Clements, George. 1990. 'The sonority cycle and syllable organization'. In W. Dressler, H. Luschützky, O. Pfeiffer & J. Rennison (eds.), *Phonologica 1988*, 283-333. Cambridge University Press.

Jackendoff, Ray. 1977. *X-bar Syntax. A study of phrase structure*. Cambridge (Mass): MIT Press.

Jespersen, Otto. 1897-99. *Fonetik – en systematisk framstællning af læren om sproglyd*. København: Schubotheske Forlag.

Sigurd, Bengt. 1955. 'Rank order of consonants established by distributional criteria'. *Studia Linguistica* 9, 8-20.

Sigurd, Bengt. 1965. *Phonotactic structures in Swedish*. Lund: Uniskol.

Sigurd, Bengt. 1994. 'Fragments of a Swedish X-bar grammar (AXG)'. In Christer Platzack et al. (ed.), *Språkbruk, grammatik och språkförändring. Festskrift till Ulf Teleman*, 219 - 228.

Sigurd, Bengt & Birgitta Lastow. 1993. 'Fragments of a Japanese Appending X-bar Grammar (AXG)'. *Proc. Natural Language Processing Pacific Rim Symposium*. Fukuoka Institute of Technology, Japan.

Wood, Sidney. 1994. 'Syllable structure and the timing of speech gestures. An analysis of speech gestures from an X-ray motion film of Bulgarian speech'. *Proc. Third Congr. International Clinical Linguistics and Phonetics Association*. Dept of phonetics, University of Helsinki.

Lund University, Dept. of Linguistics
Working Papers 42 (1994), 209-224

Nexus Grammar (NEXG) for Swedish and English

Bengt Sigurd and Barbara Gawrońska

Introduction

Nexus Grammar (NEXG) is characterized by its focus on the syntactic unit consisting of the subject and the finite verb and since Jespersen 1924 known as the sentence nexus. The importance of the sentence nexus is clear from the fact that typological studies have shown that agreement between subject and predicate is the basic kind of agreement in languages. Other evidence of the importance of the nexus is the fact that some languages treat the nexus as a combinatory unit which is found in one straight (normal, right) form and one form where the order between the subject and the finite verb is inverted. NEXG is designed to treat languages for which a sentence nexus can be identified, e.g. Swedish and English. NEXG captures the function of word order as a marker of mode and topic. It also sheds some interesting light on the restricted use of inverted nexus in English where an auxiliary *(do,* the equivalent of Swedish *göra*, if no other) is always required.

NEXG is inspired by ideas found in Diderichsen's field grammar (Diderichsen 1946), in particular the division of sentences into three parts Thus, NEXG can be regarded as an implementation of a variant of his grammar. Diderichsen divides all sentences into three parts: the initial part called the fundament *(fundamentet)*, the *nexus* part *(neksusledet)* consisting of the subject and the finite verb with an optional nexus (sentence) adverbial and a content part *(inneholdsledet)* which includes the remainder of the sentence, e.g. objects, predicatives, infinitives, participles, verb particles and certain adverbs. Diderichsen's basic idea is that the ('underlying, deep structure') order of the parts in the nexus is fixed (in Danish as in Swedish): finite verb + subject np + nexus adverb (*vsa* in his notation) and he identifies a corresponding order *(VSA)* in the content part of the sentence. This (inverted) nexus word order is found ('as the surface order') when an adverb or an object occurs before the nexus and he explains this as the

result of a movement (transformation) with gaps in the corresponding empty positions. The normal straight word order: subject + finite verb + adverb, is considered to be the result of moving the subject to the initial fundament position. We will use the term straight or right interchangeably for this word order. Diderichsen's topological model has played a seminal role in many grammatical discussions in Scandinavia as witnessed to by all the articles written about or based on it (see Heltoft & Andersson 1986).

NEXG takes a slightly different approach. Sentences are assumed to contain a nexus and a content part, but normal straight order sentences are described by the straight order nexus without subject topicalization and yes/no questions are analyzed as inverted nexus. A topicalized element is only assumed in some special sentences with a (topicalized) initial adverb, object or verb. NEXG differs from Diderichsen's model in being written as a formal grammar directly in the programming language Prolog. NEXG can therefore be tested in analysis, generation and translation of sentences by computer.

