gogo svilfs jagli darboda.

Even sentences with pronouns in different order in the languages can be handled correctly, as witnessed by the following case.

English: He gave it her. (Grammatical in some variants of English)

French: Il le lui donnait.

Swedish: Han gav henne den.

```
?- sgtrans([en,flicka,sprang,.],X).
X = [gogo, darboda, .] ?
yes

?- sgtrans([en,hund,bet,flickan,.],X).
X = [jagli, gogos, kbenda, .] ?
yes

?- setrans([den,gick,snabbt,.],X).

X = [it, went, fast, .] ?
?- setrans([ubaaten,styrde,mot,karlskrona,.],X).

X = [the, submarine, headed, towards, the, carlscrona, .] ?
yes

?-
?- setrans([ubaaten,foeljde,en,fiskebaat,.],X).
** (1) Call : intse(s(subj(np(_1, nom(m(submarine, sg), m(def)))), pred(v(m(fo
        llow, past))), aobj(np(_2, nom(m(fishingboat, sg), m(indef))))),
         _3)?
** (1) Exit : intse(s(subj(np(_1, nom(m(submarine, sg), m(def)))), pred(v(m(fo
        llow, past))), aobj(np(_2, nom(m(fishingboat, sg), m(indef))))),
         es(subj(np(_1, nom(m(submarine, sg), m(def)))), pred(v(m(follow,
         past))), aobj(np(_2, nom(m(fishingboat, sg), m(indef))))))?

X = [the, submarine, followed, a, fishing_boat, .] ?
         _3)?

?- setrans([en,jagare,gick,fraan,karlskrona,.],X).
** (1) Call : intse(s(subj(np(_1, nom(m(destroyer, sg), m(indef)))), pred(v(m(
        go, past))), advl(adv(_2, nom(m(from), np(_3, nom(m(carlscrona),
         m(def)))))), _4)?
** (1) Exit : intse(s(subj(np(_1, nom(m(destroyer, sg), m(indef)))), pred(v(m(
        go, past))), advl(adv(_2, nom(m(from), np(_3, nom(m(carlscrona),
         m(def)))))), es(subj(np(_1, nom(m(destroyer, sg), m(indef)))),
         pred(v(m(go, past))), advl(adv(_2, nom(m(from), np(_3, nom(m(carlscro
        na), m(def)))))))?

X = [a, destroyer, went, from, the, carlscrona, .] ?
yes
?-
?- setrans([ubaaten,doek,naer,jagaren,anlaende,.],X).
** (1) Call : intse(s(subj(np(_1, nom(m(submarine, sg), m(def)))), pred(v(m(di
        ve, past))), advl(adv(m(when), _2, rs(subj(np(_3, nom(m(destroyer,
         sg), m(def)))), pred(v(m(arrive, past))), advl(adv(_2)))))), _4)?
** (1) Exit : intse(s(subj(np(_1, nom(m(submarine, sg), m(def)))), pred(v(m(di
        ve, past))), advl(adv(m(when), _2, rs(subj(np(_3, nom(m(destroyer,
         sg), m(def)))), pred(v(m(arrive, past))), advl(adv(_2)))))), es(subj(
        np(_1, nom(m(submarine, sg), m(def)))), pred(v(m(dive, past))),
         advl(adv(m(when), _2, rs(subj(np(_3, nom(m(destroyer, sg), m(def)))),
         pred(v(m(arrive, past))), advl(adv(_2)))))))?

X = [the, submarine, dived, when, the, destroyer, arrived, .] ?
yes
```

