Update on Soar-based language processing

Deryle Lonsdale
(and the rest of the BYU NL-Soar Research Group)
BYU Linguistics
lonz@byu.edu

NL-Soar

NL-Soar developments

- Discourse/robotic dialogue
  - ICSLP DoD
  - BRIMS poster
- Running on Soar 8.5.2
  - Some NLG chunking issues remain
- Having trouble getting to 8.6.1
  - Fresh start with 8.6.2…

LG-Soar

Overview

- Link-Grammar Soar
  - Implements syntactic, shallow semantic processing
  - Used for information extraction
  - Components
    - Soar architecture
    - Link Grammar parser
    - Discourse Representation Theory for discourse modeling
  - Discussed in Soar 21, Soar 22

LG-Soar developments

- Predicate extraction in biomedical texts domain (www.clinicaltrials.gov)
- NLDB 2006
- Two stages:
  - Identify and extract predicate logic forms from medical clinical trial (in)eligibility criteria
  - Match up the information with other data, e.g. patients' medical records
- Result: tool for helping match patients with clinical trials

XNL-Soar

Our goals

- Integrate the MP into a cognitive modeling engine
- Explore language/task-integrations using the MP
- Test cross-linguistic implementation possibilities with the MP
- Ultimately, determine whether the MP supports incremental, operator-based processing

Our approach

- Map the syntactic parsing onto operators
- Integrate external knowledge sources
- Strengths:
  - We have already done this for NL-Soar
  - The MP has an operator-like feel to it
- Weaknesses:
  - MP lit sketchy on incremental parsing
  - External knowledge sources incommensurable
  - Our scant knowledge of human performance data

Derivational principles

- Minimalist Principles (Chomsky 1995)
  - Merge
  - Move
- Hierarchy of Projections (Adger 2003)
  - Nominal: D > (Poss) > n > N
  - Clausal: C > T > (Neg) > (Perf) > (Prog) > (Pass) > v > V
- Governed by features
  - Strong and weak features
- NP, VP symmetry including shells

