Subject-Verb and Object-Verb Agreement in Hindi

Munendra Kumar
HOOMACL20170010
MACL
Semester II
munendra7777@gmail.com

Subject-Verb and Object-Verb Agreement in Hindi

Introduction

One goal common to human sentence processing theories is to develop a cross-linguistically applicable account of human parsing processes. Hindi also has certain interesting properties which make it a useful test case for evaluating current processing models. I decided to develop a program in Amzi! PROLOG to work upon the subject-verb and object-verb agreement in Hindi. In this program I have used 4 rules to generate sentences with the lexicon I have created in the program or user can also query for a sentence to check whether it is grammatical or not. Querying is limited to the lexicon I have created; program will not provide output for the sentences whose POS are in the lexicon. However, user can edit the program and include the desired lexicon.

[Shravan Vasishth, Processing Hindi Center Embeddings, 2002]

Self-center-embedding constructions (hereafter, SCEs) are grammatical structures in which a constituent occurs medially within a larger instance of the same kind of syntactic category.

Single Embedded Sentences

- Don't you find [that sentences [that people you know produce] are easier to understand]?
- Sitaa-ne Hari-ko kitaab khariidne-ko kahaa
  Sita-erg Hari-dat book buy-inf said
  'Sita told Hari to buy a/the book.'

Double Embedded Sentences

- Sitaa-ne Hari-ko [Ravi-ko [kitaab khariid-ne-ko] bolne-ko] kahaa
  Sita-erg Hari-dat Ravi-dat book buy-inf tell-inf told\
  'Sita told Hari to tell Ravi to buy a/the book.'

Non-embedded Sentences with Ergative and Dative case

- Raam-ne aam-ko todaa
  Ram-erg mango-dat plucked
  'Ram plucked the mango.'

Simple Sentences

- Raam aam todtaa hai
  Ram Mango plucks
  Ram plucks a mango.

Methodology

To check for all these four types of sentences whether they are grammatical or not, I used PROLOG'S Definite Clause Grammar (DCG) and Concatenate Function.

A DCG (definite clause grammar) is a phrase structure grammar annotated by Prolog variables are translated by the Prolog interpreter into normal Prolog clauses.

Example: n([sitaa],f).

obj([aam],G).

Concatenate Function in PROLOG is used to append two lists together.

Example: ?- conc([a,b,c],[1,2,3],[a,b,c,1,2,3])

Linguistic Analysis

The noun class system of Hindi assigns a masculine or feminine gender to each noun in the lexicon and has a binary number distinction of singular and plural features in the grammar.

Verb agrees with the gender of the subject noun in the grammar.
For example: *Sitaa(fem.) rotii khatii hai.*
Sita bread eats.
'Sita eats bread.'

Raam(mas.) rotii khataa hai.
Raam bread eats.
Raam eats bread.

Verb agrees with the gender of the direct object in the grammar when the subject is ergative case marked.
For example: *Raam-ne aam(mas.) todaa.*
Ram-erg mango plucked.
Ram plucked a mango.

Raam-ne jalebii(fem.) khaayi.
Raam-erg jalebii ate.
Raam ate jalebii.

Verb takes the neutral form when the subject is ergative case marked and the direct object is dative case marked. This type of sentence construction shows the definiteness of the direct object.

For example: *Raam-ne aam-ko todaa.*
Raam-erg mango-dat plucked.
Raam plucked the mango.

Sitaa-ne aam-ko todaa.
Sitaa-erg mango-dat plucked.
Sitaa plucked the mango.

Verb takes the neutral form when the subject is ergative case marked, direct object is dative case marked and auxiliary verb is dative/accusative case marked.

For example: *Raam-ne Sitaa-ko aam khareedne-ko kahaa.*
Raam-erg Sitaa-dat mango buy-inf told.
Raam told Sitaa to buy mango/es.