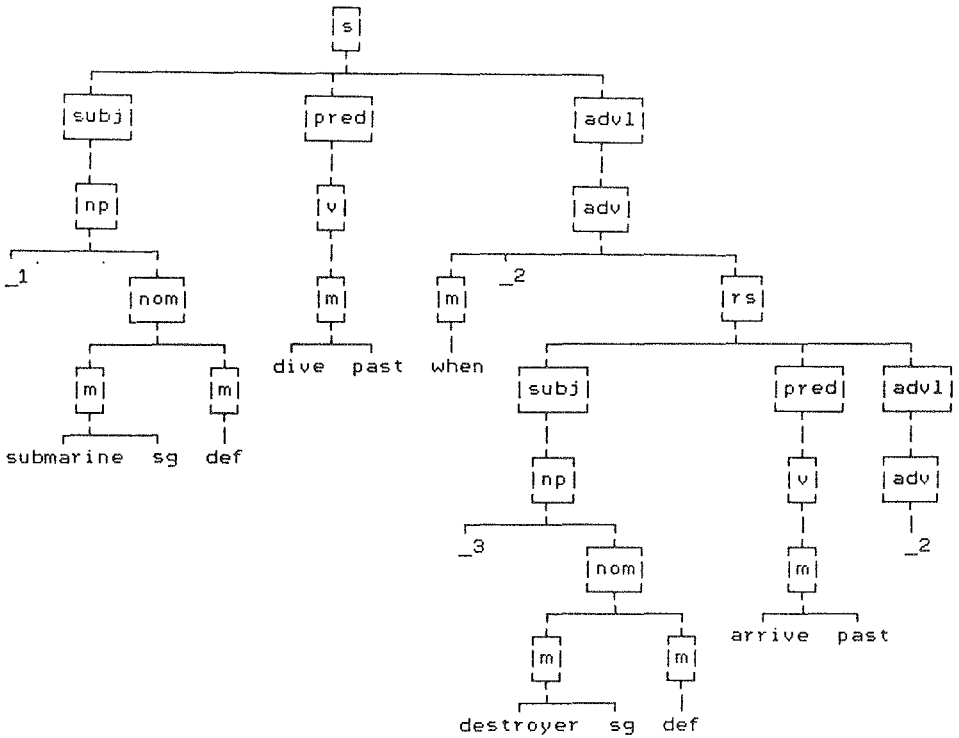
Sample translation by SWETRA. Note the intermediate representations with the referent variables.

The choice of the pronouns, which depends on the features (number, gender, as e.g. French le/la,les) poses a knotty problem, however. A dog (chien) would require a following "le", while a flower (fleur) would require "la", both corresponding only to English "it". The dog (hund) requires "den" in Swedish, a child (barn) would require "det" as a pronoun. One way to handle this situation is to use text referent grammar, keeping track of the referents in the text and the information about them. If no such solution can be found the goals of automatic translation must be more modest.

Experiments in automatic translation using referent grammar are carried out under the project title SWETRA (Swedish Computer Translation Reseach), supported by The Swedish Research Council for the Humanities and the Social Sciences. The experiments follow different lines. One line will emphasize the multilanguage translation potentials developing modules for a number of languages. This will allow translation of certain types of information e.g. weather forecasts between that set of languages. Such a system translating some 20 000 words between the EG languages is in fact the main goal of EUROTRA.

Another line followed by SWETRA is pairwise translation, e.g. between English and Georgian, between Japanese and Swedish in order to examine the problems of translation between typologically dissimilar languages. Special interest has been taken in automatic translation between Georgian and English as the tense-aspect system of Georgian requires special arrangements. An illustrative extract from a Georgian module is shown.

Translation between English and Swedish in a certain field - marine events - has been studied in some detail and a print-out from a session translating text dealing with a submarine and a destroyer outside Carlscrona is shown.



TREE DIAGRAM OF THE SYNTACTIC-SEMANTIC REFERENT REPRESENTATION OF "UBÅTEN DÖK NÄR JAGAREN ANLÄNDE", WHICH IS ALSO USED AS INTERMEDIATE REPRESENTATION IN TRANSLATION

HIGH RESOLUTION REFERENT GRAMMAR (HRG) IN ANALYSIS AND GENERATION OF TEXT

High resolution referent grammar (HRG) is a kind of grammar where the content of a sentence is dissolved into smaller pieces, each including referents which show the connections of the pieces. It is an extension of Referent Grammar (RG) presented above. Using a simple example, the sentence "Bill hit Tom" is dissolved into: somebody did something, what he did was to hit, he who hit was Bill, he who was hit was Tom. Using a formalism suitable for Prolog implementation the meaning of the sentence is represented as:

```
s(n(P,N1,N2),d(hit,P),d(bill,N1),d(tom,N2)). (Bill hit Tom)
```

Expressed in words the formula says: It is a fact that there is a predicate (P) with two arguments (N1 and N2) combined into a nexus (n);cf. the use of this term by Jespersen. The predicate (P) is denoted (d) by "hit", the first argument (N1) is denoted by Bill and the second argument by Tom. These four facts are gathered inside a sentential fact (s). The nexus fact is the first. It states that there is a finite predicate, which is a 2-place predicate. The next facts specify the predicate and the two nouns. The letters P and N are used as variable names.

