Transformational Semantics (TS) on a Tree Bank

Oleg Kiselyov

Tohoku University, Japan

LENLS
November 14, 2017

Outline

- **Introduction**

Transformational Approach by a FraCAS Example

Complications

Conclusions and Reflections

Motivation

Semantics-by-Transformations (TS)

- QR *but* restrained, rigorous, type preserving, mostly deterministic
- Negative predictions
- Quantifier ambiguity, scoping islands and binding, crossover, topicalization, inverse linking, (non-canonical) coordination

Carried out mechanically

- TS is precisely specified and can be carried out mechanically: Semantic calculator
- Do it in bulk and automatically

Motivation

Semantics-by-Transformations (TS)

- QR *but* restrained, rigorous, type preserving, mostly deterministic
- Negative predictions
- Quantifier ambiguity, scoping islands and binding, crossover, topicalization, inverse linking, (non-canonical) coordination

Carried out mechanically

- TS is precisely specified and can be carried out mechanically: Semantic calculator
- Do it in bulk and automatically

Not that straightforward

Motivation

Semantics-by-Transformations (TS)

- QR *but* restrained, rigorous, type preserving, mostly deterministic
- Negative predictions
- Quantifier ambiguity, scoping islands and binding, crossover, topicalization, inverse linking, (non-canonical) coordination

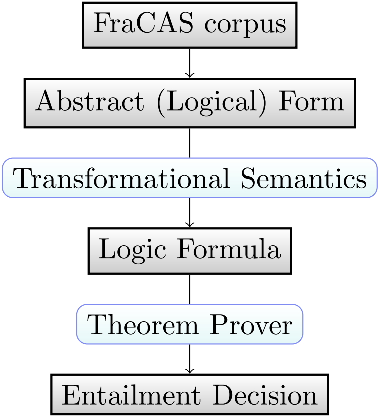
Carried out mechanically

- TS is precisely specified and can be carried out mechanically: Semantic calculator
- Do it in bulk and automatically

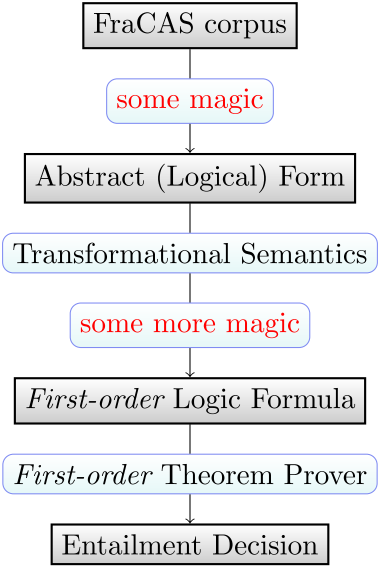
Not that straightforward

The talk is an improved version of the paper

Outline


FraCAS corpus
Abstract (Logical) Form
Transformational Semantics
Logic Formula
Theorem Prover
Entailment Decision

Outline


FraCAS corpus
some magic
Abstract (Logical) Form
Transformational Semantics
some more magic
First-order Logic Formula
First-order Theorem Prover
Entailment Decision

Outline

Introduction

- **Transformational Approach by a FraCAS Example**

Complications

Conclusions and Reflections

FraCAS

Textual inference problem set

Problem 049:

1. A Swede won a Nobel prize.
2. Every Swede is a Scandinavian.
3. A Scandinavian won a Nobel prize.

Is (3) entailed from (1) and (2)?

Annotated FraCAS

```
( (IP-MAT (NP-SBJ (D A) (ADJ Swede))
      (VBD won) (NP-OB1 (D a) (NPR Nobel) (N prize))
      (PU .))
  (ID 86_JSeM_beta_150530))

( (IP-MAT (NP-SBJ (Q Every) (ADJ Swede))
          (BEP is) (NP-OB1 (D a) (ADJ Scandinavian))
          (PU .))
  (ID 87_JSeM_beta_150530))
```

The first two Problem 049 sentences annotated within the Penn Historical Corpora System
Why To use this input format? Ask me later

The Abstract Form

```
cl (a_x (swede entity)) (won (a_y nobel_prize))

cl (every_x (swede entity)) (is_cn (scandinavian entity))
```

The Abstract Form

```
cl (a_x (swede entity)) (won (a_y nobel_prize))

cl (every_x (swede entity)) (is_cn (scandinavian entity))
```

Typed Term

`entity`: CN

`swede` : $CN \to CN$

`a_x` : $CN \to NP$

`won` : $NP \to VP$

`cl` : $NP \to VP \to S$

TS transformations

```
cl (a_x (swede entity)) (won (a_y nobel_prize))
```

TS transformations

```
cl (a_x (swede entity)) (won (a_y nobel_prize))

Ex (swede entity)
   (cl x (won (a_y nobel_prize)))
```

TS transformations

```
cl (a_x (swede entity)) (won (a_y nobel_prize))

Ex (swede entity)
   (cl x (won (a_y nobel_prize)))

Ex (swede entity)
   (Ey nobel_prize (cl x (won y)))
```

TS transformations

```
cl (a_x (swede entity)) (won (a_y nobel_prize))

Ex (swede entity)
   (cl x (won (a_y nobel_prize)))

Ex (swede entity)
   (Ey nobel_prize (cl x (won y)))
```

- QR, in a precisely specified, and typed-assured way
- Each transformation is deterministic
- The order of transformations is generally not
- We try them all