The basic unit of NEXG is this nexus which thus comes in two variants: *nex(r,F)* and *nex(i,F)*. The letters *r* and *i* denote *right* and *inverted* word order, respectively. *F* is the functional representation. The inverted variant, *nex(i,F)*, where the subject and the finite verb have changed places may be derived by an inversion transformation from *nex(r,F)*, but the word strings allowed in the inverted nexus may also be generated by phrase structure rules. The functional representation (f-repesentation) is the same in its straight and inverted form. It has the normalized form: [subj(S),pred(P),sadvl(A1)]. The inverted nexus occurs typically after an initial adverb or object in Swedish and in yes/no questions both in Swedish and English.

The content part, *cont(F,F1)* may include nothing (an empty list []) or a particle after intransitive verbs. After transitive verbs the *cont* includes the object, and after copulas predicatives. Adverbs can also be added. The functional representation *(F)* of the nexus is percolated to the content constituent in order to allow rules of agreement (not written in the grammar presented) and to allow the selection of the proper verb complements to the verbs (particles, objects, predicatives, infinitives, participles, subordinate clauses).

Table 1 shows how Nexus Grammar would analyze some Swedish and English sentences. The values given are: *T(opic)*, Nexus order *(O)*, the functional representation of the nexus *(F)* and the functional representation

Table 1. Nexus Grammar analysis. The table shows the topic (T), the nexus order (O) and the f-representations of the nexus (F) and content parts (F1) for some Swedish and English sentences.

T(opic)	*nex(O(rder),F)*	*cont(F,F1)*
T=[]	F=[subj(m(he,sg)),pred(m(come,pres),sadvl([])]	F1=[]
	O=r *Han* *kommer.*	
T=[]	F=[subj(m(he,sg)),pred(m(come,pres),sadvl(m(perhaps,_))]	F1=[]
	O=i *Kommer* *han* *kanske?*	
T=[]	F=[subj(m(who,sg)),pred(m(come,pres),sadvl(m(not,_))]	F1=[]
	O=r*Vem* *kommer* *inte?*	
T=[]	F=[subj(m(who,sg)),pred(m(see,past)),sadvl([])	F1=[obj(m(he,sg))]
	O=r*Vem* *såg*	*honom?*
T=[m(he,sg)]	F=[subj(m(she,sg)),pred(m(see,past)),...]	F1=[obj([])]
	O=i *Honom* *såg*	*hon.*
T=[m(who,sg)]	F=[subj(m(he,sg))],pred(m(run,past)),...]	F1=[advl([m(to,[])]
	O=i *Vem* *sprang* *hon*	*till?*
T=[m(who,sg)]	F=[subj(m(he,sg))],pred(m(do,past))],..]	F1=[nonf(m(see,inf)),obj([])]
	O=i *Whom* *did* *he*	*see?*
T=[]	F=[subj(m(he,sg)),pred(m(see,past)),..]	F1=[obj(m(she,sg))]
	O=r *He* *saw*	*her.*
T=[]	F=[subj(m(he,sg)),pred(m(do,past)),..]	F1=[nonf(m(come,inf))]
	O=i *Did* *he*	*come?*

of the content *(F1)*. The word meaning is given as in Referent Grammar (Sigurd 1987) as a kind of Machinese English, which should be easy to understand, e.g. m(see,past) for English *saw*, Swedish *såg*, m(he,sg) for English *he* and *him* and Swedish *han* and *honom*. (the *m* stands for 'meaning'). What the generative grammar rules presented below do is to combine the two functional representations to obtain the whole f-representation, i.e. append the content representation *(F1)* to the nexus representation *(F)* taking certain conditions into account. The rules also specify the mode and topic.

As Referent Grammar, NEXG has a functional representation which may be used as an interlingua in automatic translation. As Referent

Grammar, NEXG does not derive a separate categorial representation (parsing tree).

NEXG is formalized directly in Prolog (using the DCG formalism) and its potential can therefore be studied easily. Some sample derivations and translations between a Swedish and an English version are shown below in the Demos. NEXG has been conceived of and developed within the Swedish MT-project Swetra (supported by HSFR and NUTEK) and it also employs ideas from Referent Grammar, the grammar which is the basis of Swetra. NEXG has not been developed into a full-scale grammar comparable to the grammars and lexicons used by Swetra in Weathra, Stocktra and Knittra (cf. Sigurd et al. 1992 and Nordner 1994).

