Using Emerson-Turing Types for Unbounded Dependencies in the ERG

Dan Flickinger

DELPH-IN Summit 2023

Galicia, Spain

26 June 2023

Motivation

- Improved grammar maintenance and clarity in appending lists
- Experimenting with new tools
- Expanded coverage for phrases containing multiple gaps

  A violin this well-crafted, even the most difficult sonatas will be easy to play ___ on ___.

  This is a puzzle that I don’t know how to solve ___ ___.

Non-local features in HPSG

- SLASH, REL, QUE

  Examples like this, we seem to find quite often.

  We delight in books the covers of which disguise their worth.

  Which city did you suggest that we try to meet in next year?

Non-local features in HPSG

- SLASH, REL, QUE

  Examples like this, we seem to find quite often.

  We delight in books the covers of which disguise their worth.

  Which city did you suggest that we try to meet in next year?

- Pollard and Sag 1994: The value of each non-local feature on the mother of a phrase is the append of the values of that feature on each of the daughters.

Non-local features in HPSG

- SLASH, REL, QUE

  Examples like this, we seem to find quite often.

  We delight in books the covers of which disguise their worth.

  Which city did you suggest that we try to meet in next year?

- Pollard and Sag 1994: The value of each non-local feature on the mother of a phrase is the append of the values of that feature on each of the daughters.
- Lexical threading: The value of each non-local feature on the mother of a phrase is the value of that feature on the head daughter.

  That professor is not easy to talk to.

Diff list append vs ET list append

```
basic_two_arg := basic_lex_synsem &
  [ LOCAL.ARG-S < [ NONLOC.SLASH [ LIST #smiddle,
                                   LAST #slast ] ],
                  [ NONLOC.SLASH [ LIST #sfirst,
                                   LAST #smiddle ] ] >,
    NONLOC.SLASH [ LIST #sfirst,
                   LAST #slast ] ].
```

Diff list append vs ET list append

```
basic_two_arg := basic_lex_synsem &
  [ LOCAL.ARG-S < [ NONLOC.SLASH [ LIST #smiddle,
                                   LAST #slast ] ],
                  [ NONLOC.SLASH [ LIST #sfirst,
                                   LAST #smiddle ] ] >,
    NONLOC.SLASH [ LIST #sfirst,
                   LAST #slast ] ].
```

```
basic_two_arg := basic_lex_synsem &
  [ LOCAL.ARG-S < [ NONLOC.SLASH #s2 ],
                  [ NONLOC.SLASH #s1 ] >,
    NONLOC.SLASH.APPEND < #s1, #s2 > ].
```

Avoiding packing of phrases with and without gaps

- We use subsumption to license packing of two subtrees in a cell in the parse chart.
- Given lexical threading with some arguments still waiting to be picked up, we cannot readily determine via subsumption whether SLASH is an empty list or not.
- Added ET boolean feature –SLASHED to record whether a gap has been introduced already in a phrase.
- Also added simple boolean feature –SLPASS to enable *tough*- adjectives to block passing up of –SLASHED.

Avoiding packing of phrases with and without gaps

```
extracted_arg_phrase := unary_phrase &
  [ SYNSEM [ LOCAL.CAT.--SLASHED.BOOL + ,
             NONLOC.SLASH.LIST #slash ],
    HD-DTR.NONLOC.SLASH.LIST #slash ].

hcomp_rule := binary_rule_left_to_right &
  [ SYNSEM..--SLASHED [ OR < [ BOOL #hdsl ],
                             [ AND < [ BOOL #nhsl ],
                                     [ BOOL #slpass ]>]>],
    HD-DTR..CAT [ --SLASHED.BOOL #hdsl,
                  VAL.COMPS.FIRST..CAT.--SLPASS #slpass ],
    NH-DTR..--SLASHED.BOOL #nhsl ].
```

Performance: old vs ET on ‘hike’ profile using LKB

	old			new			reduction		
Length	**tasks**	**time**	**space**	**tasks**	**time**	**space**	**tasks**	**time**	**space**
	ϕ	ϕ (s)	ϕ (kb)	ϕ	ϕ (s)	ϕ (kb)	%	%	%
35 $<$ 40	451838	10.96	0	251899	8.67	0	44.3	20.9	0.0
30 $<$ 35	247820	5.23	0	155220	5.56	0	37.4	-6.3	0.0
25 $<$ 30	185588	3.93	0	116321	2.96	0	37.3	24.6	0.0
20 $<$ 25	87184	1.69	0	57614	1.45	0	33.9	14.0	0.0
15 $<$ 20	33827	0.64	0	23439	0.64	0	30.7	0.1	0.0
10 $<$ 15	10205	0.19	0	7551	0.20	0	26.0	-6.7	0.0
5 $<$ 10	3050	0.07	0	2410	0.06	0	21.0	12.6	0.0
0 $<$ 5	605	0.02	0	523	0.02	0	13.6	-13.5	0.0
Total	**26155**	**0.52**	**0**	**17600**	**0.48**	**0**	**32.7**	**8.4**	**0.0**

(generated by [incr tsdb()] at 24-jun-2023 (09:18 h))

Performance: old vs ET on ‘hike’ profile using ACE

Length	old			new			reduction		
	tasks	time	space	tasks	time	space	tasks	time	space
	ϕ	ϕ (s)	ϕ (kb)	ϕ	ϕ (s)	ϕ (kb)	%	%	%
$35 < 40$	-1	6.55	1099656	-1	3.70	711052	-1.0	43.4	35.3
$30 < 35$	-1	3.81	666796	-1	2.16	474712	-1.0	43.3	28.8
$25 < 30$	-1	2.45	454099	-1	1.88	372725	-1.0	23.3	17.9
$20 < 25$	-1	1.09	213749	-1	0.89	184827	-1.0	18.2	13.5
$15 < 20$	-1	0.45	93972	-1	0.40	89523	-1.0	10.4	4.7
$10 < 15$	-1	0.15	33932	-1	0.14	33803	-1.0	8.5	0.4
$5 < 10$	-1	0.05	13720	-1	0.05	14213	-1.0	1.8	-3.6
$0 < 5$	-1	0.01	4289	-1	0.01	4496	-1.0	1.1	-4.8
Total	**-1**	**0.35**	**71909**	**-1**	**0.29**	**64515**	**-1.0**	**17.7**	**10.3**

(generated by [incr tsdb()] at 24-jun-2023 (09:32 h))

Next steps

- Allowing SLASH to contain more than one gap

  This is a puzzle that I don't know how to solve ___ ___.

  Currently:

```
phrase := phrase_or_lexrule &
  [ SYNSEM.NONLOC.SLASH.LIST 0-1-list ].
```

- Using ET append for semantics features RELS, HCONS, ICONS