It is convenient to talk about these facts (nexus, denotation) as "factoms", making up "factecules", using the parallelism of atoms and molecules. The terms of the factoms can then be termed "refons", as they refer and are the structural equivalents of electrons and protons.

The factoms can be combined in different ways corresponding to different sentence types. The following table shows the set-up of factoms for some basic types of sentences.

No of factoms, set-up	Sample sentences (Eng,Swe)
2 n(P,N),d(A,P)	There was running, Det sprangs
3 n(P,N),d(A,P),d(B,N)	Tom ran,Tom sprang (intrans)
3 n(P,N1,N2),d(A,P),d(B,N2)	Tom was hit,Tom slogs (ag-less)
4 n(P,N1,N2),d(A,P),d(B,N1),d(C,N2)	Bill hit Tom,Bill slog Tom (tr)
4 n(P,N1,N2),d(A,P),d(C,N2),d(B,N1)	Tom was hit by Bill (pass)
4 n(P,N1,N2,N3),d(A,P),d(B,N2),d(C,N3)	Bill was given a book (ag-less)

The basic order in the sentence factecules above is: the nexus(predication, finite) combination first, the character of the predicate next and the characters of the arguments in order next. Such formulas offer the order of the factoms in the factecules as a way of differentiating between active and passive, as is shown above.

Note that what characterizes an impersonal construction is the lack of information about the N, which seems intuitively correct. Similarly the agentless passive lacks information about the character of the agent, i.e. the first N (N1) in the nexus factom.

EXPLAINING COMPREHENSION BY HRG

Comprehension is seen as a parsing process resulting in a factecule which is stored in the memory (data base). The individual factoms are stored in a second step. It is an open question whether further implications from these first facts, e.g. factoms equivalent to relative nps should be stored during the comprehension process or derived when needed.

This means that when the sentence "A boy hit a dog" or Swedish "En pojke slog en hund" is to be understood, a factecule such as s(n(p1,n1,n2),d(hit,p1),d(boy,n1),d(dog,n2)) is stored (asserted) along with the individual factoms: n(p1,n1,n2),d(hit,p1) etc. New constellations, e.g. s(n(p1,n1,n2),d(hit,p1)) equivalent to "There was hitting", and s(n(p1,n1,n2),d(hit,p1),d(dog,n2)) equivalent to "A dog was hit" are also true (and available as potential sentences). The existence of the factom d(boy,n1) in the data base makes it possible to identify a later mention of the boy (the boy,he,the scoundrel) with n1. The factom equivalent to the np "the boy who was hit": d(s(n(p1,n1,n2),d(hit,p1),d(dog,n2)),n2) may also be made available.

The representation uses lower case letters with numbers instead of upper case letters in accordance with Prolog conventions. The values p1,p2 etc and n1,n2 etc are assigned during the parsing procedure and stored with the factoms. This means that it is possible to follow the build-up of the database on the basis of the sequence of sentences. It is also possible to know which is the preceding verb, noun etc., which is of interest in interpreting anaphora and proforms.
(See printout of sample session)

PRO-FORMS IN HRG

Referent grammar is specifically designed to handle referential expressions. There are a number of referential expressions in natural languages. We will only discuss some here. The "It" in "It is true that Bill ran" refers to the following clause "that Bill ran", but in a sentence such as "It/That is true" the "it" refers to a preceding sentence. In the little text "Bill ran. So did Tom" the "so" refers to the action (running) mentioned just before, the denotation of the previous predicate. This information has thus to be extracted in order for the latter sentence to be understood and the factecule and factoms corresponding to "Tom ran" stored. Natural languages utilize reference extensively and in an economic way.

The most frequent cases are definite noun phrases and pronouns, whose interpretation has been discussed extensively for a long period and cause computerized text comprehension systems great problems. (See References)

High resolution grammar stores individual factoms as the comprehension proceeds and gives each referent an individual number. A referent established previously may be referred to later, thus being used as a discourse referent. After the sentence "A boy ran" is processed the factoms n(p1,n1),d(ran,p1),d(boy,n1) are stored in the data base (simulating human memory). If the next sentence is "The boy hit a dog", the referent of "the boy" is identifed with the same referent number (n1) found to be used with a boy in a factom. The dog will be given a new number (n2), however, and stored in a factom as d(dog,n2). A following noun phrase "the dog" can then be taken to refer to this n2. More technically, the factom d(dog,X) is looked for and n2 is found to be a possible value of X.

