TECHNICAL

The PAPPI System

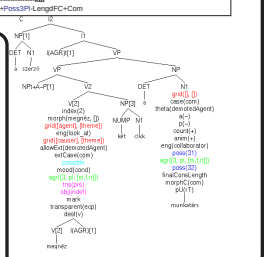
Sandiway Fong, Depts. of Linguistics and Computer Science

Hungarian

EXAMPLE:

a szerző-k megnéz-et----het----Ø------né----nek----Ø két cikk---et
the author-Agr3Pl look_at---Caus-Possib-tns(prs)-Cond-Agr3Pl-Obj(indef) two article-Acc

a munkatárs-a----------ik----------kal
the colleague----Poss3Sg-Agr3Pl+Poss3Pl-LengdFC+Com



Simple Morphology: Markers and Features

Of-Insertion, Chomsky [KofL,86]:

... proud *of* him

case(gen)

Generators

Parse PF

Parse S-Structure

Markers

```
lex(to,mrkr,[right(v,morph(_,[]),inf([])])).
lex(of,mrkr,[right(np,case(gen),[])]).
```

Contraction Rules

```
contraction([],can,'t,[can,neg]).
```

読まない
yo–manai

```
contraction(vEnd,X+nai,[X=pf([require(vNStem)]),negnpast]).
contraction(vNStem,X+ma,[X=word(base(mu))$$prefix(ma)]]).
```

Recovery of Phrase Structure

$\overline{X}$ + X^0-Movement Rules

```
rule XP -> [XB|spec(XB)] ordered specFinal st max(XP), proj(XB,XP).
rule XB -> [X|compl(X)] ordered headInitial(X)
                                  st bar(XB), proj(X,XB), head(X).

rule v(V) moves_to i provided agr(strong), finite(V).
rule v(V) moves_to i provided agr(weak), v has_feature aux.
```

Backtracking Canonical LR(1)-based Shift/Reduce Parser

```
machine(accept,[],[SS],_,accept,[],SS,_).
machine([State|CS],[Input,SS,ES,CS2,I2,SS2,ES2] :-
          Input = [I|Is], ss0Cat(I,C),
          action(State,$,[State|CS],[I|Is],SS,ES,CS1,I1,SS1,ES1),
          machine(CS,[I1,SS1,ES1,CS2,I2,SS2,ES2).
```

(State,Lookahead)

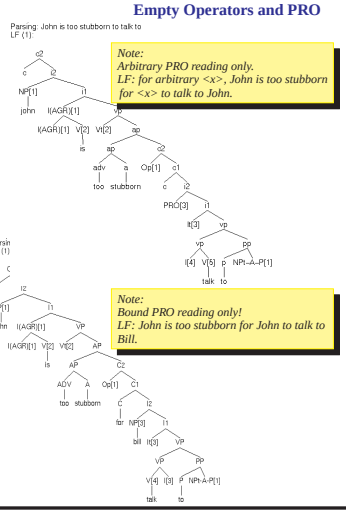
```
% 138:$ no conflict
action(138,$,[_629,_631,_633,_551|_552],_540,[_608,_604|_554],_542,[_549,_551|_
552],_540,[[c2$_561$_562,_604,_608]|_554],_546):-close(_542,_146),specifi-
erR(_608,_604),_608 has_features _562,transi-
tion(_551,c2,_549),c2rgoal([c2$_561$_562,_604,_608]).% reduce:c2-
>[left:c2,specCPwhadv,c1]/s2/2
% 138:c1 ERROR
% 137:_2180 no conflict
action(137,_180,[_290,_292,_199|_200],_188,[_263,_259|_202],_190,[_197,_199|_20
```

LR Actions

GB Theory: English Examples

Empty Operators and PRO

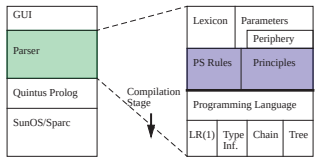
Parsing: John is too stubborn to talk to
LF (1)



Note:
Arbitrary PRO reading only.
LF: for arbitrary <x>, John is too stubborn for <x> to talk to John.

Note:
Bound PRO reading only!
LF: John is too stubborn for John to talk to Bill.

Architecture



Avoid hardwiring

Inference engine: unification. Choice points: single thread, parallel

Direct interpretation: infeasible. Specialization permits re-targeting

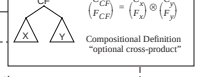
Sicstus Prolog (native: `fastcode`) port: factor of 0.425

Recovery of Movement

Determination of chain features

```
chain(NP[1], lasl, [[pp, vp, vp, i1], []])
chain(NP[1], medial, [[c2, vp, vp, i1, i2, c1], []])
chain([], head, [])
```

index(1)



$$\binom{C_{CF}}{F_{CF}} = \binom{C_X}{F_X} \otimes \binom{C_Y}{F_Y}$$

Compositional Definition
"optional cross-product"

Soundness, completeness

```
"Wh" Chain Formation
resolveTraces produces [Chains,Free]
      cases CF where
        CF with_constituents X and Y st
              X produces [Cx,Fx],
              Y produces [Cy,Fy],
              CF produces [Ccf,Fcf]
           then  crossProduct(Cx,Fx,Cy,Fy,Ccf,Fcf)
           finally baseCase(CF,Cm,Fm),
                   crossProduct,Ccf,Fcf,Cm,Fm,Chains,Free)
      else moves(CF) then baseCase(CF,Chains,Free).

baseCase(X,    [X],    []) :- ec(X).          baseCase(X,    [],    [X]).

crossProduct(Cx,Fx,Cy,Fy,Chains,Free) :-
        extendChains(Cx,Fx,Fx,Cx,Fy,Fy).
```

Recovery of Movement contd.