First-Order Formulas

```
fof(s1,axiom,
?[X]: ((in(X,swede) & in(X,entity)) &
   (?[Y]: (in(Y,nobel_prize) & rel(Y,won,X))))).

fof(s2,axiom,
![X]: ((in(X,swede) & in(X,entity)) =>
  (in(X,scandinavian) & in(X,entity)))).

fof(c,conjecture,
?[X]: ((in(X,scandinavian) & in(X,entity)) &
  (?[Y]: (in(Y,nobel_prize) & rel(Y,won,X))))).
```

TPTP Notation

Demo

Outline

Introduction

Transformational Approach by a FraCAS Example

- **Complications**

Conclusions and Reflections

Tecto-grammatization

Given: messy, flat annotated tree

```
( (IP-MAT (NP-SBJ (D A) (ADJ Swede))
     (VBD won) (NP-OB1 (D a) (NPR Nobel) (N prize))
     (PU .))
  (ID 86_JSeM_beta_150530))
```

Want: clean formula

```
cl (a_x (swede entity)) (won (a_y nobel_prize))
```

Tecto-grammatization

Given: messy, flat annotated tree

```
( (IP-MAT (NP-SBJ (D A) (ADJ Swede))
     (VBD won) (NP-OB1 (D a) (NPR Nobel) (N prize))
     (PU .))
  (ID 86_JSeM_beta_150530))
```

Cleaned-up and binarized tree

```
(IP-MAT (NP-SBJ (Q a) (nc (adj swede) (N entity)))
 (tv-app (tv won) (NP (Q a) (N nobel_prize))))
```

Want: clean formula

```
cl (a_x (swede entity)) (won (a_y nobel_prize))
```

Tecto-grammatization

Given: messy, flat annotated tree

```
( (IP-MAT (NP-SBJ (D A) (ADJ Swede))
     (VBD won) (NP-OB1 (D a) (NPR Nobel) (N prize))
     (PU .))
  (ID 86_JSeM_beta_150530))
```

Cleaned-up and binarized tree

```
(IP-MAT (NP-SBJ (Q a) (nc (adj swede) (N entity)))
 (tv-app (tv won) (NP (Q a) (N nobel_prize))))
```

Want: clean formula

```
cl (a_x (swede entity)) (won (a_y nobel_prize))
```

Tecto-grammatization is general-purpose, a composition of (many small) macro-tree transducers

Magic of type-checking

Tecto-grammatization is untyped and ad hoc

TS transformations are typed and type-preserving

Magic of type-checking

Tecto-grammatization is untyped and ad hoc

TypeChecking

TS transformations are typed and type-preserving

Logical problems

First-order Meaning

+ Semi-decidable
+ Excellent *automatic* first-order theorem provers
- How to deal with `many`, `most`, `few`, `at least three`, etc?

Logical problems

```
<problem id="002" fracas_answer="yes">
  <p idx="1">
    Every Italian man wants to be a great tenor.
  </p>
  <p idx="2">
    Some Italian men are great tenors.
  </p>
  <h>
    There are Italian men who want to be a great tenor.
  </h>
  <a> Yes </a>
  <note> Note that second premise is unnecessary and
    irrelevant </note>
</problem>
```

Conversion to XML by Bill MacCartney

Several

Every Italian man wants to be a great tenor.

```
fof(s1,axiom,
?[Y]: ((in(Y,great) & in(Y,tenor)) &
  (![X]: ((in(X,italian) & in(X,man)) =>
               rel(Y,wantToBe,X))))).
```

Several

Some Italian men are great tenors.

```
fof(s2,axiom,
(((![X]: (in(X,sks23) <=> (in(X,great) & in(X,tenor)))) &
  (![Xs21]: (in(Xs21,sks22) <=>
      (in(Xs21,italian) & in(Xs21,man))))) &
  several(sks22,sks23))).
```

Several

```
There are Italian men who want to be a great tenor.

fof(c,conjecture,
?[Y]: ((in(Y,great) & in(Y,tenor)) &
  (((![X]: (in(X,skc3) <=> in(X,exist))) &
    (![Xc1]: (in(Xc1,skc2) <=>
               ((in(Xc1,italian) & in(Xc1,man)) &
               rel(Y,wantToBe,Xc1)))))
   => several(skc2,skc3)))).
```

Several

```
fof(several1,axiom,![P,P1,Q,Q1]:
 ((several(P,Q) &
   (![X]: ((in(X,P) & in(X,Q)) => ((in(X,P1) & in(X,Q1)))))
    => several(P1,Q1))).

fof(sevmany,axiom,![P,Q]:
   (many(P,Q) => several(P,Q))).
fof(sevmost,axiom,![P,Q]:
   (most(P,Q) => several(P,Q))).
```

Definite descriptions

Problem 017:

- An Irishman won the Nobel prize for literature.
- An Irishman won a Nobel prize.

Problems

Bare Plurals

Problem 013

1. Both leading tenors are excellent.
2. Leading tenors who are excellent are indispensable.
3. Both leading tenors are indispensable.

Outline

Introduction

Transformational Approach by a FraCAS Example

Complications

- **Conclusions and Reflections**

Conclusions

TS does work on a tree bank

- QR, movement, Cooper storage,...
  in a precisely specified, and a typed-assured way
- (although that doesn't say much)
- Can do everything that natural semantics can

One Sentence vs Corpus, Manual vs. Automatic

A world of difference

Future Work

- Plurality (definite plurals, bare plurals and their multiple meanings, distributivity)
- TS with the event semantics (to deal with tense, etc)

`http://okmij.org/ftp/gengo/transformational-semantics/`