Similarly, a pronoun is taken to refer to a current referent mentioned in a factom, if the requirements of number, gender etc are met. This is handled by implicational rules in the Prolog implementation. If there is a factom d(boy,n9), then d(he,n9) is available. Pronouns are often ambiguous, and we have not as yet a solution to this problem, which also pesters human communication. It is not reasonable to expect that computers should be able to solve problems of ambiguous reference, which human beings hardly can solve, in spite of all their experience and information processing capacity.

It is also a well-known fact that discourse referents are referred to by hyponymic expressions. A hyponymic expression such as "the animal" can thus be used about a dog ("Bill hit the dog. Tom hit the animal too"). This problem can be solved by including hyponymic implicational rules. If there is a factom d(dog,n7) then d(animal,n9) is available, assuming that animal is given as a hyponym of dog.

Verbal hyponyms are handled in a similar way. If d(ran,p9) is true then d(moved,p9) is available, as moved is a hyponym of ran. It is also known that various pejorative and affectionate expressions may serve as referent expressions ("Bill hit the dog. Tom hit the beast too"). This could also be handled by implicational rules, but it seems dangerous to state generally that if somebody is called a boy he can also be called a scoundrel. As has been observed by Merle Horne(1986) coreferential terms such as "the animal", "the beast" and pro-forms of nouns and verbs do not take focal stress (are destressed).

One of the rationales for High Resolution Referent Grammar is to offer units which are suitable for handling reference expressions of natural languages. The factecules corresponding to sentences are clearly needed for the referring "it,this" in cases such as "Philosophy is important. It/This is true." But there seems to be need for a still bigger unit in cases where the whole content of the preceding text is referred to. Such a discourse referent may be called "circumscriptive" (a term used by Bonnie Webber in a lecture) or "summative" as it sums up what has been said. A typical case is when a person says e.g. "This is all I have to say."

One may carry the resolution suggested above further in order to satisfy further needs of referential processes. One may consider whether tense should not be factored out from the main predicate and "Bill ran" be represented by n(t1,p1,n1),d(run,p1),d(past,t1), d(bill,n1), where we have a tense variable (t) in the first factom. One reason for doing this could be that one may (possibly) say: "Bill ran. So does Tom". In that case only the component "run", not the tense, is to be copied into the factecule of the second sentence.

The sentence type "So did Bill" (Swedish "Det gjorde Bill ocksaa") seems to be a transitive sentence in both languages, where "so/det" is the preposed (topicalized) object. Swedish uses "gjorde" (goera) as an equivalent to English "did" (do) in this case, but not generally. The "so/det" may refer to the denotation of the verb (as shown) but also to the verb including an object (the whole VP) as in: "Tom hit a dog. So did Bill", where the last sentence means that Bill hit a dog, not only that he hit. This suggests that factoms corresponding to the whole predicate phrase should be copied.

GENERIC SENTENCES

Generic sentences such as: "(All) boys are animals","A boy is (always) an animal" (Swedish "(All) pojkar aer djur", "En pojke aer (alltid) ett djur", and even "Pojken aer ett djur" (The boy is an animal) require special interpretation, as is well-known. There is an extensive linguistic and philosophical literature on the problem. The standard philosophical interpretation is: "If something is a boy, then it is an animal", but this seems rather to be a secondary implication. We suggest that generic sentences be rendered by special factoms (g), and that a later implicational rule allow the program to infer facts as suggested by the standard philosophical interpretation. The implication is straightforward given our analysis in factoms (and in Prolog). If there is a factom d(bird,n9), and a generic factom g(animal,bird), then d(animal,n9) is true, or more generally (in Prolog) d(X,N) :- d(M,N),g(X,M). There is, of course, much more to be said about this problem.

COMMENTS ON AN EXPERIMENTAL PROLOG IMPLEMENTATION OF HRG

The following program only includes a few basic types of sentences and a few vocabulary items. It is to be noted that the built-in predicate "gensym" is called to give successive referent numbers to verbs and nouns. This is handled on the sentence level for verbs which get p1,p2,p3 etc successively and in the np submodule for nouns. The numbers generated in the lower np module have to be carried into the nexus factom by the rules of the "sent" predicate.

