Optimization of HPSG Grammar Implementations in Trale

Georgiana Dinu

Summary

- The optimization problem
  - Background
  - Examples

Summary

- The optimization problem
  - Background
  - Examples
- Outlines of a possible solution

Summary

- The optimization problem
  - Background
  - Examples
- Outlines of a possible solution
- A simple example
  - Transformation
  - Results

Summary

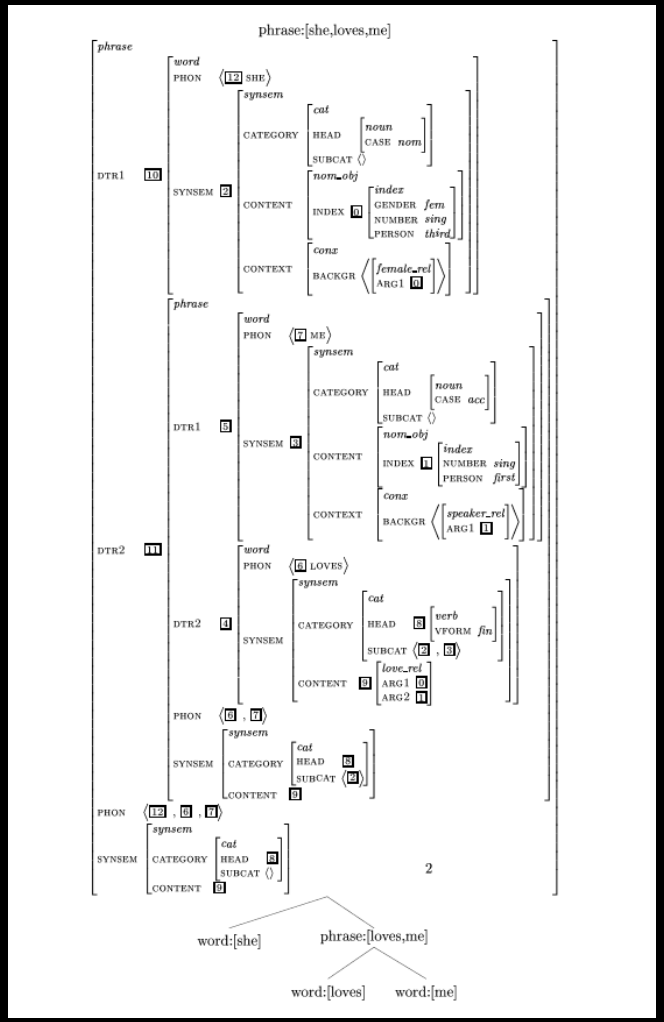
- The optimization problem
  - Background
  - Examples
- Outlines of a possible solution
- A simple example
  - Transformation
  - Results
- Discussion and Conclusion
  - Extending the solution
  - Computational aspects
  - Further work

HPSG implementations in TRALE

Roughly:

- Signature
- Lexicon (Lexical Items, Lexical Rules)
- Phrase Structure Rules
- Principles
- Additional Utilities (e.g. Macros, Definite Clauses, Functional Descriptions)

A parse query



Query:

rec[she, loves, me]

PS rules: Example 1

```
bot
   sign
      phrase dtr1:sign dtr2:sign

 (phrase, synsem:category:subcat:[],
           dtr1:Subj,
           dtr2:Head)   ===>
 cat> Subj,
 cat> Head.
```

(Example 3.1.4 in ’A Web-based course in Grammar Formalisms and Parsing’, Frank Richter)

PS rules: Example 2

```
sign
  phrase daughters:const_struc
  const_struc hdtr:sign ndtr:sign
    ...
    hs_struc
    ...

(phrase, synsem:loc:cat:val:(subj:e_list, comps:e_list),
          daughters:(hs_struc,
                     hdtr:Hdtr,
                     ndtr:Ndtr))   ===>
cat> (Ndtr, synsem:Synsem),
cat> (Hdtr, synsem:loc:cat:val:(subj:[Synsem], comps:e_list)) .
```

(Example 3.2.3 in ’A Web-based course in Grammar Formalisms and Parsing’, Frank Richter)

PS rules: Example 3

```
sign
   intro_complex_sign
      phrase non_head_dtrs:list
        headed_phrase head_dtr:sign
          ...
             head_adjunct_phrase
          ...
        non_headed_phrase
          ...
h_adj ## (head_adjunct_phrase,
             head_dtr:HeadDtr,
             non_head_dtrs:[(NonHeadDtr,
                             synsem:loc:cat:head:pre_modifier:minus)])
  ===>  cat> HeadDtr,
        cat> NonHeadDtr.
```