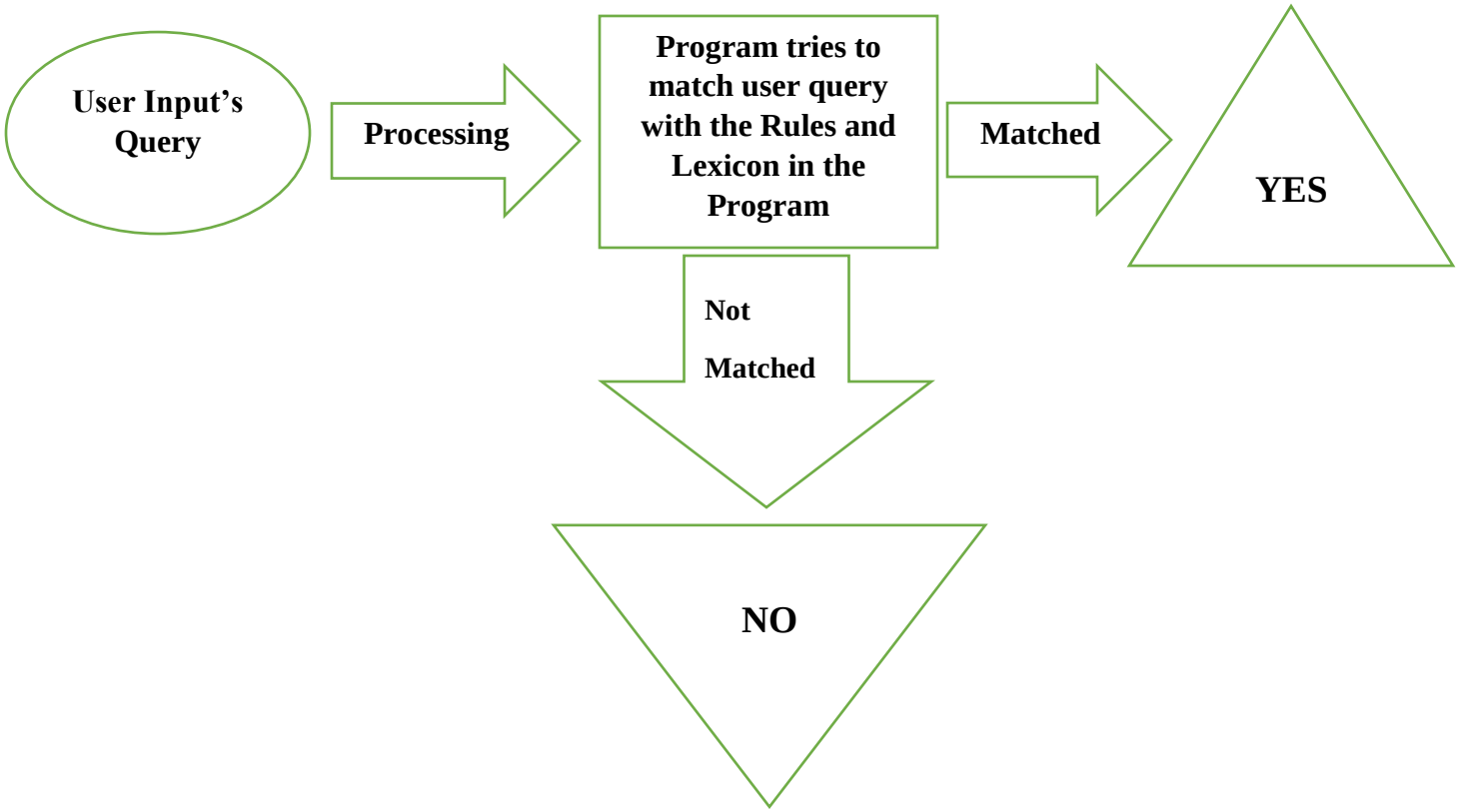
Sitaa-ne Raam-ko aam khareedne-ko kahaa.
Sitaa-erg Raam-dat mango buy-inf told.
Sitaa told Raam to buy mango/es.

Main Verb takes the gender of the most recent verb in the sentence when the sentence is double center embedded.

For example: *Raam-ne Rohan-ko Sitaa_ko aam khareedne-ko kahney-ki salaah-di.*
Raam-erg Rohan-dat Sitaa-dat mango/es buy-inf tell-inf advised.
Raam advised Rohan to tell Sita to buy a mango/es.

