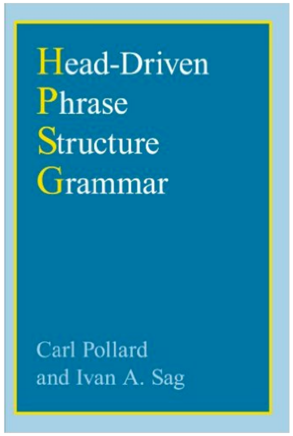
Theoretical Linguistics and Grammar Engineering as Mutually Constraining Disciplines

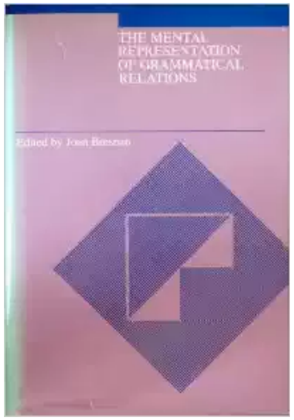
Tracy Holloway King

2016-07-26

Grammar Engineering Programs

- LFG ParGram
- HPSG Grammar Matrix and DELPH-IN
- Goals:
  - Test underlying tenets of the theory: universality
  - Build resources to be used in applications





ParGram and ParSem

- Broad coverage grammars for diverse languages
  - Began with English, German, and French
- XLE as development platform
- Test universality and parallel nature of functional-structures

- Issues around linguistic analysis
  - adjectives, copulas, negation, auxiliaries, coordination
- Issues around grammar engineering
  - morphology-syntax interface
  - ambiguity

Grammar Matrix

- Broad coverage, precision grammars for diverse languages
- Leverage expertise from existing grammars to build new ones
- Develop grammars with a common MRS format
- Exchange data and analyses of phenomena
- Tools for field linguists to build grammars
  - test hypotheses
  - encode results

Grammar Engineering to Confirm Hypotheses

- Low effort grammar engineering
  - Platform issues may limit usefulness

- Some examples:
  - LFG Linking Theory (Butt, Dalrymple, Frank 1997)
  - LFG resumptive pronouns (Asudeh 2004)
  - HPSG resumption & extraction (Crysmann 2015) & tone (Crysmann 2009)
  - LFG adjective coordination (Beyaev, Dalrymple, Lowe 2015)

Indeterminacy by Underspecification:
Dalrymple, King, Sadler 2009

- How to formally encode indeterminacy
  - Indeterminate forms can simultaneously satisfy conflicting requirements

- Proposal: value of an indeterminate feature (case) is a complex, possibly underspecified feature structure
  - allows incremental, monotonic refinement
  - uses only atomic, boolean-valued features
  - covers indeterminate arguments and predicates

Underspecified, indeterminate case: German

Papageien
parrots
NOM/ACC/DAT/GEN
'parrots' (nominative, accusative, dative, or genitive)

Indeterminate nouns with case-assigning predicates

a.

Er	findet	Papageien.
he	finds	parrots
	OBJ=ACC	NOM/ACC/DAT/GEN

'He finds parrots.'

b.

Er	hilft	Papageien.
he	helps	parrots
	OBJ=DAT	NOM/ACC/DAT/GEN

'He helps parrots.'

c.

Er	findet	und	hilft	Papageien.
he	finds	and	helps	parrots
	OBJ=ACC		OBJ=DAT	NOM/ACC/DAT/GEN

'He finds and helps parrots'

Proposed structure for case on German nouns

Determinate accusative case:

$$\left[\text{CASE} \begin{bmatrix} \text{NOM} & - \\ \text{ACC} & + \\ \text{GEN} & - \\ \text{DAT} & - \end{bmatrix}\right]$$

Determinate dative case:

$$\left[\text{CASE} \begin{bmatrix} \text{NOM} & - \\ \text{ACC} & - \\ \text{GEN} & - \\ \text{DAT} & + \end{bmatrix}\right]$$

Indeterminate case:

$$\left[\text{CASE} \begin{bmatrix} \text{NOM} \\ \text{ACC} \\ \text{GEN} \\ \text{DAT} \end{bmatrix}\right]$$

Indeterminate noun with dative verb

a.