A simple NEXG for Swedish

Nexus Grammar uses the generative rewriting rules of DCG and derives a functional representation *(F)* for all sentences (seen to the left of the arrow), but no categorial representation as mentioned. NEXG also registers mode (M) and topic (T) in the first two slots of the top predicate *s(entence)*. The mode values demonstrated here are *d(eclarative)* and *q(uestion)*. We will start our presentation by giving generative rules for some basic sentences. The grammatical categories used by *nex* and *cont* will be shown below. The lexicon presented below is restricted to words representing the main types of grammatical categories. Conditions are given within { } in DCG.

```
s(d,[],F2) --> nex(r,F),cont(F,F1),{append(F,F1,F2)}. % Han kommer (idag)
```

The rule states that a (Swedish) sentence such as *Han kommer idag* (literally: He comes today) consists of a nexus in the right order *(nex(r,F))* followed by a content constituent *(cont(F,F1))*. The functional representation of the nexus is found in the variable *F* and the functional representation of the content is found in *F1*. The functional representation of the content constituent *(F1)* is appended to the functional representation of the nexus *(F)* in order to get the full sentence representation: *F2* (the predicate *append* is a built-in predicate in Prolog). The following rule covers simple yes/no questions. The mode variable is now set at *q(uestion)*. The topic slot is set at no topic: []. The question uses an inverted nexus *(nex(i,F))*.


```
s(q,[],F2) --> nex(i,F),cont(F,F1),{append(F,F1,F2)}. %Kommer han (idag)?
```

The following rule covers the case where there is an initial question adverbial *(qadvp)* followed by an inverted nexus in Swedish *(När kommer han?)* as in English *(When is he coming?)*. The question word is registered as a topic. In the two sentences above no topic, [], was registered in the topic slot.The value of the adverbial phrase *(A1)* is added later to the functional representation of the *cont* in order to receive the normalized order of the functional roles in a list as in RG: [subj(S), pred(P), obj(O), advl(A1)].

```
s(q,A1,F3) --> qadvp(A1),nex(i,F),cont(F,F1),
      {append(F1,[advl(A1)],F2),append(F,F2,F3)}. % När kommer han?
```

The following rule illustrates the common case where there is another type of initial adverbial phrase. In this case the mode variable is set at *d(eclarative)*. The initial adverbial (A1) must not be missing (must not equal []).

```
s(d,A1,F3) --> advp(A1),{A1\=[]},nex(i,F),cont(F,F1),
      {append(F1,[advl(A1)],F2),append(F,F2,F3)}. % Idag kommer han
```

Types of nexus

We are now ready to look at the different types of nexus and cont(ent)s. The different types of nexus are illustrated by the following rules.

```
nex(r,[subj(A),pred(B),sadvl(A1)]) --> np(A),vi(B),sadvp(A1). % intr
nex(r,[subj(A),pred(B),sadvl(A1)]) --> np(A),aux(B),sadvp(A1). % aux
nex(r,[subj(A),pred(B),sadvl(A1)]) --> np(A),vt(B),sadvp(A1). % trans
```

The first rule above states that there is a nexus type with right order realized by an np occurring before an intransitive verb such as *kommer* (comes) and a sentence adverbial phrase (which is optional, i.e. may be []). The functional representation is seen to the left of the arrow.

NEXG makes a difference between ordinary (subject) np and object np *(npo)* although this difference is only realized with pronouns in Swedish and English. This was found to be of value when Referent Grammar was developed.

Types of inverted nexus

```
nex(i,[subj(A),pred(B),sadvl(A1)]) -->vi(B), np(A),sadvp(A1). % intr
nex(i,[subj(A),pred(B),sadvl(A1)]) -->vi(B), sadvp(A1),np(A). % intr
nex(i,[subj(A),pred(B),sadvl(A1)]) --> aux(B),np(A),sadvp(A1). % aux
nex(i,[subj(A),pred(B),sadvl(A1)]) --> aux(B),sadvp(A1),np(A). % aux
nex(i,[subj(A),pred(B),sadvl(A1)]) -->
                              vt(B), np(A),sadvp(A1). % trans: slog S inte
nex(i,[subj(A),pred(B),sadvl(A1)]) -->
                              vt(B), sadvp(A1),np(A). % trans: slog inte S
```

Note that the rules cover two word orders for inverted nexus illustrated by: *(Idag) kom regnet inte* and *Idag kom inte regnet.*

Nexus inversion transformation

The inverted nexus strings may be derived by a transformation from the right order nexus strings. The following is such a transformation.

```
nex(i,F,ListInv,Rest) :-
ListInv=[V|RestInv],
append(Np_and_Nexfin,Rest,RestInv),
append(Np,Nexfin,Np_and_Nexfin),Np\=[],
np(B,Np,[]),
append(Np,[V],SubjV),
append(SubjV,Nexfin,ListR),
nex(r,F,ListR,[]).
```

