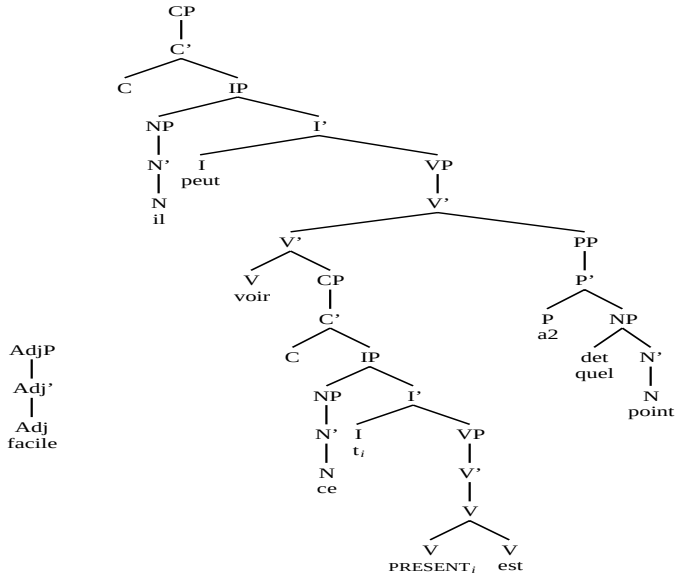
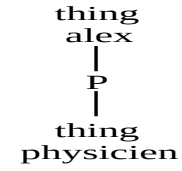
NL-Soar Generation Update

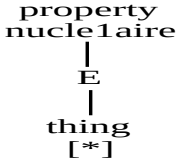
Deryle Lonsdale
Computational Linguistics
Carnegie Mellon
lonz@cs.cmu.edu
Wean 5103
(412) 268-7571

NLC review

- NLC: NL-Comprehension
  maps input sentences to structure
  1. syntax: GB-based u(tterance)-model
     via u-constructors
  2. semantics: LCS-based s(emantic)-model
     via s-constructors