The basic predicate "sent" processes some Swedish sentences, but equivalent English sample sentences are given as comments. This program understands sentences such as: [en_pojke,sprang] (A boy ran), [en_pojke,slog,en_hund] (A boy hit a dog) and can store the corresponding factecules and factoms if asked to by using "lagra (X)" (store). The first line in the program reads:" if something (X) is a noun phrase (np) and the following item (Y) a verb, then build the structure (tree) indicated to the left (as an argument with s as the root)". In that structure the values of the variables are given by gensym, except for referring expressions, where the value is taken from a factom stored previously.

The meaning is dissolved into factoms as mentioned above. The word recognition procedures offer the semantic representations of the words which are given in "Semantic English", a representation which is also used in the intermediate representations of the machine translation system (SWETRA) mentioned above. Only simple noun and verb phrases are covered and there are only a few lexical items in this demo program.

Some basic sentence types are included in the program. The predicate with the prefix g (gsent) is used in generation, when gensym is not to be used. There are interactive commands in the program, which generate all possible sentences within the grammar (say) and all possible sentences which are true (truesay).

The program includes intransitive, transitive (active and passive, with and without agent), impersonal, generic and coordinated sentences. Some sentences with verb pro (so,det) are also covered. The factecule which is the first argument of sent shows how the different sentences are analysed.

If we write sent(T,[en,pojke,sprang],[]), we get T= s(n(p1,n1), d(ran,p1),d(boy,n1)), which may be spelled out as "there is an event involving P and N, where P is "ran" and N is "boy". The factoms may be combined in several ways to make up different factecules, as mentioned. If asked to generate true sentences by the command truesay the program will search the data base for primary and implicated factecules and combine primary and implicated factoms into all possible factecules. It will e.g. generate "Det sprangs" (there was running) as the factecule s(n(p1,n1),d(ran,p1)) can be constructed.

The factoms from a sentence such as ([en_pojke,slog,en_hund] ("A boy hit a dog") may be combined in various ways as indicated by the syntactic rules. Beside the original sentence it may evoke:[en_hund, slog,s,av,en_pojke] ("a dog was hit by a boy"), [en_hund,slog,s] ("a dog was hit"), [det,slog,s] ("there was hitting"). Coordinated sentences, e.g. [en_pojke,sprang,och,hoppade] ("A boy ran and jumped") are also available, if the same boy is known to have jumped. If asked to generate all possible sentences on the basis of the factoms stored one may get unexpected coordinated sentences, such as [en_pojke,sprang,och, sprang] ("A boy ran and ran"), on the basis of the previous sentence [en_pojke,sprang]. The reduplicated verb, however, carries a special meaning! Definite nouns, pronouns, and some hyponyms and pejoratives can be used if the conditions are met (as discussed above).

There are various interactive predicates. Storing is handled by the procedure "lagra(X)". The whole factecule is later split into factoms, which are also stored (asserted). The predicate "truesay"gives all sentences which can be rendered by combinations of the factoms learnt or derived by the system. The predicate "answer(X)" gives answers (or rather confirmations) to declarative sentences, which have been given or which can be derived as true by the implicational rules. Note that the predicate "gensym(a,X)" has to be loaded by "library(gensym)". The predicate "library(useful)" should also be loaded in order to make "append" and some other commands available. Debugging and extension of the program is still going on, but the program can analyze some short sample texts nicely (as illustrated).

```
yes

?- library(gensym).
yes

?- load(nex).
yes

?- lagra([en_pojke,slog,en_hund]).
n(p4, n7, n8)d(hit, p4)d(m(boy, sg), n7)d(m(dog, sg), n8)yes

?- lagra([den,sprang]).
n(p7, n8)d(ran, p7)d(rpro, n8)yes

?- lagra([det,gjorde,pojken,ocksaa]).
n(p10, n7)d(ran, p10)d(m(boy, sg), n7)yes

?- lagra([djuret,lekte]).
n(p13, n8)d(played, p13)d(m(animal, sg), n8)yes

?- lagra([det,gjorde,han,ocksaa]).
n(p16, n7)d(played, p16)d(mpro, n7)yes

?-
```

Analysis and comprehension of a short Swedish text corresponding to English: A boy hit a dog. It ran. So did the boy. The animal played. So did he.

Note that the referents n7 and n8 introduced in the first sentence are identified thoughout the text.