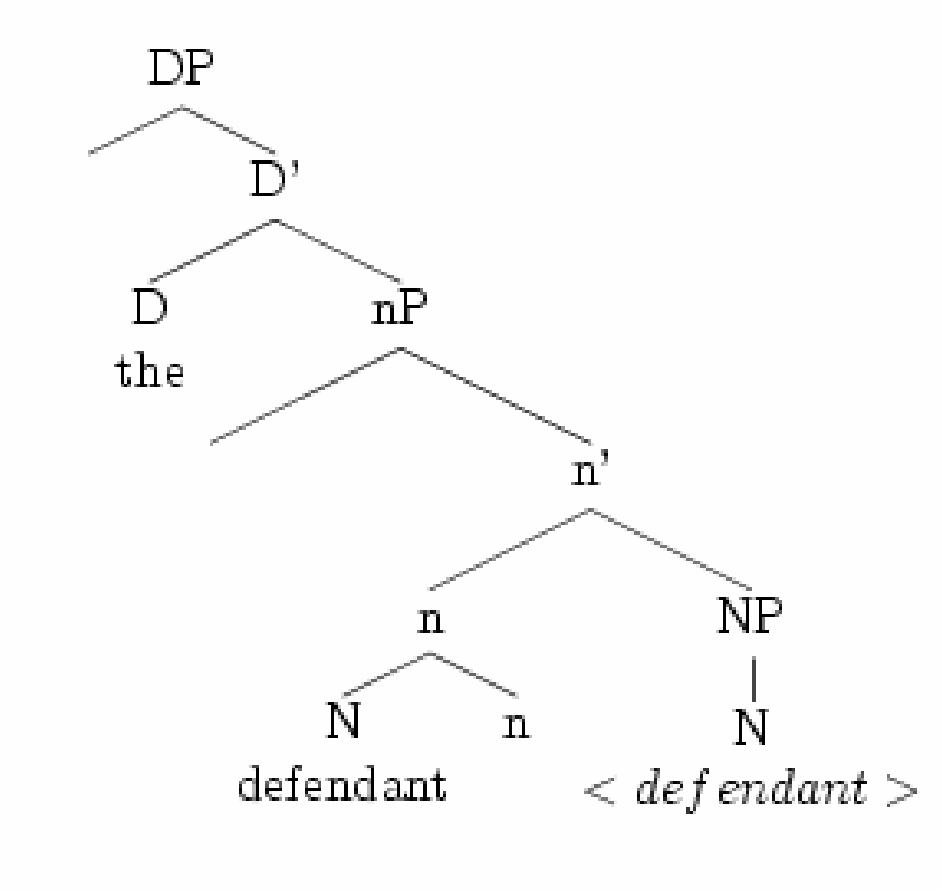
Operators and operator types

- XNL-Soar op types and functions:
  - Lexical access: retrieve & store lexically-related information
  - Merge: construct syntax via MP-specified merge operations
  - Movehead: perform head-to-head movement (via adjunction)
  - HoP: consult hierarchy of projections, return next possible target level
  - Project: create bare-structure maximal projection from lexical item

Nominal projections

- DP's
- NP Shells
- Feature checking
  - HoP for nominals
  - Projection of N to NP
    - Bare phrase structure for lexical heads
- Operators: Project, Merge1, Merge2, Check-Root, and HoP