It should be noted, that all inverse strings are not simple inversions or straight order strings. The straight order is only *Regnet kommer inte* (literally: The rain comes not) and not **Regnet inte kommer* (in main clauses). But there are two almost equally good inverse equivalents with different placement of the adverb as mentioned above: *Kommer regnet inte* and *Kommer inte regnet* (if the subject is an unstressed pronoun e.g. *han* the latter order is acceptable only if *han* is stressed: *Kommer inte han*). If we want to cover the latter inverted pattern we need a transformation such as the following as well.

```
nex(i,F,ListInv,Rest) :-
ListInv=[V|RestInv],
append(Np_Sadvl_and_Nexfin,Rest,RestInv),
append(Sadvl,Np_and_Nexfin,Np_Sadvl_and_Nexfin),
sadvp(_,Sadvl,[]),
append(Np,Nexfin,Np_and_Nexfin),
```



```
np(B,Np,[]),
append(Np,[V],SubjV),
append(SubjV,Sadvl,SubjVSadvl),
append(SubjVSadvl,Nexfin,ListR),
nex(r,F,ListR,[]).
```

One may choose to derive the right order nexus from the inverted nexus in the same way. If only inverted nexus is used in the grammar (as with Diderichsen) one would have to allow empty subjects, subj([]), corresponding to topicalized noun phrases (cf. the treatment of topicalized objects below).

Nexus in subordinate clause (nex(s,F))

The word order in subordinate clauses deviates somewhat from the order in main clauses in Swedish. The difference is that the sentence adverb may occur immediately after the subject (before the finite verb) or before the subject and this difference may be dealt with by giving the following special types of nexus.

```
nex(s,[subj(A),pred(B),sadvl(A1)]) -->
                    np(A),sadvp(A1),vi(B). % att B inte kommer
nex(s,[subj(A),pred(B),sadvl(A1)]) -->
                    sadvp(A1),np(A),vi(B). % att inte B kommer
nex(s,[subj(A),pred(B),sadvl(A1)]) -->
                    np(A),sadvp(A1),aux(B). % att B inte kan (komma)
nex(s,[subj(A),pred(B),sadvl(A1)]) -->
                    sadvp(A1),np(A),aux(B). % att inte B kan (komma)
nex(s,[subj(A),pred(B),sadvl(A1)]) -->
                    np(A),sadvp(A1),vt(B). % att B inte slår
nex(s,[subj(A),pred(B),sadvl(A1)]) -->
                     sadvp(A1),np(A),vt(B). % att inte B slår
```

The special types of nexus needed in subordinate clauses may alternatively be derived by transformations like the following, which cover the two cases.

```
/* att X inte kan */
nex(s,F,ListInv,Rest) :-
append(Np_Sadvl_and_Nexfin,Rest,ListInv),
append(Np,Sadvl_and_Nexfin,Np_Sadvl_and_Nexfin),%hittar NP
np(_,Np,[]),
append(Sadvl,Nexfin,Sadvl_and_Nexfin), %hittar sadv
sadvp(_,Sadvl,[]),
```

```
append(Np,Nexfin,SubjV),
append(SubjV,Sadvl,ListR),
nex(r,F,ListR,[]).

/* att inte X kan */
nex(s,F,ListInv,Rest) :-
append(Sadvl_Np_and_Nexfin,Rest,ListInv),
append(Sadvl,Np_and_Nexfin,Sadvl_Np_and_Nexfin),%hittar sadvl
sadvp(_,Sadvl,[]),
append(Np,Nexfin,Np_and_Nexfin), %hittar sadv
np(_,Np,[]),
append(Np,Nexfin,SubjV),
append(SubjV,Sadvl,ListR),
nex(r,F,ListR,[]).
```

Subordinate nexus are used in clauses with subjunctions and relative clauses, which we have not implemented in this experimental grammar.

Types of contents

```
cont(F,F1) --> cont1(F,F1).
cont(F,F2) -->
      cont1(F,F1),advp(A1),{append(F1,[advl(A1)],F2)}. % adv added
cont1([subj(S),pred(P),sadvl(A1)],[]) -->[],{vi(P,Vf,[])}. % after intrans
cont1([subj(S),pred(P),sadvl(A1)],[obj(Ob)]) -->
      npo(Ob), {vt(P,Vf,[])}. % obj with trans verb
cont1([subj(S),pred(P),sadvl(A1)],[nonf(I)]) -->
      viinf(I), {aux(P,Vf,[])}. % infinitive with aux
```

Note how the percolated nexus representation permits the formulation of conditions on the combination of nexus and cont. The cont must thus be empty after vi, but it has to include an object after a transitive verb. After an auxiliary in nexus the cont must include a non-finite verb form, e.g. an infinitive. Adverbs may be added after all sentence types. This is handled by the top second rule.