Er	hilft	Papageien.
he	helps	parrots
	OBJ=DAT	NOM/ACC/GEN/DAT

'He helps parrots.'

b.

$$\left[\text{OBJ}\ \left[\begin{array}{ll}\text{PRED} & \text{'PARROTS'} \\ \text{CASE} & \left[\begin{array}{ll}\text{NOM} & \\ \text{ACC} & \\ \text{GEN} & \\ \text{DAT} & +\end{array}\right]\end{array}\right]\right]$$

Coordinating indeterminate noun with acc and dat verb

a.	Er	findet	und	hilft	Papageien.
	he	finds	and	helps	parrots
		OBJ=ACC		OBJ=DAT	NOM/ACC/DAT/GEN

‘He finds and helps parrots’

b.

$$\left\{ \begin{array}{l} \begin{bmatrix} \text{PRED} & \text{‘FIND’} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{‘PARROTS’} \\ \text{CASE} & \begin{bmatrix} \text{NOM} & \\ \text{ACC} & + \\ \text{GEN} & \\ \text{DAT} & + \end{bmatrix} \end{bmatrix} 1 \end{bmatrix} \\ \begin{bmatrix} \text{PRED} & \text{‘HELP’} \\ \text{OBJ} & [\ \ ]1 \end{bmatrix} \end{array} \right\}$$

Determinate adjectives with indeterminate nouns

a.

*Er	findet	und	hilft	alten	Papageien.
he	finds	and	helps	old	parrots
	OBJ=ACC		OBJ=DAT	DAT	NOM/ACC
					DAT/GEN

‘He finds and helps old parrots.’

b. Ill-formed f-structure:

$$\begin{bmatrix} \text{PRED} & \text{‘PARROTS’} \\ \text{CASE} & \begin{bmatrix} \text{NOM} & - \\ \text{ACC} & +/- \\ \text{GEN} & - \\ \text{DAT} & + \end{bmatrix} \end{bmatrix}$$

Indeterminate adjectives with indeterminate noun

rosa: [no case restrictions]

a.	Er	findet	und	hilft	rosa	Papageien.
	he	finds	and	helps	pink	parrots
		OBJ=ACC		OBJ=DAT		NOM/ACC
						DAT/GEN

‘He finds and helps pink parrots.’

b.

PRED	‘PARROTS’	
CASE	NOM	
	ACC	+
	GEN	
	DAT	+

Indeterminacy Grammar Engineering Implementation

- One exemplar of each type
- Basic constituent- and functional-structure rules
- Tested whether analysis would work
  - all grammatical sentences accepted
  - all ungrammatical sentences rejected

Linguistic Issues Highlighted by Grammar Engineering

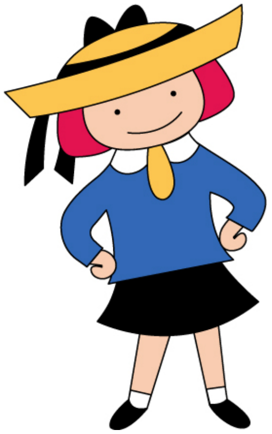
- Discover interesting facts from:
  - implementing each detail
  - working on corpora
    - interaction of phenomena
    - complexity of phenomena

- Caveat: things not working could be due to implementation limitations

Copulas, Adjectives and Subjects: Dalrymple, Dyvik, King 2004

- Second argument of copulars: subject or no?
- Predicate adjective agreement: simple with subjects

Elle est petite.
she.F.Sg is small.F.Sg
'She is small.'