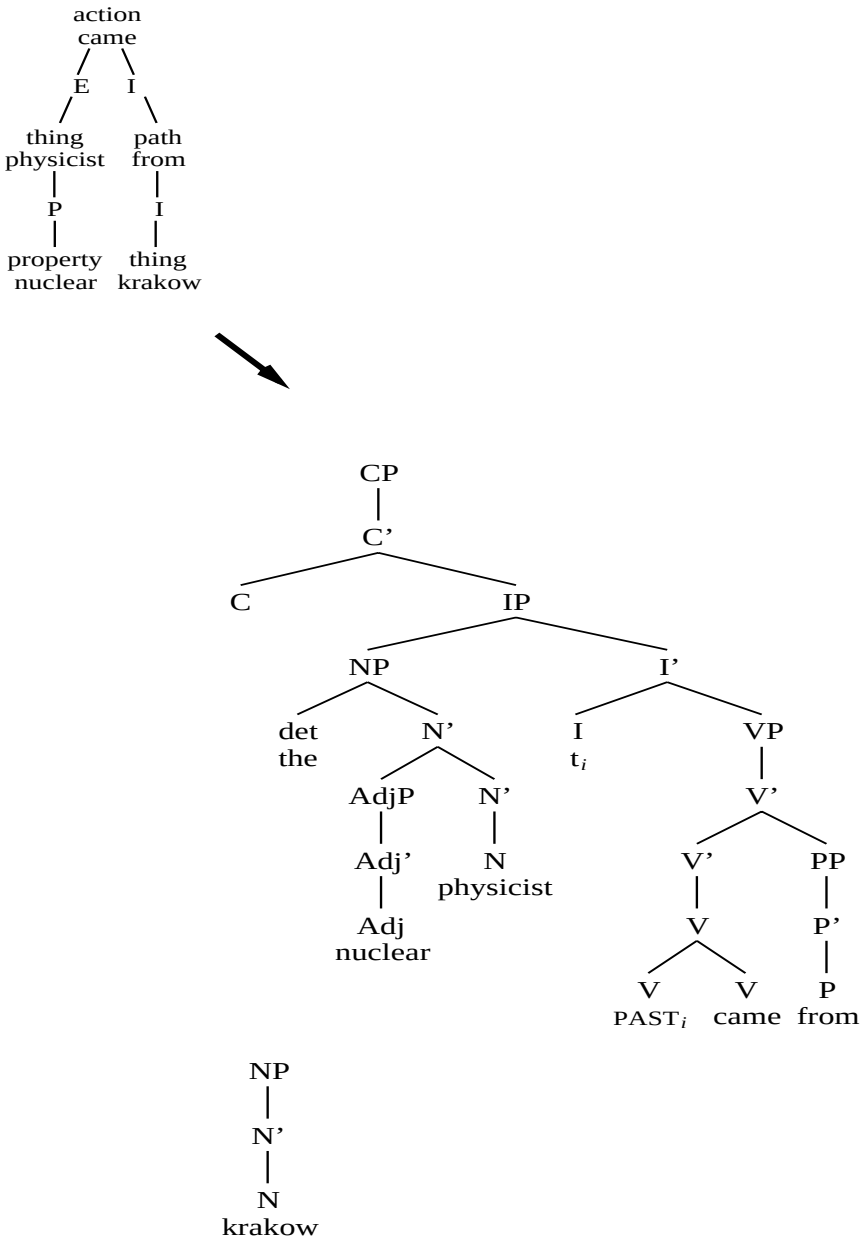


NLC progress

- Syntactic coverage
  - compound tenses
  - passives
  - relative clauses
  - infinitival clauses
- Semantic coverage
  - relative clauses
- Syntax/semantics correspondence
  - enforced precedence: syntax $>$ semantics
  - categorial mappings: nouns = things, adjective = property
  - s-constructor constraints based on syntactic model
  - syntactic snips initiate semantic snips
  - semantic-based proposals of u-model snips

NLG processing

- NLG: NL-Generation
  maps structure to output sentences



NLG characteristics

- Integration with NLC
  1. same lexical access operator, lexicon
  2. same learn-language operator, learning
  3. same architecture: A/R set, i/o
  4. same u-constructor operators
- Integration with Discourse
  1. turn-taking
  2. reference

Brief overview

Tactical generation is initiated when:

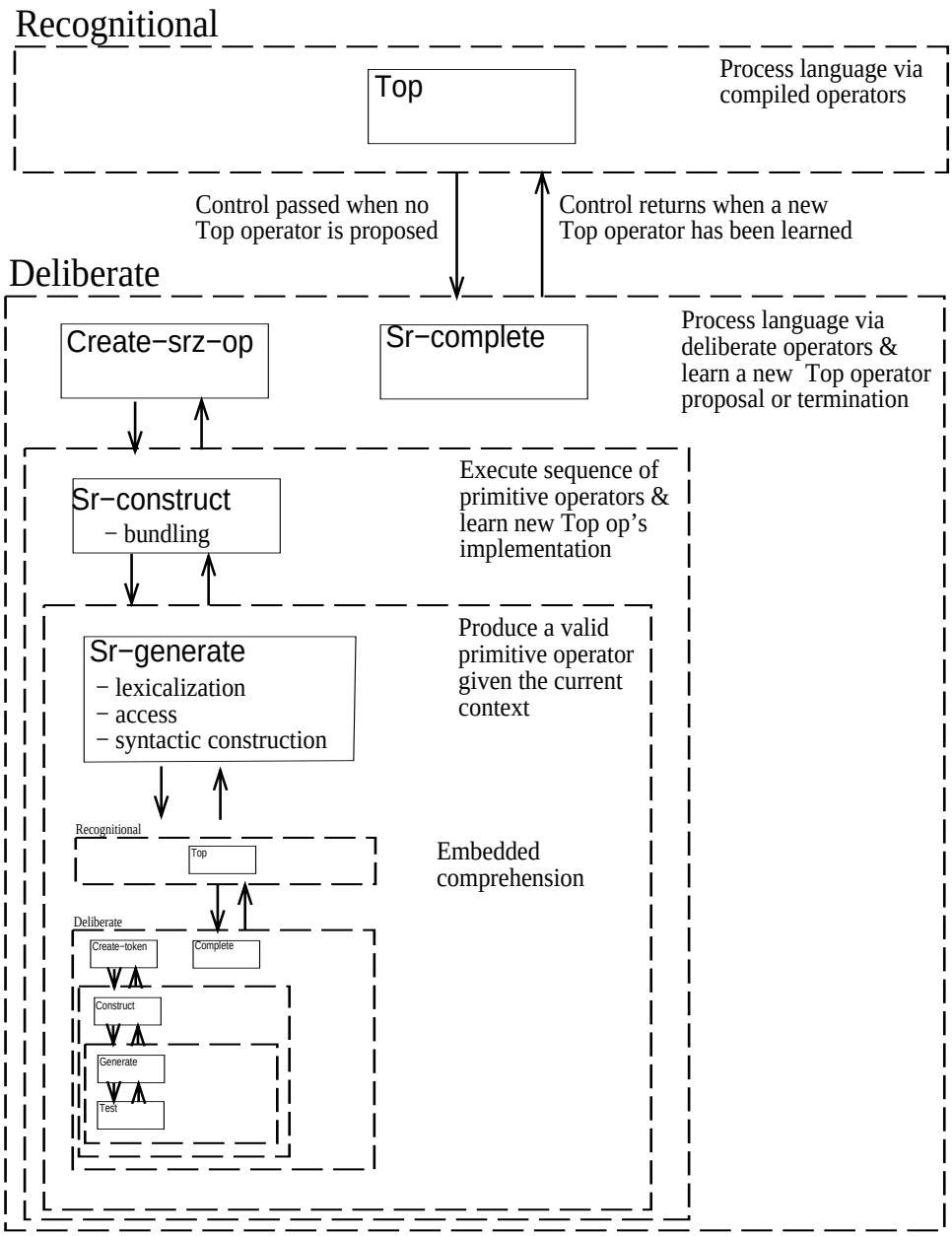
- discourse move prompts agent to communicate
- connected LCS exists