PROGRAM
Initialize by lagra([en_pojke,sprang]). This version generates np at module level and inserts the gensym generated noun variable (N) at the sentence level. The verb variable (P) is generated
at sentence level. Relativization is handled by deriving d-facts from s-facts.
Various interactive commands available:lagra,answer,say,truesay,facts.

```
sent(s(n(P1,N1),d(Y,P1),d(Z,N1))) --> np(X),v(Y),{gensym(p,P1),
   asserta(lastv(P1)),d(Z,N1)=X}.
/* current(N1)? */
gsent(s(n(P1,N1),d(Y,P1),d(X,N1))) --> gnp(X),v(Y). /* for generation */

/* following takes verb value from previous sentence */
sent(s(n(Z,N1),d(Y,Z),d(W,N1))) --> [det],[gjorde],np(X),[ocksaa],
   {d(W,N1)=X,lastv(Z),d(Y,Z)}.
gsent(s(n(P1,N1),d(Y,P1),d(W,N1))) --> [det],[gjorde],gnp(X),[ocksaa],
   {lastv(Z),d(Y,Z),d(W,N1)=X}.

/* sentence with unstressed hyponymic coreferent with  verb, np additive
    ocksaa  */
sent(s(n(P1,N1),d(Y,P1),d(X,N1))) --> np(X),v(Y),[ocksaa],{d(X,N1),
    hyp(Z,Y),d(Z,P2),gensym(p,P1)}.
/* sentence with stressed (nonhypo) verb, verb additive ocksaa */
sent(s(n(P1,N1),d(Y,P1),d(X,N1))) --> np(X),v(Y),[ocksaa],{gensym(p,P1),
    asserta(lastv(P1))}.
/* following coordinates predicates */
sent(s(n(P1,N1),d(Y,P1),n(P2,N1),d(Z,P2),d(W,N1))) --> np(X),v(Y),
   [och],v(Z),{d(W,N1)=X,gensym(p,P1),gensym(p,P2)}.
gsent(s(n(P1,N1),d(Y,P1),n(P2,N1),d(Z,P2),d(W,N1))) --> gnp(X),v(Y),
   [och],v(Z),{d(W,N1)=X}.

/* following establishes 2 factoms  */
sent(s(n(P1,N1),d(Y,P1))) --> [det],v(Y),[s],{gensym(p,P1),asserta
   (lastv(P1))}. /* there is dancing */
gsent(s(n(P1,N1),d(Y,P1))) --> [det],v(Y),[s].
/* following is a predicative non-generic (plural) sentence */
sent(s(n(P1,N1),d(Y,P1),d(W,N1))) --> ni(X),[aer],pf(Y),{gensym(p,P1),
   d(W,N1)=X}.
gsent(s(n(P1,N1),d(Y,P1),d(W,N1))) --> gni(X),[aer],pf(Y),{asserta
   (lastv(P1))}.
/* following are some tentative generic sentences */
sent(s(g(Y,X))) --> [alla],ni(X),v(Y).
gsent(s(g(Y,X))) --> [alla],ni(X),v(Y).
sent(s(g(Y,X))) --> [alla],ni(X),[aer],pf(Y). /* all boys are nice */
gsent(s(g(Y,X))) --> [alla],ni(X),[aer],pf(Y).
sent(s(ng(Y,X))) --> [inga],ni(X),[aer],pf(Y). /* no boys are nice */
/* following establishes 4 factoms */
sent(s(n(P1,N1,N2),d(Y,P1),d(W,N1),d(U,N2))) --> np(X),vt(Y),np(Z),
   {gensym(p,P1),asserta(lastv(P1)),d(W,N1)=X,d(U,N2)=Z}.
gsent(s(n(P1,N1,N2),d(Y,P1),d(W,N1),d(U,N2))) --> gnp(X),vt(Y),gnp(Z),
   {d(W,N1)=X,d(U,N2)=Z}.
sent(s(n(P1,N1,N2),d(Y,P1),d(W,N1),d(U,N2))) --> np(Z),vt(Y),[s_av],
   np(X),{gensym(p,P1),asserta(lastv(P1)),d(W,N1)=X,d(U,N2)=Z}.
```

```
gsent(s(n(P1,N1,N2),d(Y,P1),d(W,N1),d(U,N2))) -->gnp(Z),vt(Y),[s_av],
   gnp(X),{d(W,N1)=X,d(U,N2)=Z}.
sent(s(n(P1,N1,N2),d(Y,P1),d(U,N2))) --> np(Z),vt(Y),[s],{gensym(p,P1),
   d(U,N2)=Z}. /* pass without ag */
gsent(s(n(P1,N1,N2),d(Y,P1),d(U,N2))) --> gnp(Z),vt(Y),[s],{d(U,N2)=Z}.
sent(s(n(P1,N1,N2),d(Y,P1))) --> [det],vt(Y),[s],{gensym(p,P1)}. /*
   impersonal trans */
gsent(s(n(P1,N1,N2),d(Y,P1))) --> [det],vt(Y),[s].
np(d(X,N1)) --> [A],{isn(A),bet(A,X),gensym(n,N1)}. /* indefinite
   (new N1) */
gnp(d(X,N1)) --> [A],{isn(A),bet(A,X)}.
np(d(X,N1)) --> [A],{isnd(A),bet(A,X),d(X,N1)}. /* definite np (old N1) *
gnp(d(X,N1)) --> [A],{isnd(A),bet(A,X),d(X,N1)}.
/* relative nps */
np(d(L,N1)) --> np(X),[som],v(Y),{d(W,N1)=X,L=s(n(P1,N1),d(Y,P1),
   d(W,N1)),L}.
np(d(L,N1)) --> np(X),[som],vt(Y),np(Z),{d(W,N1)=X,L=s(n(P1,N1,N2),
   d(Y,P1),d(W,N1),Z),L}.
np(d(L,N2)) --> np(Z),[som],np(X),vt(Y),{d(W,N2)=Z,L=s(n(P1,N1,N2),
   d(Y,P1),X,d(W,N2)),L}.
/*Lexicon */
isn(ett_barn).
bet(ett_barn,m(child,sg)).
isn(ett_djur).
bet(ett_djur,m(animal,sg)).
ne(m(animal,sg)).
ne(m(child,sg)).
isn(en_hund).
bet(en_hund,m(dog,sg)).
re(m(dog,sg)).
isn(en_idiot).