Sitaa-ne Rohan-ko Raam_ko aam khareedne-ko kahney-ko bola.
Sitaa-erg Rohan-dat Raam-dat mango/es buy-inf tell-inf told.
Sitaa told Rohan to tell Raam to buy a mango/es.

Algorithms



User inputs the query in the PROLOG LISTENER, query is then processed using the rules and lexicon in the program. If there is a match between the query and the POS in the lexical. Program shows output as "YES" else "NO".

I used the following algorithms in PROLOG to check for the grammaticality of the sentence.

% Rule 1 for sentences like: [raam, aam, todtaa, hai] and [raam_ne, aam_ko, todaa].

```
concat([],List,List).
concat([Head|List1],List2,[Head|List3]):-
concat(List1,List2,List3).

s(X,[]):-
np(Y,G),
vp(Z,G),
```

```
concat(Y,Z,X).

np(X,G):-
n(Y,G),
obj(Z,G),
concat(Y,Z,X).

vp(X,G):-
v(Y,G),
aux(Z,G),
concat(Y,Z,X).

% Rule 2 for sentences like:
 [raam_ne, sitaa_ko, aam, diye] and [raam_ne, sitaa_ko, kapdey, khareedne_ko, kahaa] .

s-->np(Num,Per,Gen),vp(Num,Per,Gen,T).
np(Num,Per,Gen)-->n_erg(Num,P,Gen),obj_dat(Num,P,Gen).
np(Num,Per,Gen)-->n_erg(Num,P,Gen),obj(Num,P,Gen).
vp(Num,Per,Gen,T)-->tv(Num,Per,Gen,T).
vp(Num,Per,Gen,T)-->mav(Num,Per,f,pres).

%Rule 3 for sentences like:
 [raam_ne, sitaa_ko, kavitaa, padhaai] and [raam_ne, gitaa_ko, aam, khareedne_ko, kahaa].

s-->kp(Num,Per,Gen),pvp(Num,Per,Gen,T).
kp(Num,Per,Gen)-->n_erg(Num,P,Gen),n_dat(Num,P,Gen),obj(Num,P,Gen).
pvp(Num,Per,Gen,T)-->perfv(Num,Per,Gen,Per).
kp(Num,Per,Gen)-->n_erg(Num,P,Gen),n_dat(Num,P,Gen),obj(Num,P,Gen).
pvp(Num,Per,Gen,T)-->v_erg(Num,Per,Gen,Per),mv(Num,Per,Gen,Per).

%Rule 4 for sentences like:
 [raam_ne, rohan_ko, sitaa_ko, aam, khareedne_ko, kahney_ki, salaah_di].

s-->kcp(Num,Per,Gen),cvp(Num,Per,Gen,T).

kcp(Num,Per,Gen)-->n_erga(Num,P,Gen),n_dative(Num,P,Gen),n_accu(Num,P,Gen),
obje(Num,P,Gen).

cvp(Num,Per,Gen,T)-->v_erga(Num,Per,Gen,Per),v_gen(Num,Per,Gen,Per),
mdv(Num,Per,Gen,Per).
```

Output

To check for the grammaticality of any sentence from the following rules and lexicon the user needs to enter a query in the PROLOG LISTENER as follows:

?-s([raam, aam, todtaa, hai],[]). (for sentences which follow Rule 1)

?-s([raam_ne, sitaa_ko, aam, diye] and [raam_ne, sitaa_ko, kapdey, khareedne_ko, kahaa],[].)
(for sentences which follow Rule 2)

?-s([raam_ne, sitaa_ko, kavitaa, padhaai] and [raam_ne, gitaa_ko, aam, khareedne_ko, kahaa],[]). (for sentences which follow Rule 3)

?- s([raam_ne, rohan_ko, sitaa_ko, aam, khareedne_ko, kahney_ki, salaah_di],[]).
(for sentences which follow Rule 4)

User can also generate grammatical sentences with the lexicon in the program by backtracking, using following query in PROLOG LISTENER:

?-s([X,[]).