Maps s-model $\longmapsto$ u-model $\longmapsto$ output word sequence

- Traverses s-model concepts and annotations
- Selection operators choose, prioritize which to process when
- Recognitional processing via s-realize operators (1 per concept)
- S-realize operators learned via learn-language ops
  lexicalize concept +
  perform lexical access +
  build u-constructor(s)
  (bundle these together)
- Synthesizes words/phrases from concepts/features
- Employs embedded comprehension in generate-and-test framework

NLG (and NLC)

- Comprehension in the service of generation



Sample trace

```
 545:       O: O359 (select-semobj)
***selected S101
  546:       O: O550 (learn-language)
***realizing: S101
  549:          O: O556 (s-realize64)
  553:                O: O559 sr-generate-op(realize-lexically)
  556:                   O: O563 (pick-word)
  560:                      O: O569 check-candidate: \"patient\"
  561:                      O: O570 (imagine-word)
  564:                         O: O573 (access word: 'patient' spkr: user)
  567:                         O: O576 (u-constructor24)
  576:             O: C680 (realize-lexically)
  577:             O: O580 (access word: 'patient')
  580:             O: O576 (u-constructor24)
  586:       O: O556 (s-realize64)
  589:       O: O590 (say-word)
saying he on frequency radio-100
  590:       O: O597 (say-word)
saying prepared on frequency radio-100
  591:       O: O602 (say-word)
saying the on frequency radio-100
  592:       O: O606 (say-word)
saying next on frequency radio-100
  593:       O: O609 (say-word)
saying patient on frequency radio-100
```

Syntactic transfer

- Bootstrapping u-constructors is possible intermodally
  - NLG can leverage NLC-learned syntactic knowledge
  - NLC can leverage NLG-learned syntactic knowledge
- Processing asymmetries exist
  - NLC: *I have...* local, temporary ambiguity
  - NLG: *I have...* unambiguous (s-model disambiguates)
- Top-level tasking flag controls masking, infelicitous transfer

Recent work

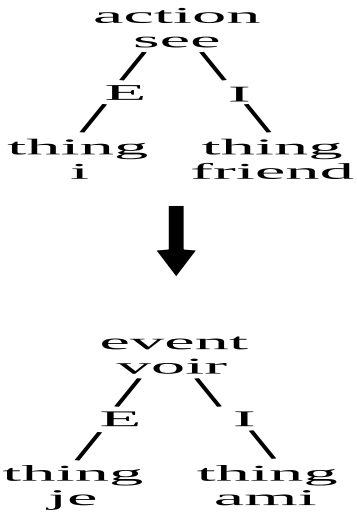
- Results
  - Use of LCS to drive generation
  - Coverage: determiners, prepositions, complementizers, dependent clauses, copulas, auxiliaries, modals
  - Remove operator, snips, empty operators
  - Integration: TacAir, discourse
  - Different languages
- Some data
  - Average 190 ec's to learn s-realize op, 20 ec's recognitionally
  - Average 36 conditions in s-realize op proposal chunk
  - Average 44 actions in s-realize op proposal chunk
  - Average 8.5 sec. cpu to learn s-realize op, 1 sec. recognitionally

Ongoing work

- Common problems
  - considerable chunking
  - chunk ordering problems
  - memory exhaustion due to deep goal stack and state copies
  - specificity of chunks
  - masking (intermodal and intra/inter-sentential)
- Interleaving with NLC
- Coverage extension
- Documentation

Mapping

Maps s-model to s-model



- Learned via m-constructors
  lexicalize concept +
  lexical access +
  build s-constructor
  (bundle these together)
- Translates concepts/features to concepts/features
- Useful for source-target language mappings
- Leverages semantic transfer (s-constructors) for target language
- Exploring deliberate/recognitional translation

Modeling simultaneous interpretation

- source language comprehension (NLC: English)
- translation (Mapping: English $\mapsto$ French)
- target language generation (NLG: French)
- concurrent use of multiple languages
- interleaved tasks (hearing, translating, speaking)
- matching time scale from recorded protocols
- task control, resource usage, expert/novice contrast