bet(en_idiot,m(idiot,sg)).
ma(m(idiot,sg)).
fe(m(idiot,sg)).
isn(en_pojke).
bet(en_pojke,m(boy,sg)).
ma(m(boy,sg)).
isn(en_flicka).
bet(en_flicka,m(girl,sg)).
fe(m(girl,sg)).
/* definite. Gender based on bedeutung (in indefinite) */
isnd(pojken).
bet(pojken,m(boy,sg)).
isnd(flickan).
bet(flickan,m(girl,sg)).
isnd(barnet).
bet(barnet,m(child,sg)).
isnd(idioten).
bet(idioten,m(idiot,sg)).
isnd(den_idioten). /* used in pejorative sense */
bet(den_idioten,m(idiot,sg)).
isnd(den_boven).
bet(den_boven,m(scoundrel,sg)).
isnd(djuret).
bet(djuret,m(animal,sg)).
isnd(hunden).
bet(hunden,m(dog,sg)).
isnd(den).
bet(den,rpro).
isnd(han).
bet(han,mpro).
```

```
isnd(hon).
bet(hon,fpro).
isnd(det).
bet(det,npro).
/* pronoun identification.If eg n9 is denoted by boy it can be denoted
   by han  */
d(rpro,Y) :- re(X),d(X,Y). /* rpro is substitutable if noun is re (den) *,
d(mpro,Y) :- ma(X),d(X,Y). /* mpro is substitutable if noun is ma (han) *,
d(fpro,Y) :- fe(X),d(X,Y). /* fpro is substitutable if noun is fe (hon) *,
d(npro,Y) :- ne(X),d(X,Y). /* npro is substitutable in noun is ne (det) *,

/* pejorative and hypochoristic (affectionate) substitution rules  */
d(Y,Z) :- pej(X,Y),d(X,Z). /* if n9 is a boy, n9 is also a scoundral */
pej(m(boy,sg),m(idiot,sg)).
pej(m(boy,sg),m(scoundrel,sg)).
pej(m(dog,sg),m(scoundrel,sg)).
/* hyponym substitution rule  */
d(Y,Z) :- hyp(X,Y),d(X,Z).
hyp(m(dog,sg),m(animal,sg)).
hyp(jumped,played). /* played is assumed to be hypon ym of jump here */
ni(X) --> [A],{isni(A),bet(A,X)}.
isni(pojkar).
bet(pojkar,m(boy,pl)).
isni(hundar).
bet(hundar,m(dog,pl)).
v(X) --> [A],{isv(A),bet(A,X)}.
isv(sprang).
bet(sprang,ran).
isv(gick).
bet(gick,went).
isv(hoppade).
bet(hoppade,jumped).
isv(lekte).
bet(lekte,played).
vt(X) --> [A],{isvt(A),bet(A,X)}.
isvt(slog).
bet(slog,hit).
isvt(tog).
bet(tog,took).
isvt(gillade).
bet(gillade,liked).
pf(X) --> [A],{ispf(A),bet(A,X)}.
ispf(pigga).
bet(pigga,alert).
ispf(trevliga).
bet(trevliga,nice).
ispf(dumma).
bet(dumma,silly).
/*
```

```
STORING (Lagra[]).
stores and prints factecule and (by extore) each factom */
lagra(X) :- sent(L,X,[]),print(L),asserta(L),extore(L).
extore(L) :- s(X,Y)=L,asserta(X),asserta(Y).
extore(L) :- s(X,Y,Z)=L,asserta(X),asserta(Y),asserta(Z).
extore(L) :- s(X,Y,Z,W)=L,asserta(X),asserta(Y),asserta(Z),asserta(W).

/* derives new facts from generic information;
the quantifier alla */
d(A,X) :- g(A,m(B,pl)),d(m(B,_),X),print(alla). /*  if generally (g) B
   is A and X is B then X is A */
/*  construction of potential factecules from factoms */
s(n(A,B),d(C,A)) :- n(A,B),d(C,A),print(impersonalitr).
s(n(A,B),d(X,A),d(Y,B)) :- n(A,B),d(X,A),d(Y,B),print(itr).
s(n(A,B),d(X,A),d(C,B),d(Y,C),d(Z,B)) :- n(A,B),d(X,A),d(C,B),d(Y,C),
   d(Z,B),print(coo).
s(n(A,B,C),d(X,A)) :- n(A,B,C),d(X,A),print(imperstr).
s(n(A,B,C),d(X,A),d(Z,C)) :- n(A,B,C),d(X,A),d(Z,C),print(passtr).
s(n(A,B,C),d(X,A),d(Y,B),d(Z,C)) :- n(A,B,C),d(X,A),d(Y,B),d(Z,C),
   print(tr).
/* derives relative np facts (d(X,Y) from factecules (s(X,Y,Z))) */
d(L,N1) :- L=s(n(P1,N1),X,Y),L. /* if N1 ran, N1  ran is true */
d(L,N1) :- L=s(n(P1,N1,N2),X,Y,Z),L. /* if N1 hit N2, N1 who hit N2 is
    true */
/* Answers declarative statement by ja(yes) if true */
answer(X) :- gsent(L,X,[]),L,print(ja),nl,print(L).
/* prints sentences which are true given primary and implicated facts */
truesay :- gsent(L,X,[]),L,print(X),;.
/* prints all sentences */
say :- sent(L,X,[]),print(X),;.
/* prints possible sentences and their factecules */
fsay :- sent(L,X,[]),print(X),print(L),;.
/* prints all factoms */
facts :- F=n(P,N),F,print(F),;.
facts :- F=n(P,N,M),F,print(F),;.
facts :- F=j(X,A),F,print(F),;.
facts :- F=d(X,Y),F,print(F),;.
```