Limitations

The program is limited to the lexicon provided in the program since all the parts of speech in Hindi are not included in the program. The part of speech used in the lexicon can only be used to query in the PROLOG LISTENER otherwise the program will return the answer as 'NO'. It is to be kept in mind that this program will respond to the queries in a specific format for different types of sentence constructions. (see: output: page5)

Appendix

Here is the complete Prolog Program:

```
%       Munendra Kumar
%       MACL Semester 2
%       HOOMACL20170010
%       Prolog Semester End Submission
%       Topic- Subject-Verb,Object-Verb Agreement in Hindi with simple sentences,single and double embedded dative/accusative case.



        % Rule 1 for sentences like: [raam, aam, todtaa, hai] and [raam_ne, aam_ko, todaa].


concat([],List,List).
concat([Head|List1],List2,[Head|List3]):-
concat(List1,List2,List3).


s(X,[]):-
np(Y,G),
vp(Z,G),
concat(Y,Z,X).
```

```
np(X,G):-
n(Y,G),
obj(Z,G),
concat(Y,Z,X).

vp(X,G):-
v(Y,G),
aux(Z,G),
concat(Y,Z,X).
```

% Rule 2 for sentences like: [raam_ne, sitaa_ko, aam, diye] and [raam_ne, sitaa_ko, kapdey, khareedne_ko, kahaa] .

```
s-->np(Num,Per,Gen),vp(Num,Per,Gen,T).
np(Num,Per,Gen)-->n_erg(Num,P,Gen),obj_dat(Num,P,Gen).
np(Num,Per,Gen)-->n_erg(Num,P,Gen),obj(Num,P,Gen).
vp(Num,Per,Gen,T)-->tv(Num,Per,Gen,T).
vp(Num,Per,Gen,T)-->mav(Num,Per,f,pres).
```

%Rule 3 for sentences like: [raam_ne, sitaa_ko, kavitaa, padhaai] and [raam_ne, gitaa_ko, aam, khareedne_ko, kahaa].

```
s-->kp(Num,Per,Gen),pvp(Num,Per,Gen,T).
kp(Num,Per,Gen)-->n_erg(Num,P,Gen),n_dat(Num,P,Gen),obj(Num,P,Gen).
pvp(Num,Per,Gen,T)-->perfv(Num,Per,Gen,Per).
kp(Num,Per,Gen)-->n_erg(Num,P,Gen),n_dat(Num,P,Gen),obj(Num,P,Gen).
pvp(Num,Per,Gen,T)-->v_erg(Num,Per,Gen,Per),mv(Num,Per,Gen,Per).
```

%Rule 4 for sentences like: [raam_ne, rohan_ko, sitaa_ko, aam, khareedne_ko, kahney_ki, salaah_di].

s-->kcp(Num,Per,Gen),cvp(Num,Per,Gen,T).

kcp(Num,Per,Gen)-->n_erga(Num,P,Gen),n_dative(Num,P,Gen),n_accu(Num,P,Gen),obje(Num,P,Gen).

cvp(Num,Per,Gen,T)-->v_erga(Num,Per,Gen,Per),v_gen(Num,Per,Gen,Per),mdv(Num,Per,Gen,Per).

%lexicons

%Lexicon 1

n([sitaa],f).

obj([aam],G).

v([todtii],f).

v([khaatii],f).

v([khaataa],m).

aux([hai],G).

n([raam],m).

v([todtaa],m).

%masculine nouns and objects

n(sing,third,m)-->[raam].

n(sing,third,m)-->[shyaam].

obj_dat(sing0,P,m)-->[aam_ko].

obj_dat(sing0,P,m)-->[amrood_ko].

obj_dat(sing0,P,m)-->[anaar_ko].

obj(sing0,third,Gen)-->[aam].

obj(sing0,third,Gen)-->[amrood].

obj(sing0,third,Gen)-->[anaar].

tv(sing0,third,Gen,perf)-->[todaa].

tv(sing0,third,Gen,perf)-->[khaayaa].