(Chapter 13 Example Grammar in ’Head-Driven Phrase Structure Grammar: Eine Einführung’, Stefan Müller)

PS rules: BNF syntax

```
<rule> ::= <rule_name> rule <desc> ===> <rule_body>.
<rule_body> ::= <rule_clause>
              |<rule_clause>, <rule_body>
<rule_clause> ::= cat> <desc>
                |cats> <desc>
                |sem_head> <desc>
                |goal> <goal>
                |sem_goal> <goal>
<desc> ::= <type>
          |<variable>
          |(<feature>:<desc>)
          |(<desc>,<desc>)
          |(<desc>;<desc>)
          |(=\= <desc>)
          |<path>==<path>
```

Eliminating ’daughters’ attributes

- Remove ’daughters’ attributes from mother nodes in PS rules
- Maintain the same functionality
- Approach:
  - Check the effects of the removal
  - Recover the original grammar

Eliminating ’daughters’ attributes

’daughters’ attributes references in implementations

- Signature
- Descriptions/Functional Descriptions
  - Macros
  - PS rules
  - Constraints

Head-Feature Principle

```
% Head Feature Principle 1
phrase *> (synsem:category:head:H,
     dtr2:synsem:category:head:H).

% Head Feature Principle 2
phrase *> (synsem:loc:cat:head:Head,
           daughters:hdtr:synsem:loc:cat:head:Head).

% Head Feature Principle 3
headed_phrase *>
   (synsem:loc:cat:head:Head,
    head_dtr:synsem:loc:cat:head:Head).
```

Procedural attachments to PS rules

```
Subcategorization Principle
phrase *> (synsem:category:subcat:PhrSubcat,
           dtr1:synsem:Synsem,
           dtr2:synsem:category:subcat:HeadSubcat)
goal
   append(PhrSubcat,[Synsem],HeadSubcat).

append(X,Y,Z) if
   when(  ( X=(e_list;ne_list)
          ; Y=e_list
          ; Z=(e_list;ne_list)
               ),
            undelayed_append(X,Y,Z)).

 undelayed_append(L,[],L) if true.
 undelayed_append([],(L,ne_list),L) if true.
 undelayed_append([H|T1],(L,ne_list),[H|T2]) if append(T1,L,T2).
```

An implementation that we cannot hope to optimize this way

```
schema1 rule (Mother,phrase,synsem:loc:cat:subcat:[])
 ===> cat> (SubjDtr,non_word,synsem:SubjSyn),
      cat> (HeadDtr,phrase),
      goal>(head_feature_principle(Mother,HeadDtr),
       inv_minus_principle(Mother),
       subcat_principle(Mother,HeadDtr,[SubjSyn]),
       marking_principle(Mother,HeadDtr),
       spec_principle(SubjDtr,HeadDtr),
       semantics_principle(Mother,HeadDtr,[SubjDtr]),
       parochial_trace_principle(SubjDtr),
       nonlocal_feature_principle(Mother,HeadDtr,[SubjDtr]),
       single_rel_constraint(Mother),
       clausal_rel_prohibition(Mother),
       relative_uniqueness_principle(Mother,[SubjDtr,HeadDtr]),
       conx_consistency_principle(Mother,[SubjDtr,HeadDtr]),
       deictic_cindices_principle(Mother,[SubjDtr,HeadDtr])).
```

(HPSG 2.0, Gerald Penn, available at http://www.cs.toronto.edu/ gpenn/ale/files)

Transformation

```
% lexical entries

%phrase structure rules
subject_head_rule ##
 (phrase,
     synsem:category:subcat:[],
            dtr1:Subj, dtr2:Head)
   ===>
 cat> Subj,
 cat> Head.
```

```
%lexical entries

%phrase structure rules
subject_head_rule ##
 (Mother, phrase,
     synsem:category:subcat:[])

   ===>
   cat> Subj,
   cat> Head,
   goal> smp(Mother, Head),
   goal> hfp(Mother, Head),
   goal> scp(Mother, Subj, Head).
```

Transformation

```
head_complement_rule ##
 (phrase,
   synsem:category:subcat:ne_list,
            dtr1:Subj, dtr2:Head)
   ===>
 cat> Head,
 cat> Comp.
```

```
head_complement_rule ##
 (Mother, phrase,
   synsem:category:subcat:ne_list)

  ===>
 cat> Head,
 cat> Comp,
 goal> smp(Mother, Head),
 goal> hfp(Mother, Head),
 goal> scp(Mother, Comp, Head).
```