Subject analysis: adjective and copula

$$\left[\begin{array}{ll} \text{PRED} & \text{`be<XCOMP>SUBJ'} \\ \text{SUBJ} & \left[\begin{array}{ll} \text{PRED} & \text{`she'} \\ \text{NUM} & \text{sg} \\ \text{GEND} & \text{fem} \end{array}\right] 1 \\ \text{XCOMP} & \left[\begin{array}{ll} \text{PRED} & \text{`small<SUBJ>'} \\ \text{SUBJ} & [\ ] 1 \end{array}\right] \end{array}\right]$$

petite (↑ PRED) = ‘small<SUBJ>’
(↑ SUBJ NUM) =c sg
(↑ SUBJ GEND) =c fem

est (↑ PRED) = ‘be<XCOMP>SUBJ′
(↑ SUBJ) = (↑ XCOMP SUBJ)

Non-subject Analysis: adjective and copula

Non-subject analysis with closed copular complement:

PRED	‘be<SUBJ,PREDLINK>’		
SUBJ	PRED	‘she’	
	NUM	sg	
	GEND	fem	
PREDLINK	PRED	‘small’	

petite (↑ PRED) = ‘small’
((PREDLINK ↑) SUBJ NUM) =c sg
((PREDLINK ↑) SUBJ GEND) =c fem

Pro: Raising adjectives

- It is likely/bound/certain to rain.
- They are eager/foolish/loathe to leave.

- Open complement with subject

$$\begin{bmatrix} \text{PRED} & \text{`be}<\text{XCOMP}>\text{SUBJ'} \\ \text{SUBJ} & \begin{bmatrix}\text{PRON-FORM} & \text{it}\end{bmatrix}1 \\ \text{XCOMP} & \begin{bmatrix} \text{PRED} & \text{`likely}<\text{XCOMP}>\text{SUBJ'} \\ \text{SUBJ} & [\ ]1 \\ \text{XCOMP} & \begin{bmatrix}\text{PRED} & \text{`rain}<>\text{SUBJ'} \\ \text{SUBJ} & [\ ]1\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

Con: If the post-copular has a subject

- The problem is that they appear / their appearing / for them to appear.

Open Complement

PRED	‘be<XCOMP>SUBJ’			
SUBJ	PRED	‘problem’		
XCOMP	PRED	‘appear<SUBJ>’		
	SUBJ	PRED	*‘they/problem’	

Closed Complement

PRED	‘be<PREDLINK>SUBJ’			
SUBJ	PRED	‘problem’		
PREDLINK	PRED	‘appear<SUBJ>’		
	SUBJ	PRED	‘they’	

What did we learn?

- Open complement good for:
  - Agreeing predicate adjectives
  - Raising adjectives
- Open complement bad for:
  - Predicate nouns/adjectives with a subject
- No elegant overall solution

Implementation Guiding Analyses

- Complex Predicates via Restriction:
  - Butt, King, Maxwell 2009
- Complex predicates heavily studied in LFG
- ParGram implementation: restriction

nAdiyah	nE	kahAnI	yAd	k-I
Nadya.F.Sg	Erg	story.F.Sg.Nom	memory.F.Sg.Nom	do-Perf.F.Sg

‘Nadya remembered a/the story.’

a. $(\uparrow \textsc{pred}) = \text{'yAd/memory}\langle\textsc{obj}\rangle\text{'} \mid (\uparrow \textsc{pred}) = \text{'yAd/memory'}$

b. $(\uparrow \textsc{pred}) = \text{'kar/do}\langle\ \textsc{subj}\ \textsc{obj}\ \rangle\text{'}$

c. Standard LFG PRED: $(\uparrow \textsc{pred}) = \text{'yAd/memory-kar}\langle\textsc{subj},\textsc{obj}\rangle\text{'}$

Restriction Operator (Kaplan & Wedekind 1993)

- Restrict out features of f-structure
- New f-structure is identical except for restricted features
- Allows dynamic composition of predicates

Original f-structure

$$\begin{bmatrix} \text{PRED} & \text{'Nadya'} \\ \text{NUM} & \text{sg} \\ \text{PERS} & 3 \\ \text{CASE} & \text{ERG} \end{bmatrix}$$

F-structure with case restricted out

$$\begin{bmatrix} \text{PRED} & \text{'Nadya'} \\ \text{NUM} & \text{sg} \\ \text{PERS} & 3 \end{bmatrix}$$