%Verbs with ergative case

v_erga(Num,third,Gen,Per)-->[khareedne_ko].

v_gen(Num,third,Gen,Per)-->[kahney_ki].

v_erga(Num,third,Gen,Per)-->[khareedne_ko].

n_erga(Num,third,m)-->[raam_ne].

n_erga(Num,third,m)-->[shyaam_ne].

v_erg(sing00,third,Gen,Per)-->[khareedne_ko].

mdv(Num,third,Gen,Per)-->[salaah_di].

mv(sing00,third,Gen,Per)-->[kahaa].

%Nouns with accusative case

n_accu(Num,third,f)-->[raam_ko].

n_accu(Num,third,f)-->[shyaam_ko].

n_accu(Num,third,m)-->[sitaa_ko].

n_accu(Num,third,m)-->[gitaa_ko].

%Objects(Nouns)

obje(Num,third,Gen)-->[aam].

obje(Num,third,Gen)-->[amrood].

obje(Num,third,Gen)-->[anaar].

obje(Num,third,Gen)-->[khilauney].

obje(Num,third,Gen)-->[kapdey].

%Nouns with Dative case

n_dative(Num,third,m)-->[rohan_ko].

n_dative(Num,third,m)-->[sohan_ko].

n_dat(Num,third,f)-->[raam_ko].

n_dat(Num,third,f)-->[shyaam_ko].

n_dat(Num,third,m)-->[sitaa_ko].

n_dat(Num,third,m)-->[gitaa_ko].

%Objects(Nouns)

obj(sing00,third,Gen)-->[aam].

obj(sing00,third,Gen)-->[amrood].

obj(sing00,third,Gen)-->[anaar].

obj(sing00,third,Gen)-->[khilauney].

obj(sing00,third,Gen)-->[kapdey].

obj(plu1,third,Gen)-->[aam].

obj(plu1,third,Gen)-->[amrood].

obj(plu1,third,Gen)-->[anaar].

obj(plu2,third,Gen)-->[khilauney].

obj(plu2,third,Gen)-->[kapdey].

obj(sing1,third,Gen)-->[paani].

obj(sing2,third,Gen)-->[khaanaa].

obj(sing3,third,Gen)-->[sharaab].

obj(sing4,third,Gen)-->[baazaar].

obj(sing4,third,Gen)-->[shahar].

obj(sing4,third,Gen)-->[desh].

obj(sing5,third,Gen)-->[duniya].

obj(sing6,third,Gen)-->[ghar].

obj(sing6,third,Gen)-->[vidyalaya].

obj(sing6,third,Gen)-->[vishwavidyalaya].

obj(sing7,third,Gen)-->[saaikil].

obj(sing7,third,Gen)-->[motarsaaikil].

obj(sing7,third,Gen)-->[kaar].

obj(sing8,third,Gen)-->[likhnaa].

obj(sing8,third,Gen)-->[bolnaa].

obj(sing8,third,Gen)-->[padhnaa].

obj(sing9,third,Gen)-->[kitaab].

obj(sing9,third,Gen)-->[kavitaa].

%Nouns with Ergative case

n_erg(Num,third,m)-->[raam_ne].

n_erg(Num,third,m)-->[shyaam_ne].

n_erg(Num,third,f)-->[sitaa_ne].

n_erg(Num,third,f)-->[gitaa_ne].

%perfective verbs

perfv(plu1,third,Gen,Per)-->[diye].

perfv(plu2,third,Gen,Per)-->[dilaaye].

perfv(sing1,third,Gen,Per)-->[pilaayaa].

perfv(sing2,third,Gen,Per)-->[khilaayaa].

perfv(sing3,third,Gen,Per)-->[pilaayi].

perfv(sing4,third,Gen,Per)-->[ghumaayaa].

perfv(sing5,third,Gen,Per)-->[ghumaayii].

perfv(sing6,third,Gen,Per)-->[bulaayaa].

perfv(sing6,third,Gen,Per)-->[bhejaa].

perfv(sing7,third,Gen,Per)-->[dilaayii].

perfv(sing7,third,Gen,Per)-->[chalaanii_sikhaayii].

perfv(sing8,third,Gen,Per)-->[sikhaayaa].

perfv(sing9,third,Gen,Per)-->[padhaai].