REFERENCES

Cooper,R. Meaning representation in Montague Grammar and Situation Semantics. Workshop on Theoretical Approaches to Natural Language Understanding. May,1985. Halifax, Nova Scotia

Dahl,O. The interpretation of bound pronouns. In F.Karlsson (ed), Papers from the fifth Scandinavian conference of computational linguistics, Helsinki(1986:Dept of linguist.)

Horne,M. Focal prominence and the 'phonol ogical phrase' within some recent theories. Studia Linguistica 1986:2, 101-121

Johnson,M. & I.Klein. Discourse anaphora and parsing. Proc of COLING 1986, Bonn. (1986)

Kamp,H. A theory of truth and semantic representation. In Groenendijk,J.A.G., Janssen,T.M.V. & Stokhof,M.B.J. (eds), Formal methods in the study of language, Vol 136. Amsterdam: Mathematical Centre Tracts.

Karttunen,L. Discourse Referents, in J.McCawley (ed), Syntax and Semantics, vol 7. New York (1976:Academic Press)

Sidner,C. Focusing in the comprehension of definite anaphora. In Brady,A. & R.C.Berwick (eds), Computational Models of discourse. Cambridge, Mass.(1983:MIT press)

Sigurd,B. Text representation in a text production model. In Allen,S. (ed), Text processing, Proc. of Nobel Symposium, Stockholm. (1982: Almqvist & Wiksell International)