Lexical entries and f-structure for example

a. Standard LFG PRED: (↑ PRED) = 'memory-do<SUBJ,OBJ>'

b. Proposed restricted PRED: (↑ PRED) = 'do<SUBJ,'memory<OBJ>'>'

PRED	'do<SUBJ,'memory<OBJ>'>'
SUBJ	[PRED 'Nadya']
OBJ	[PRED 'story']

How it works

- F-structure is formed by dynamic composition of the subcategorization frames from *do* and *memory*
- Restriction operator invoked on annotated c-structure rules

$$\begin{array}{lcc} \mathbf{Vcp} \rightarrow & \mathbf{N} & \mathbf{Vlight} \\ & \downarrow\backslash\mathrm{PRED}{=}\uparrow\backslash\mathrm{PRED} & \uparrow{=}\downarrow \\ & (\uparrow \mathrm{PRED\ ARG2}){=}(\downarrow \mathrm{PRED}) & \end{array}$$

Lexical entries

$$\text{Vcp} \rightarrow \quad \underset{\substack{\downarrow\backslash\text{PRED}=\uparrow\backslash\text{PRED} \\ (\uparrow \text{PRED ARG2})=(\downarrow \text{PRED})}}{\text{N}} \quad \underset{\uparrow=\downarrow}{\text{Vlight}}$$

- *do* contributes subject and needs a further predicate

$$(\uparrow \text{PRED}) = \text{'do} < \text{SUBJ } \%\text{Pred} >\text{'}$$

- *memory* contributes an object

$$(\uparrow \text{PRED}) = \text{'memory} < \text{OBJ} > \text{'}$$

- combined as complex predicate

$$(\uparrow \text{PRED}) = \text{'do} < \text{SUBJ}, \text{'memory} < \text{OBJ} > \text{'} > \text{'}$$

N

$\downarrow \backslash \text{PRED} = \uparrow \backslash \text{PRED}$

$(\uparrow \text{PRED ARG2}) = (\downarrow \text{PRED})$

PRED	'memory<OBJ>'
NUM	sg
GEND	fem
OBJ	[ ]

Vlight

$\uparrow = \downarrow$

PRED	'do<SUBJ, %Pred>'
PERF	+
NUM	sg
GEND	fem
SUBJ	[ ]

Vcp

PRED	'do<SUBJ, 'memory<OBJ>'>'
PERF	+
NUM	sg
GEND	fem
SUBJ	[ ]
OBJ	[ ]

Complex Predicates via Restriction

- Creates the predicates and functional-structures we want
- Accounts for stacked complex predicates
- Allows a ParGram Urdu analysis compatible with output of other grammars

- Theory:
  - Should this be used theoretically?

- Grammar Engineering:
  - Issues with interaction with lexical rules
  - Hassle to maintain

Insights into Architecture Issues

- Passive-Causative Interactions
- Urdu and Turkish ParGram grammars
- Passive done by lexical rule
- Causative done by restriction
- Interaction gave exactly the wrong results

Urdu Causatives

- Morphological
- Causer subject
- Underlying subject of verb becomes oblique

a. yassIn=nE kHAnA kHa-yA
Yassin=Erg food.M.Sg.Nom eat-Perf.M.Sg
'Yassin ate food.'

b. nAdyA=nE yassIn=kO kHAnA kHil-A-yA
Nadya=Erg Yassin=Dat food.M.Sg.Nom eat-Caus-Perf.M.Sg
'Nadya had Yassin eat (fed Yassin).'

