LKB-FOS Update 2025

John Carroll

University of Sussex, UK

DELPH-IN Summit, Amsterdam, July 2025

Outline

One LKB-FOS release since last year's summit, on 22 June

A few issues fixed

Improvements to efficiency and scalability

Extension to local ambiguity packing under generalisation

Issues Fixed

Fixed some issues

- edges that are packed inside others are now shown in LUI chart display
- more compact reporting of Lisp warnings when loading a grammar
- fixes to incorrect/unintuitive behaviour involving parameters `*first-only-p*`, `*unpacking-scoring-hook*`, `*feature-ordering*`, `*start-symbol*`, `*gen-start-symbol*`
- corrected documentation strings for a few parameters

Queries answered on Discourse site about: turning off packing, interface to Utool, Apple silicon version, running old unsupported Lisp code, building from source

Continued to clean up and modernise code

Efficiency and Scalability

Rectified unfortunate design decisions in database / cache modules

- type unification cache changed from open to closed hashing → faster access
- reworked lexicon database to address scaling problems
  - load factor reduced from 1 to 0.5 → no longer ends up as a linear search
  - cached entries evicted when a size threshold reached → growth limited

Improved memory management

- further reduced garbage collection overheads; temporary data discarded in a more effective and timely manner (currently only in `batch-parse-sentences` function, not [incr tsdb()])
- agenda-based processing often not friendly to CPU caches; refined task priority computation so consecutive tasks likely to use much of the same data

Reduced the cost of cyclic FS checking

- when applying a rule, the features in `*deleted-daughter-features*` (= ACE's `deleted-daughters`) are removed; if their values contain cyclic structure then the rule application should fail
  - these cyclic checks are now deferred from parse forest construction to unpacking; forest won't contain any (invalid) cyclic feature structures, but may represent invalid constituents
  - unpacking replays rule applications *with* cyclic checks, as previously
  - to restore old behaviour: `(setq *defer-dd-cyclic-check-p* nil)`

Ambiguity Packing in the Parser

Originally, local ambiguity packing for DELPH-IN grammars used FS *subsumption*: an edge may be ‘packed’ with another if FSes are in a subsumption relationship

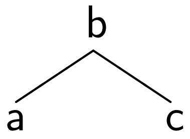
ACE and LKB-FOS (as of 2024) can also pack under a version of *generalisation* in which corresponding pairs of types may be generalised to the *more general type*

Added a further kind of generalisation, in which corresponding pairs of types may be generalised to their *least common supertype*

- input types and supertype must have the same appropriate features
- reentrancies are not generalised (they must subsume in one direction or other)
- generalising FS is created from the input FS that has the more general set of reentrancies, with types in it being replaced with supertypes as necessary

Enable by `(setq *generalising-p* :lcs)`

a := b
c := b



subsumption

$$\begin{tikzcd}
\begin{bmatrix} \mathsf{x} & \\ \mathsf{F} & \mathsf{a} \\ \mathsf{G} & \mathsf{b} \end{bmatrix} & \begin{bmatrix} \mathsf{x} & \\ \mathsf{F} & \boxed{1}\ \mathsf{a} \\ \mathsf{G} & \boxed{1} \end{bmatrix} & {} \arrow[from=1-1, to=1-3, bend right]
\end{tikzcd}$$

more general type (MGT)

$$\begin{tikzcd}
\begin{bmatrix} \mathsf{x} & \\ \mathsf{F} & \mathsf{a} \\ \mathsf{G} & \mathsf{b} \end{bmatrix} & \begin{bmatrix} \mathsf{x} & \\ \mathsf{F} & \boxed{1}\ \mathsf{b} \\ \mathsf{G} & \boxed{1} \end{bmatrix} & \begin{bmatrix} \mathsf{x} & \\ \mathsf{F} & \mathsf{b} \\ \mathsf{G} & \mathsf{b} \end{bmatrix} \arrow[from=1-1, to=1-3, bend right]
\end{tikzcd}$$

least common supertype (LCS)

$$\begin{tikzcd}
\begin{bmatrix} \mathsf{x} & \\ \mathsf{F} & \mathsf{a} \\ \mathsf{G} & \mathsf{b} \end{bmatrix} & \begin{bmatrix} \mathsf{x} & \\ \mathsf{F} & \boxed{1}\ \mathsf{c} \\ \mathsf{G} & \boxed{1} \end{bmatrix} & \begin{bmatrix} \mathsf{x} & \\ \mathsf{F} & \mathsf{b} \\ \mathsf{G} & \mathsf{b} \end{bmatrix} \arrow[from=1-1, to=1-3, bend right]
\end{tikzcd}$$

Limitations

- LCS generalisation needs a pre-subsumption quickcheck that uses the (stricter) test for MGT; otherwise get more edges and/or slow unpacking
- packing under MCT / LCS generalisation only implemented in the parser, and only tested with ERG and SRG
- generalisation does not solve issues with large numbers of edges or long parse times with SRG on problematic *Tibidabo Corpus* items, including *Ya está bien, hombre ... es que no puede ser.*

LCS generalisation still experimental; default for `*generalising-p*` is therefore `t` (= MGT)

Practical Results

Parsing *Rondane* with ERG 2023 on iMac i7 3.8GHz, no PoS tagging, computing top-ranked parse, resource limits giving ∼25 timeouts (*mm:ss*)

	LKB-FOS	*ACE*
Subsumption	10:48	19:31
Generalisation		
more general type	7:49	13:54
least common supertype	7:01	–

Parsing *The Cathedral and the Bazaar* under LCS generalisation, able to compute parse forests for all sentences apart from one of >100 tokens

Parsing *Tibidabo* with SRG, gains for generalisation less than those with ERG

Summary

Reasonable progress over the past year

- fixed a few issues
- improved efficiency and scalability
- added a novel, more radical option for ambiguity packing

To-do list includes

- output full-forest parse results
- implement unified grammar configuration file format
- add ‘grandparent’ features in selective unpacking
- change post-generation chart mapping to act on ‘full’ FS, not ‘edge’ FS