Projection of a DP



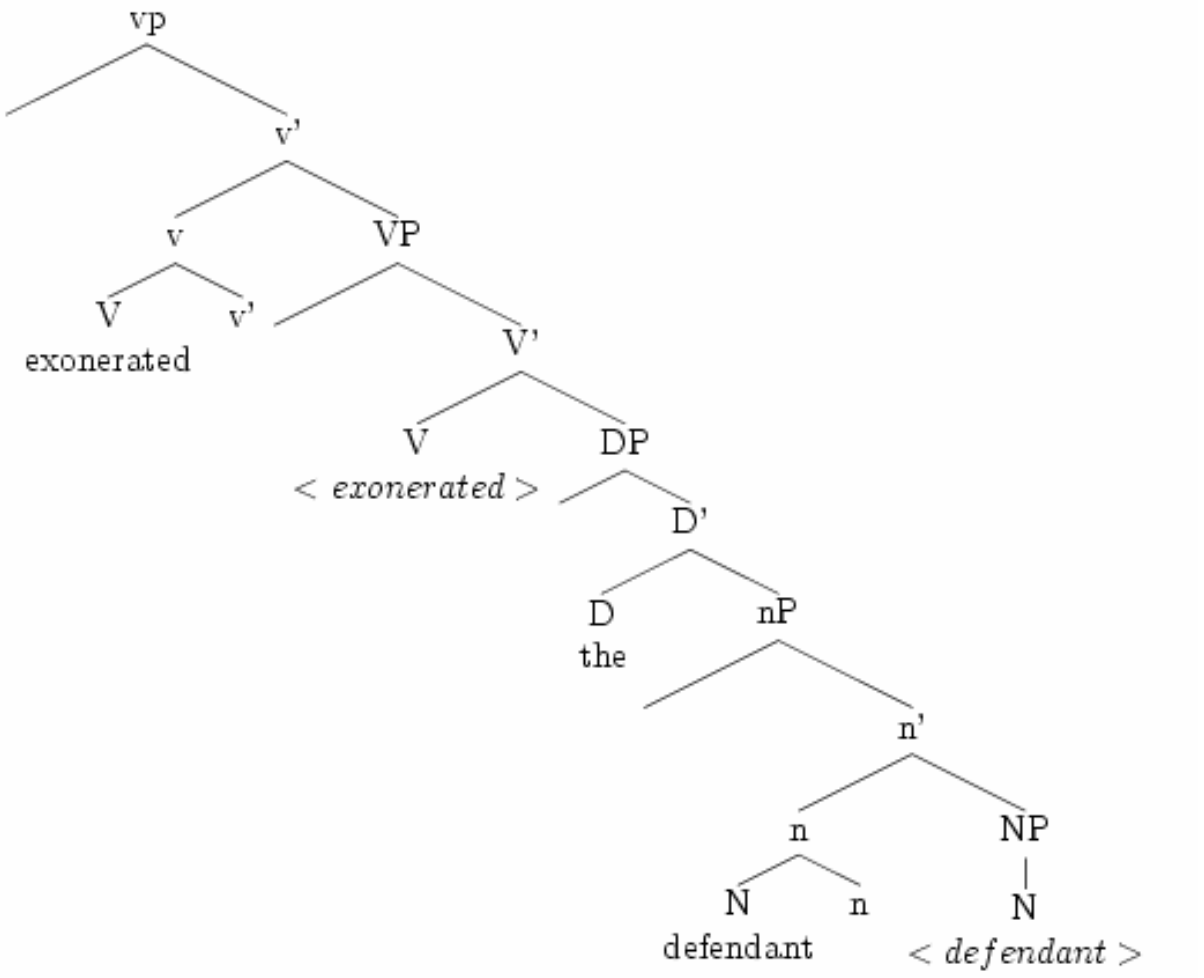
Verbal projections

- VP Shells
- Theta roles & the LCS lexicon
- ucat grids
- The HoP for the clause
- Operators: Merge1, Merge2, Check-root, HoP.

Merge

- 1st Merge
  - Complement merge based on ucat features
- 2nd Merge
  - Specifier merge based on a second ucat feature

Projection of a VP: HoP & Merge



Move

- Governed by strong ucat features
- Two types of movement
  - Head-head movement
    - Creates new structure at the head level
  - Phrasal movement
- Operators: Copy, Hadjunction

External knowledge sources

- WordNet 2.0 (wordnet.princeton.edu)
  - Lexical semantics: part-of-speech, word senses, subcategorization
  - Inflectional and derivational morphology
- English LCS lexicon
  (www.umiacs.umd.edu/~bonnie/verbs-English.lcs)
  - Thematic information: θ-grids, θ-roles
  - Used to derive uninterpretable features
  - Triggers syntactic construction
  - Aligned with WordNet information