Lexicon

```
vi(m(come,pres)) --> [kommer].
viinf(m(come,inf)) --> [komma].
vt(m(hit,pres)) --> [drabbar].
aux(m(may,pres)) --> [kan].
aux(m(do,pres)) --> [gör].
npo(F) --> np(F).
```



```
npo(m(he,sg)) --> [honom].
npo([]) --> []. % no obj (empty list [])
np(m(rain,sg)) --> [regn].
np(m(who,_)) --> [vem].
np(m(he,sg)) --> [han].
np(m(it,sg)) --> [det].
np(m(skåne,prop)) --> [skåne].
sadvp([]) --> []. % no sentence adverb
sadvp([m(not,_)]) --> [inte].
sadvp([m(perhaps,_)]) --> [kanske].
advp([]) --> []. % no adverb
advp([m(today,_)]) --> [idag].
advp([m(here,_)]) --> [här].
qadvp([m(when,_)]) --> [när].
advl([P,N]) --> p(P),npo(N).
advp([P,[]]) --> p(P). % defective prepositional phrase with no np
p(m(to,_)) --> [till].
```

Treating topicalized objects and verbs

Sentences with an initial (topicalized) object are a well-known problem in the grammars for Scandinavian languages. There are two cases: the initial np may be the object of a transitive verb or the object of a (stranded) preposition.

The object of a transitive verb is given its place in the functional representation by appending it to a functional representation with a transitive verb but empty object. This approach is also found in GPSG and resembles the defective categories used in Referent Grammar (empty objects have to be blocked in other cases, but we do not show all such details here).

The following rules cover cases of preposed (topicalized) objects.

```
s(d,Np,F3) --> npo(Np),nex(i,F),cont(F,F1),
      {on(obj([]),F1),% is there an empty obj?
      append(First,[obj([])],F1), % find the empty obj
      append(First,[obj(Np)],F2),
      append(F,F2,F3)}.
```

This rule would, for example, be able to analyze: *Honom drabbar det* (literally: Him it hits). The case with a stranded preposition is handled accordingly as shown by the following rule.

```
s(d,Np,F3) --> npo(Np),nex(i,F),cont(F,F1),
                {on(advl([P,[]]),F1), % prep phrase lacking np?
                append(First,[advl([P,[]])],F1),
                append(First,[advl([P,Np])],F2),
                append(F,F2,F3)}.
```

This rule would be able to analyse e.g. *Honom kommer det till* (literally: Him it comes to). This type of sentences with topicalized np is rare, except with question nouns as in: *Vem drabbar det? Vem kommer det till?*

There are interesting Swedish sentences with topicalized verbs such as: *Regnar/Regna gör det* (literally: Rains/Rain does it). The topicalized verb is mainly in the finite in modern Swedish, but the infinitive form is also possible – and it was recommended by earlier grammarians. These sentences can be handled in the same way, but an additional type of inverted nexus has to be established for Swedish using a form of *göra* 'do'. The following is a rule which covers this case.

```
s(d,m(V,T),F2) --> verb(m(V,T)),nex(i,F),cont(F,F1),{T=T1;T=inf},
      {on(pred(m(do,T1)),F),append([subj(S)],[pred(m(do,T1))|R],F),
            append([subj(S)],[pred(m(V,T1))|R],Fb),
            append(Fb,F1,F2)}.
```

Subordinate clauses are generated by the following rule, which needs a special nexus *(nex(s,F))* in Swedish (not in English) as the word order (adverb placement) is different in Swedish main and subordinate clauses (see above).

```
s(s,T,F2) -->
      nex(s,F),cont(F,F1),
      {append(F,F1,F2)}. % subordinate clause e.g. after subjunction att
```