Transformation

```
% Principles


% Semantics Principle
phrase *> (synsem:content:C,
      dtr2:synsem:content:C).


% Head Feature Principle
phrase *> (synsem:category:head:H,
     dtr2:synsem:category:head:H).
```

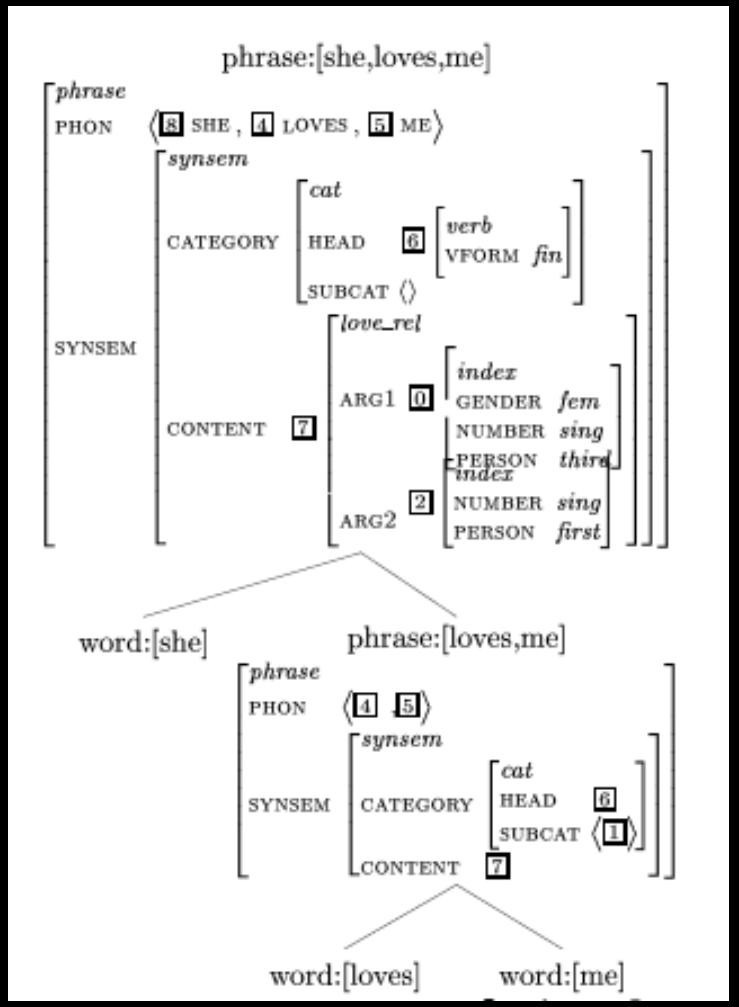
```
% Goal Definitions
% Semantics Principle Goal
smp(synsem:content:C
    synsem:content:C)
    if true.


% Head Feature Principle Goal
hfp(synsem:category:head:H,
    synsem:category:head:H)
    if true.
```

Transformation

```
% Subcategorization Principle

phrase *>
 (synsem:category:subcat:PhrSC,
   dtr1:
     synsem:Synsem,
   dtr2:
     synsem:category:subcat:HeadSC)
goal
  append(PhrSC,[Synsem],HeadSC).

% Goal Definitions
```

```
% Subcategorization Principle Goal

 scp(synsem:category:subcat:PhrSC,
     synsem:Synsem,
     synsem:category:subcat:HeadSC)
      if
     append(PhrSC,[Synsem],HeadSC).


%Goal Definition
```

Same parse query



Query:

rec[she, loves, me]

Formulating the Task

- What subset of Trale implementations?
  - HPSG grammars
  - Grammars with restrictions in PS rules and/or principles
- What do we want from the user?
  - Content
  - Form

A more complex example

```
sign
  phrase daughters:const_struc
  const_struc hdtr:sign ndtr:sign
    ...
    hs_struc
    ...

(phrase, synsem:loc:cat:val:(subj:e_list, comps:e_list),
          daughters:(hs_struc,
                     hdtr:Hdtr,
                     ndtr:Ndtr))   ===>
cat> (Ndtr, synsem:Synsem),
cat> (Hdtr, synsem:loc:cat:val:(subj:[Synsem], comps:e_list)) .
```

Computational Aspects

- What kind of transformation is involved?
  - What representation of the data we use?
  - Using already available computations
- Implementation-related technical decisions

Conclusion

- Optimization is possible
- Automatization of the process seems to be possible
- Further work
  - Finding solutions for more complex theories
  - Extending grammar examples to classes of grammars, in respect to the applied transformation
  - Explicitly defining the task
  - Implementation and testing