English LCS lexicon data

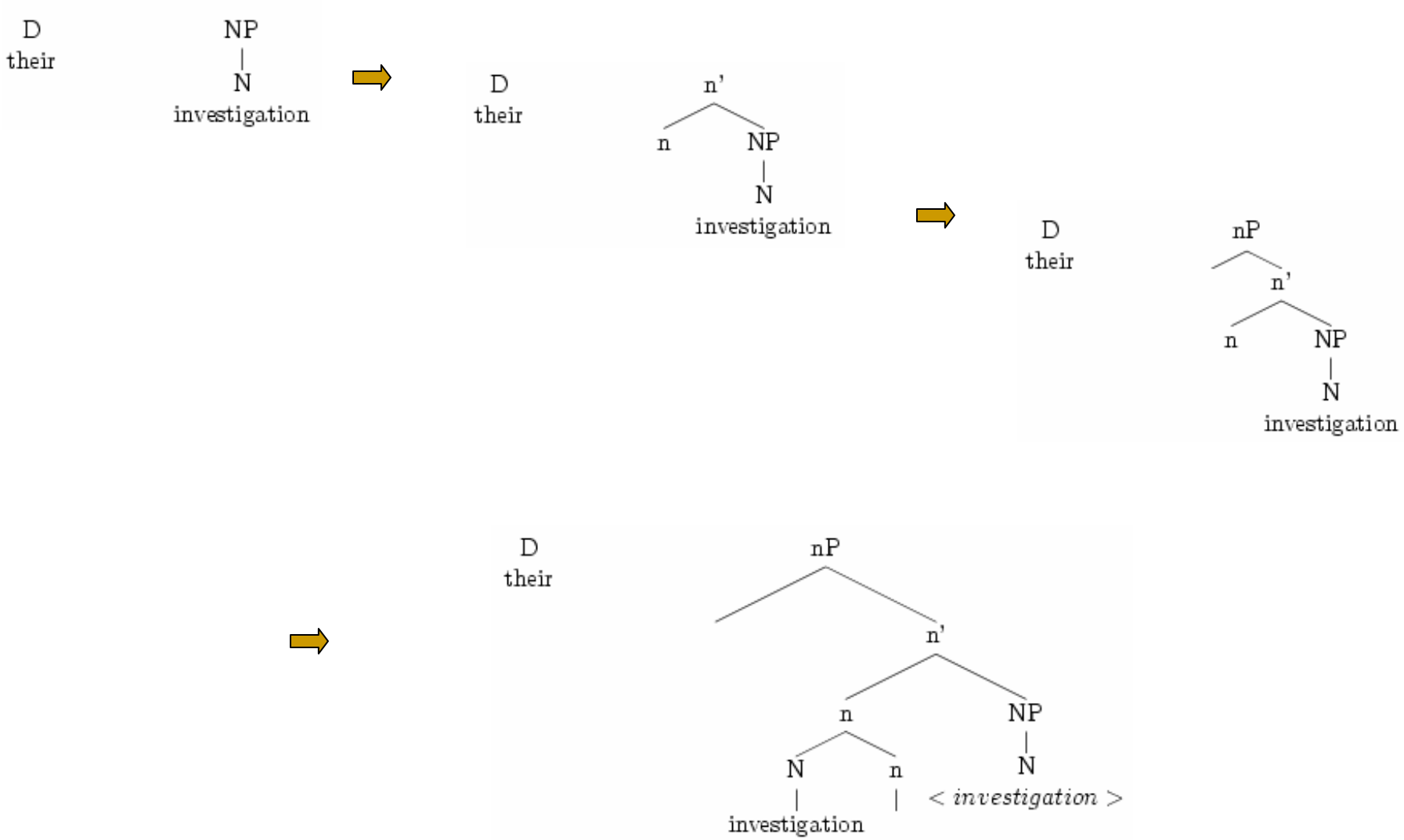
```
10.6.a#1#_ag_th,mod-
poss(of)#exonerate#exonerate#exonerate#exonerate+ed#
(2.0,00874318_exonerate%2:32:00::)

10.6.a Verbs of Possessional Deprivation: Cheat Verbs/-of
WORDS (absolve acquit balk bereave bilk bleed burgle cheat
cleanse con cull cure defraud denude deplete depopulate
deprive despoil disabuse disarm disencumber dispossess divest
drain ease exonerate fleece free gull milk mulct pardon
plunder purge purify ransack relieve render rid rifle rob sap
strip swindle unburden void wean)

((1 "_ag_th,mod-poss()")
 (1 "_ag_th,mod-poss(from)")
 (1 "_ag_th,mod-poss(of)"))

"He !!+ed the people (of their rights); He !!+ed him of his
sins"
```