A corresponding simple NEXG for English

We have also developed an English version for experimental purposes. It has similar rules except for the use of inverted nexus, which is employed very restrictively in English and must include an auxiliary – *do* if no other auxiliary is present. We use the prefix *e-* to distinguish English categories from Swedish ones.

```
es(d,[],F2) --> enex(r,F),econt(F,F1),{append(F,F1,F2)}.% He comes today

es(q,[],F2) --> enex(i,F),econt(F,F1),{append(F,F1,F2)}. % Does he come ?
```



```
es(q,A1,F3) --> eqadvp(A1),{A1\= []},enex(i,F),econt(F,F1),
      {append(F1,[advl(A1)],F2),append(F,F2,F3)}. % When does he come?

es(q,A1,F3) -->eadvp(A1),{A1\=[]},enex(r,F),econt(F,F1),
      {append(F1,[advl(A1)],F2),append(F,F2,F3)}. % Today he comes

es(d,Np,F3) --> enpo(Np),{Np=[m(who,_)]},
            enex(i,F),econt(F,F1),
            {on(obj([]),F1),% empty obj?
      append(First,[obj([])],F1), % find empty obj
      append(First,[obj(Np)],F2),
            append(F,F2,F3)}. % Whom does it hit

es(d,Np,F3) --> enpo(Np),{Np=[m(whom,_)]},
            enex(i,F),econt(F,F1),
            {on(advl([P,[]]),F1), % defect advl?
            append(First,[advl([P,[]])],F1),
            append(First,[advl([P,Np])],F2),
            append(F,F2,F3)}. % Whom does it come to
```

Types of English nexus

```
enex(r,[subj(A),pred(B),sadvl(A1)]) -->
      enp(A),evi(B),esadvp(A1),{A1\=m(not,_)}. % intr, no not allowed
enex(r,[subj(A),pred(B),sadvl(A1)]) -->
                  enp(A),evt(B),esadvp(A1),{A1\=m(not,_)}. % trans
enex(r,[subj(A),pred(B),sadvl(A1)]) -->
      enp(A),eaux(B),esadvp(A1). % aux may take not
```

Note that the adverb *not* is not allowed in sentences which only include a main verb and no auxiliary (*not* is traditionally said to require *do*-support, which is not completely true as it is allowed with all auxiliaries). English is said always to require straight word order, which is not true as all questions except subject questions have inverted word order.

Types of inverted English nexus (could be generated by a transformation as the Swedish ones)

```
enex(i,[subj(A),pred(B),sadvl(A1)]) --> eaux(B),esadvp(A1),enp(A).
      % When didn't he come?

enex(i,[subj(A),pred(B),sadvl(A1)]) --> eaux(B),enp(A),esadvp(A1).
      % When did he not come?
```

These rules state that only clauses with auxiliaries are allowed as inverted nexus in English. All such English clauses with auxiliaries may include *not.*

Types of English cont(ent)s

```
econt(F,F1) --> econt1(F,F1).
econt(F,F2) --> econt1(F,F1),eadvp(A1),{append(F1,[advl(A1)],F2)}. % adv
econt1([subj(S),pred(P),sadvl(A1)],[]) -->[],{evi(P,Vf,[])}. % empty after vi
econt1([subj(S),pred(P),sadvl(A1)],[obj(O)]) --> enpo(O), {evt(P,Vf,[])}.
econt1([subj(S),pred(P),sadvl(A1)],[nonf(I)]) --> eviinf(I), {eaux(P,Vf,[])}.
```

Lexicon

```
evi(m(come,pres)) --> [comes].
eviinf(m(come,inf)) --> [come].
evt(m(hit,pres)) --> [hits].
evtinf(m(hit,inf)) --> [hit].
eaux(m(do,pres)) --> [does].
eaux(m(may,pres)) --> [may].
enp(m(rain,sg)) --> [rain].
enp(m(it,_)) --> [it].
enp(m(who,_)) --> [who].
enp(m(skåne,prop)) --> [skåne].
enpo(m(he,sg)) --> [him].
enpo(m(who,_)) --> [whom].
enpo(F) --> enp(F).
enpo([]) --> [].
esadvp([m(not,_)]) --> [not].
esadvp([m(perhaps,_)]) --> [perhaps].
esadvp([]) --> [].
eadvp([m(today,_)]) --> [today].
eadvp([m(here,_)]) --> [here].
eadvp([m(when,_)]) --> [when].
eadvp([]) --> [].
eadvp([P,N]) --> ep(P),enpo(N).
eadvp([P,[]]) --> ep(P).
ep(m(to,_)) --> [to].
```

Interactions

```
nex(O,F,X,[]) % gives the Order and f-representation of X
s(M,T,F,X,[]) % gives the Mode, Topic and f-representation of X
```



```
setrans(X):-
print(X),nl,s(M,T,F,X,[]),print(M),nl,print(T),nl,print(F),nl,setransf(F,F1),
es(M,T,F1,X1,[]),print(F1),nl,print(X1),nl. % translates X and gives Mode,
Topic, and interlingua f-representations
```