Complexity:

NPs	Indexings	NPs	Indexings
1	1	7	887
2	2	8	4140
3	5	9	21147
4	15	10	115975
5	52	11	678570
6	203	12	4123597

Upper-bounded by *Bell's Exponential Number*

$$B_n = \sum_{m=1}^{n} \sum_{k=0}^{m} \frac{(-1)^{m-k}}{(m-k)!\,k!} k^n \qquad \frac{m_n{}^n e^{m_n - n - \frac{1}{2}}}{\sqrt{\ln n}} \qquad m_n \ln m_n = n - \frac{1}{2}$$

Partial Solution: Merge constraints

m	Subjacency
m	Lowering Filter
i	Wh-movement in Syntax

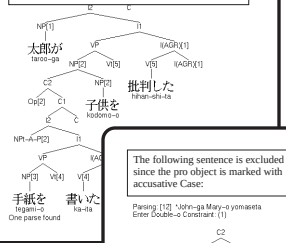
```
subjacency in_all_configurations CF where
        isTrace(CF), upPath(CF,Path) then lessThan2BoundingNodes(Path).

loweringFilter in_all_configurations CF where
        isTrace(CF), downPath(CF,Path) then Path=[].
```

Kanda COE: Double-O Constraint

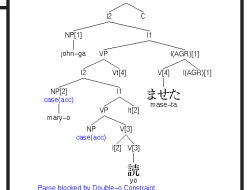
Parsing: [11] Taroo–ga tegami–o kaita kodomo–o hihanshita

Here, the two *o*-marked NPs are separated by a clause boundary (CP). It goes through as expected:



The following sentence is excluded since the pro object is marked with accusative Case:

Parsing: [12] *John–ga Mary–o yomasetta
Enter Double–o Constraint (1)



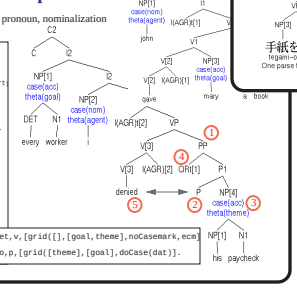
Plug-Ins: Double Object Constructions and the Zero Morpheme G

Anaphor binding, NPI, Quan/bound pronoun, nominalization

[Ch5, *Zero Syntax*, Pesetsky]

1. X-bar theory
   VP-internal structure
   `bar(v1). max(vp). proj(v1,vp).`
   `rule compl(X) -> [pp] st lexProperty`
   PP-internal structure
   `spec(p1,np). spec(p1,[]).`
2. The empty preposition G
   `rule p(Fs) -> [] st emptyP(Fs,ES).`
   `[grid([goal],[theme]),`
   `specR([goal(animate(X),X)]),`
   `docase(acc)]`
3. + 4. Case and θ-theory
   Mediated θ-selection
   V -> [Spec,PP]
   Co-opt ECM for acc Case
5. G incorporates into V
   V pushes Roles onto stack
   G/P pops Roles off stack,
   checks them off.

```
lex(get,v,[grid([],[goal,theme],noCasemark,ecm]
lex(to,p,[grid([theme],[goal],doCase(dat)].
```



Conditions on Trees and Domains

Binding Conditions A and B:

An anaphor must be A-bound in its GC.
A pronominal must not be A-bound in its GC.

Governing Category (GC):

GC(α) is the smallest NP or IP containing:
(A) α, and
(B) a governor of α, and
(C) an accessible SUBJECT for α.

```
conditionA(A) :- condA(A,B).
condA(A,B) :-
        A has_constituents C, !,
        condAl(C,D), condAml(D,A,E),
        (anaphor(A)
        -> condAmo(A,E,B)
        ;  B=E).
condA(A,B) :-
        anaphor(A), !,
        initgc(A,C), B=[C].
condA(A,[]).
gc(A,gc(A,B,C),D) :- %% GC(A)
        (var(B) -> (governs(B,A,
        (var(C) -> (localyaccSu
completegc(gc(A,B,C),D) :-
        nonvar(A), nonvar(B), no
        cat(D,E), member(E,[np,2
initgc(A,gc(A,B,C)).
```

Code: program clichés

```
conditionA in_all_configurations CF where
        anaphor(CF) then gc(CF,GC), aBound(CF,GC).
anaphor(NP) :- NP has_feature apos, NP has_feature a(+).

gc(X) smallest_configuration CF st cat(CF,C), member(C,[np,i2])
        with_components
                X,
                G given_by governs(G,X,CF),
                S given_by accSubj(S,X,CF).
```

Implementation: bottom-up domain instantiation

University of Arizona