Similar Work

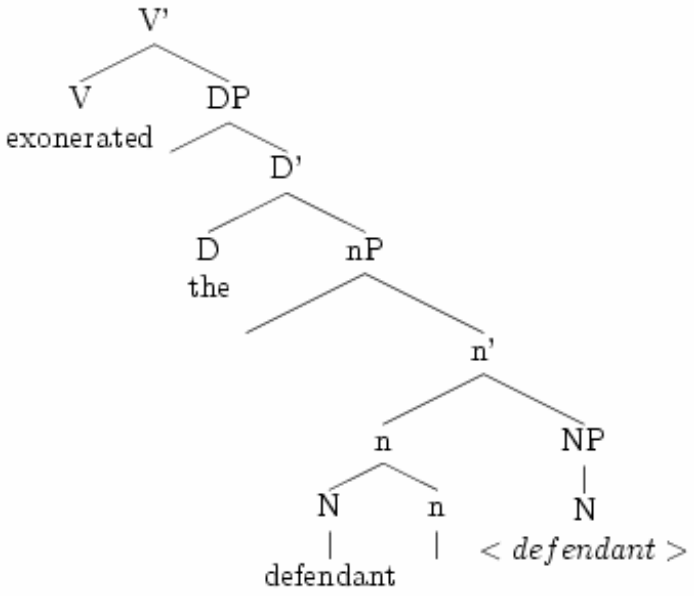
- Incremental parsing in general (Phillips '03)
- Other linguistic theories for incremental parsing
  - GB (Kolb 1991)
  - Dependency grammar (Milward 1994, Ait-Mokhtar et al. 2002)
  - Categorial Grammar (Izuo 2004)
- Finite-state methods (Ait-Mokhtar & Chanod 1997)