Causative Analysis

eat.Causative

kHilvAyA ⇔ kHA +Verb +Cause +Perf +Masc +Sg

V →	V_BASE	CAUSE
	↓\PRED\SUBJ=↑\PRED\SUBJ	↑=↓
	(↓SUBJ)= (↑OBJ-GO)	
	(↑ PRED ARG2)=(↓ PRED)	

a. +Cause: (↑ PRED) = 'cause< SUBJ %Pred >'

b. eat (↑ PRED) = 'eat< SUBJ OBJ >'

V →	V_BASE	CAUSE
	↓\PRED\SUBJ=↑\PRED\SUBJ	↑=↓
	(↓SUBJ)= (↑OBJ-GO)	
	(↑ PRED ARG2)=(↓ PRED)	

V-S_BASE

PRED	'eat<SUBJ,OBJ>'
SUBJ	[ ]1
OBJ	[ ]2

CAUS_BASE

PRED	'cause< SUBJ %Pred >'
PERF	+
GEND	masc
NUM	3
SUBJ	[ ]3

V

PRED	'cause<SUBJ,'eat<OBJ-GO,OBJ>'>
PERF	+
GEND	masc
NUM	3
SUBJ	[ ]3
OBJ-GO	[ ]1
OBJ	[ ]2

Passives

- Formed by combining the verb *jA* ‘go’ with the perfect form of the main verb
- Object of the base verb is realized as a subject
- Subject/agent of the base verb is realized as an adjunct

a.

yassIn=nE	kHAnA	kHa-yA
Yassin=Erg	food.M.Sg.Nom	eat-Perf.M.Sg

‘Yassin ate food.’

b.

kHAnA	yassIn=sE	kHa-yA	ga-yA
food.M.Sg.Nom	Yassin=Inst	eat-Perf.M.Sg	go-Perf.M.Sg

‘The food was eaten by Yassin.’

Passive with Causative

- Subject of the causative becomes an adjunct (*sE* marked)

a. nAdyA=**nE** yassIn=kO kHAnA kHil-**A**-yA
Nadya=Erg Yassin=Dat food.M.Sg.Nom eat-**Caus**-Perf.M.Sg
'Nadya had Yassin eat (fed Yassin).'

b. yassIn=kO nAdyA=**sE** kHAnA kHil-**A**-yA ga-yA
Yassin=Dat Nadya=Inst food.M.Sg.Nom eat-**Caus**-Perf.M.Sg go-Perf.M.Sg
'The food was fed to Yassin by/through Nadya.'

- Subject of original main verb cannot be passivized

*nAdyA=**nE** yassIn=sE kHAnA kHil-**A**-yA ga-yA
Nadya=Erg Yassin=Inst food.M.Sg.Nom eat-**Caus**-Perf.M.Sg go-Perf.M.Sg
'Nadya made the food be eaten by/through Yassin.'

- Grammar had reverse pattern

Architecture Issue with Implementation

- Passive via lexical rules when compiling lexicon

a. $\text{TRANS(P)} = @(\text{PASSIVE}\ (\uparrow\text{PRED})='\text{P}<(\uparrow\text{SUBJ})\ (\uparrow\text{OBJ})>')$

b. $\text{PASSIVE(P)} = \quad (\uparrow\text{SUBJ}) \longrightarrow \text{NULL}$

$\qquad\qquad\qquad\quad (\uparrow\text{OBJ}) \longrightarrow (\uparrow\text{SUBJ})$

- Causative via restriction applies afterwards

Ungrammatical derivation of passive+causative:

a. Original Predicate: $(\uparrow\text{PRED})='\text{eat}<(\uparrow\text{SUBJ})\ (\uparrow\text{OBJ})>'$

b. Lexical Rule Passive: $(\uparrow\text{PRED})='\text{eat}<\text{NULL}\ (\uparrow\text{SUBJ})>'$

c. Restriction Causative:
$(\uparrow\text{PRED}) = '\text{cause}<\text{SUBJ},'\text{eat}<\text{NULL},\text{OBL-GO}>'>$

Architecture continued

- Have passive also apply via restriction

Grammatical derivation of passive+causative:

a. Original Predicate(↑PRED)='eat<(↑SUBJ) (↑OBJ)>'

b. Restriction Causative: (↑PRED) = 'cause<SUBJ,'eat<OBL-GO,OBJ>'>

c. Restriction Passive: (↑PRED) = 'cause<NULL,'eat<OBL-GO,SUBJ>'>

- Works but not satisfactory
- Argument alternations in argument-structure
  – but not implemented
- Traditional lexical rules: wrong order of application when causative is morphological but passive syntactic