The following is the Swedish-English transfer rule (setransf) which can be used when there is an identical functional reprsentation to be found in English.

```
setransf(F,F1) :- F1=F. % same F is tried first
```

The following is the transfer rule which introduces *do*-support. It is called when there is no identical f-representation in English. This is the case when *not* is found in a sentence where there is no auxiliary. Such sentences are blocked in the rules above. The other case occurs when an inverted nexus is required as in yes/no questions and after a question adverb or object noun. All these cases are covered by the same simple rule (which also generates English sentences with emphatic *do).* The rule states that a functional representation with a finite main verb may be expanded to a sentence where *do* takes on the same tense and the main verb occurs in the infinitive. As can be seen this rule only concerns the first elements of the functional representations leaving the rest (R) unchanged.

```
setransf(F,F1) :- F= [subj(S), pred(m(M, T)), sadvl(A1)|R],
                  F1=[subj(S), pred(m(do, T)), sadvl(A1),
                  nonf(m(M, inf))|R]. % do
```

Demo

```
:- s(M, T, F, [regn, drabbar, inte, skåne, idag], [])
      M = d, T = [], F = [subj(m(rain, sg)), pred(m(hit, pres)),
      sadvl(m(not, _)), obj(m(skåne, prop)), advl([m(today, _)])]

:- s(M, T, F, [honom, drabbar, det], [])
      M = d, T = m(he, sg), F = [subj(m(it,sg)), pred(m(hit, pres)),
      sadvl([]), obj(m(he, sg))]

:- s(M, T, F, [honom, kommer, det, till], [])
      M = d, T = m(he, sg), F = [subj(m(it, sg)), pred(m(come, pres)),
      sadvl([]), advl([m(to, _), m(he, sg)])]

:- s(M, T, F, [drabbar, regn, skåne, idag], [])
      M = q, T = [], F = [subj(m(rain, sg)), pred(m(hit, pres)), advl([]),
      obj(m(skåne, prop)), advl([m(today, _)])]
```

```
:- s(M, T, F, [honom, kan, det, inte, komma, till], [])
      M = d, T = m(he, sg), F = [subj(m(it,sg)), pred(m(may, pres)),
      sadvl(m(not, _)), nonf(m(come, inf)), advl([m(to, _), m(he, sg)])]

setrans([det, regnar, inte, idag]).
d
[]
[subj(m(it, sg)), pred(m(rain, pres)), sadvl([m(not, _)]), advl([m(today, _)])]
[subj(m(it, sg)), pred(m(do, pres)), sadvl([m(not, _)]), nonf(m(rain, inf)),
advl([m(today, _)])]
[it, does, not, rain, today]

setrans([när, kommer, han]).
q
[m(when, _)]
[subj(m(he, sg)), pred(m(come, pres)), sadvl([]), advl([m(when, _)])]
[subj(m(he, sg)), pred(m(do, pres)), sadvl([]), nonf(m(come, inf)),
advl([m(when, _)])]
[when, does, he, come]

setrans([regn, drabbar, skåne]).
d
[]
[subj(m(rain, sg)), pred(m(hit, pres)), sadvl([]), obj(m(skåne, prop))]
[subj(m(rain, sg)), pred(m(hit, pres)), sadvl([]), obj(m(skåne, prop))]
[rain, hits, skåne]

setrans([kommer, det, idag])
q
[]
[subj(m(it, sg)), pred(m(come, pres)), sadvl([]), advl([m(today, _)])]
[subj(m(it, sg)), pred(m(do, pres)), sadvl([]), nonf(m(come, inf)),
advl([m(today, _)])]
[does, it, come, today]

setrans([vem, drabbar, det]).
d
[]
[subj(m(who, _)), pred(m(hit, pres)), sadvl([]), obj(m(it, sg))]
[subj(m(who, _)), pred(m(hit, pres)), sadvl([]), obj(m(it, sg))]
[who, hits, it]
N°1
```



```
m(who, _)
[subj(m(it, sg)), pred(m(hit, pres)), sadvl([]), obj(m(who, _913))]
[subj(m(it, sg)), pred(m(do, pres)), sadvl([]), nonf(m(hit, inf)), obj(m(who,
_))]
[whom, does, it, hit]
N°2

setrans([vem, drabbar, det, inte]).
m(who, _)
[subj(m(it, sg)), pred(m(hit, pres)), sadvl([m(not, _)]), obj(m(who, _))]
[subj(m(it, sg)), pred(m(do, pres)), sadvl([m(not, _)]), nonf(m(hit, inf)),
obj(m(who, _))]
[whom, does, it, not, hit]
N°1

[subj(m(it, sg)), pred(m(do, pres)), sadvl([m(not, _)]), nonf(m(hit, inf)),
obj(m(who, _))]
[whom, does, not, it, hit]
N°2
```