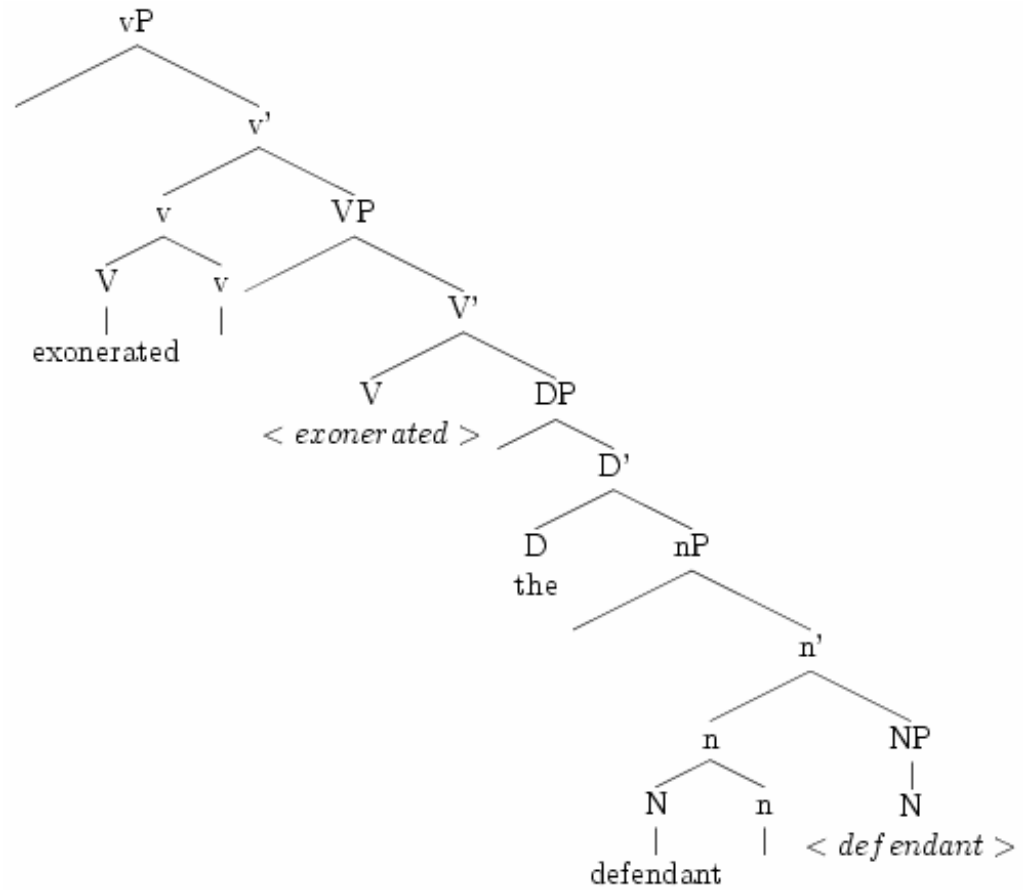
Similar Work

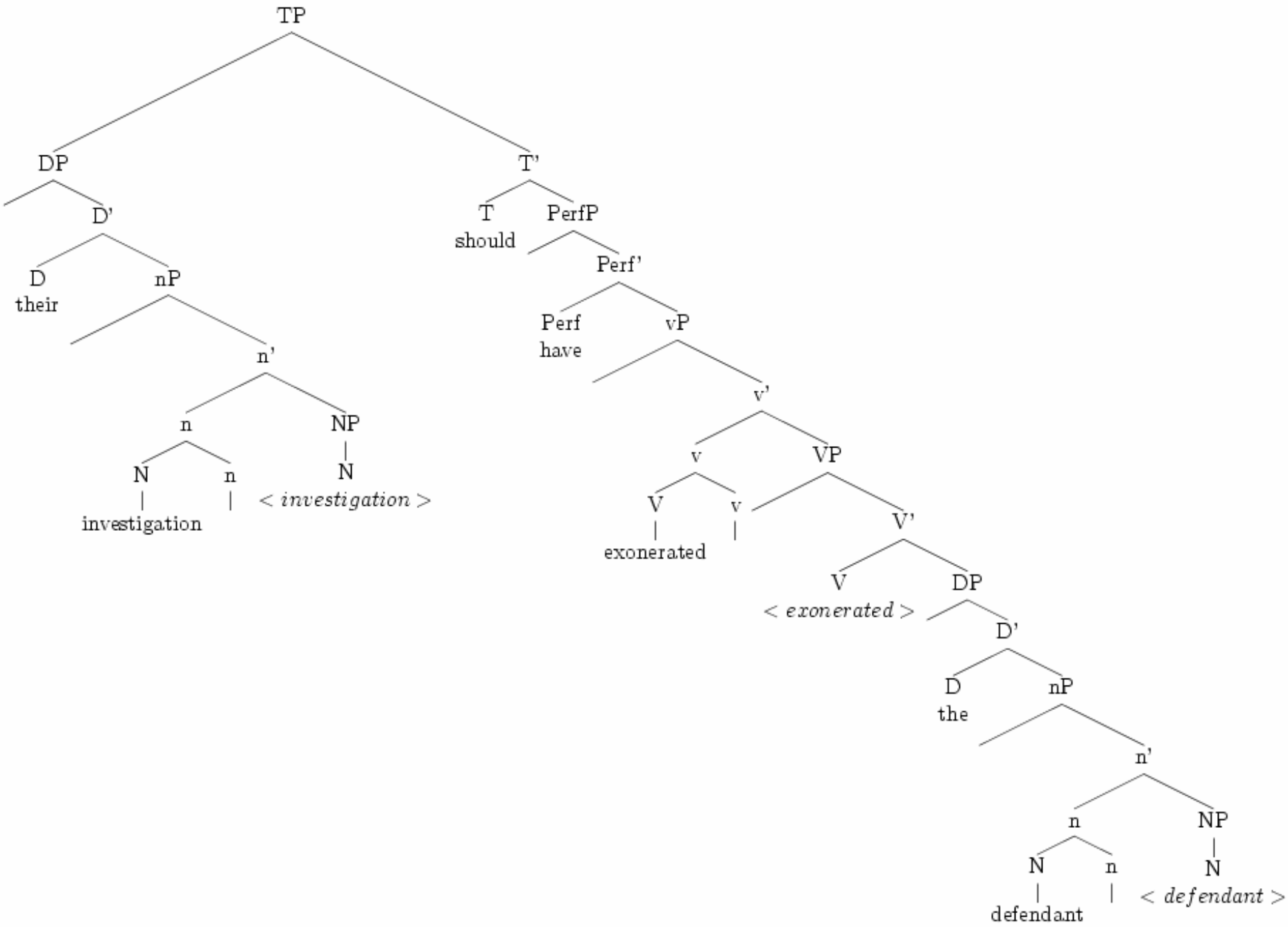
- Minimalist parsing in other frameworks (Stabler 1997, Harkema 2001)
- Thematic information and parsing (Schlesewsky & Bornkessel 2004)
- Crosslinguistic considerations in incremental parsing (Schneider 2000)
- Human studies on ambiguity, reanalysis
  - Eye tracking (Kamide, Altmann, & Haywood '03)
  - ERP (Bornkessel, Schlesewsky, & Friederici '03)


D
their
NP
N
investigation
D
their
n'
n
NP
N
investigation
D
their
nP
n'
n
NP
N
investigation
D
their
nP
n'
n
NP
N
n
N
investigation
< investigation >

T	Perf	V	V	D	N
should	have	have	exonerated	the	defendant




vP
v'
v
VP
V
v
exonerated
V'
V
DP
< exonerated >
D'
D
nP
the
n'
n
NP
N
n
N
defendant
< defendant >


TP
DP
D'
D
their
nP
n'
n
N
investigation
n
NP
N
< investigation >
T'
T
should
PerfP
Perf'
Perf
have
vP
v'
v
V
exonerated
v
VP
V'
V
< exonerated >
DP
D'
D
the
nP
n'
n
N
defendant
n
NP
N
< defendant >

Building a full sentence

```
agent> init-soar
agent> r
     0: ==>S: S1

     2:    O: O1 (getword)
Input a word: exonerated

     4:    O: O2 (getword)
Input a word: defendants
     5:    O: O3 (project) --> NP

     7:    O: O5 (hop)
     8:    O: O8 (merge1)
    10:    O: O9 (merge2) --> nP

    12:    O: O14 (hadjoin) --> move N

    13:    O: O13 (hop)
    14:    O: O16 (merge1)
    16:    O: O17 (merge2) --> DP

    18:    O: O21 (merge1)
    20:    O: O23 (merge2) --> VP

    22:    O: O27 (hop)
    23:    O: O30 (merge1)
    25:    O: O31 (merge2) --> vP

    27:    O: O36 (hadjoin) --> move V

    28:    O: O35 (hop)
    29:    O: O38 (merge1)
    31:    O: O39 (merge2) --> TP
```

Current status

- POC for fundamental syntactic structures
- Basic sentence types (transitives, unergatives, unaccusatives)
- All functional and lexical projections in syntactic structure
- Most feature percolation, feature checking
- Current system: about 60 productions (cf. 3500 NL-Soar)
- External knowledge sources: interfaced via 1000+ lines of Tcl/Perl

Issue

- Find a balance between generation and parsing
  - Most MP descriptions are generative, not recognitional in focus
  - Is it advisable and well motivated to "undo" or "reverse" movements?
  - If not, is generate-and-test the right mechanism for parsing input?
  - What are the implications for learning and bootstrapping language capabilities (e.g. parsing in the service of generation)?

Future functionality

- XP adjunction
- Assigners/receivers set?
- Wider coverage of complex constructions
  - Ditransitives, resultatives, causatives, etc.
- More semantics/deeper semantics.
  - Quantifier raising
  - Scopal relationships
  - C-command and other interpretive mechanisms
  - More detailed LCS structures
- Web-based Minimalist Parser grapher

Future applications: cf. NL-Soar

- Explore human parser robustness, processing of ambiguity, learning
- Integrate syntax/semantics into discourse/conversation component
- Bootstrapping: parsing and generation
- Develop human-agent & agent-agent comm
- Parameterize XNL-Soar for processing of other languages besides English
- Model cognition in reading
- Model real-time language/task integrations

Conclusion

- Coals
  - Performance?
  - MP not fully explored
  - Reconciling disparate lexical resources is non-trivial (WordNet + LCS)
  - Redoing learning in Soar8
  - Graphing is more complicated
- Nuggets
  - Better coverage (Engl. & crosslinguistically)
  - New start in Soar8
  - State-of-the-art syntax
  - Interest: CUNY Sentence Processing, CogSci