Discussion

The capacity of NEXG is obvious from the rules and demonstrations. NEXG shows the character and function of nexus in straight and inverted form in Swedish and English. The grammar and lexicon presented here are, of course, incomplete. There are also various complexities in the verb phrase e.g. particles (as in *give in* 'yield') and series of non-finite auxiliaries (as in *will be able to begin swimming*), which we have not implemented in this experimental grammar. Such problems as well as verb agreement, coordinated clauses and relative clauses can be handled as in Referent Grammar. It would be interesting to see how the Nexus approach outlined in this paper would work for other languages. Swedish and English may be called nexus languages as a sentence unit can be identified in their syntax as demonstrated. Both the subject and the finite verb are obligatory in a nexus language and they occur close together. A few adverbs, in particular the marker of negation may occur close to the subject and finite verb. The nexus unit signals information about mode and truth. Russian and Georgian are examples of languages which do not include such a nexus unit. The predicate plays the fundamental role in those languages instead (see Gawrońska 1993). They may therefore be called Predicate languages.

Acknowledgement

We are grateful for comments by Mats Eeg-Olofsson and Birgitta Lastow, Swetra, Dept. of Linguistics, Lund.

References

Diderichsen, Paul. 1946. *Elementær dansk grammatik.* København: Gyldendal.

Gawrońska, Barbara. 1993. *An MT oriented model of aspect and article semantics.* Lund: Lund University Press.

Heltoft, Lars & John E. Andersson (eds.). 1986. 'Sætningsskemaet og dets stilling – 50 år efter'. *Nydanske studier & almen komminkationsteori* 16-17. Roskilde: Akademisk forlag.

Jespersen, Otto. 1924. *The philosophy of grammar.* New York: W.W. Norton & Company.

Nordner, Anders. 1994. 'Automatic translation of knitting instructions – KNITTRA'. *Working Papers* 42, 141-58. Dept. of Linguistics, Lund University.

Sigurd, Bengt. 1987. 'Referent Grammar (RG). A generalized phrase structure grammar with built-in referents'. *Studia Linguistica* 41:2, 115-135.

Sigurd, Bengt, Mats Eeg-Olofsson, Caroline Willners & Christer Johansson. 1992. *Automatic translation in specific domains: weather (Weathra) and stock market (Stocktra, Vectra)* (=*Praktisk lingvistik* 15). Dept. of Linguistics, Lund University.

Lund University, Dept. of Linguistics
Working Papers 42 (1994), 225-239

Mongolian Syllable Structure

Jan-Olof Svantesson

Introduction

In this article, the syllable structure of standard Khalkha Mongolian is described and exemplified in some detail. The related problems of how Mongolian words are divided into syllables, and how epenthetic vowels are introduced are treated in Svantesson forthc.

The description is based on recordings and observations of Ulaanbaatar speakers. The semi-official orthographic dictionary by Damdinsürèn & Osor 1983 has been used as an authoritative source of standard Khalkha forms, and the reverse alphabetical dictionaries by Vietze & Zenker 1976 and Bold 1976 have been especially useful for finding examples.

The phoneme system of standard Mongolian is given here for reference:

Vowels:	i	u		iː	uː			ui				
		ɷ			ɷː			ɷi				
	e	ɵ		eː	ɵː							
	a	ɔ		aː	ɔː		ai	ɔi				
Consonants:												
Labials		p	b					m				w
Palatalized labials		p′	b′					m′				w′
Dentals		t	d	c [ts]	z [dz]		s	n	l [ɮ]		r	
Palatalized dentals		t′	d′					n′	l′ [ɮ′]		r′	
Alveopalatals				č [tʃ]	ž [dʒ]		š [ʃ]					
Palatal												j
Velars			g				x	ŋ				
Palatalized velars			g′				x′					
Uvular			ɢ									

(Note that *z* and *ž* denote the affricates [dz] and [dʒ], respectively.)

My analysis differs from what is usually given in Western sources (e.g. Poppe 1951, 1970, Street 1963, Beffa & Hamayon 1975), but is rather similar to the analysis of many Mongolian, Russian and Japanese writers (e.g. Todaeva 1951, Nadeljaev 1957, Sanžeev 1959, Coloo 1976, Möömöö