FrameNet Resource Grammar Library for GF (FN-CNL)

Normunds Grūzītis, Pēteris Paikens, Guntis Bārzdinš

Institute of Mathematics and Computer Science, University of Latvia

3rd Workshop on Controlled Natural Language (CNL), 29–31 August 2012, Zurich
Our Goal

• To develop a wide coverage CNL
  1. For paraphrasing NL in an unambiguous CNL
  2. For semantically correct translation of CNL paraphrases (a multilingual CNL)
  3. For question answering (about the discourse conveyed by the CNL text)

• This is a complex task – we build on two resources
  A. FrameNet (FN)
  B. Grammatical Framework (GF)

• The result we call FrameNet-CNL (FN-CNL)

<table>
<thead>
<tr>
<th>NL text</th>
<th>Objects</th>
<th>FN Events</th>
<th>GF-EN Paraphrase</th>
<th>GF-LV Paraphrase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sophie Amundsen was on her way home from school.</td>
<td>X1:Sophie Amundsen; X72:home; X73:school; X3:way;</td>
<td>E1:Self_motion(self_mover:X1; source:X73; goal:X72; path:X3)</td>
<td>E1:Sophie Amundsen moved from school to home.</td>
<td>E1:Sofija Amundsena pārvietojās no skolas uz mājām</td>
</tr>
<tr>
<td>She had walked the first part of the way with Joanna.</td>
<td>X4: the first part of X3; X5:Joanna;</td>
<td>E2: Self_motion(self_mover:X1; path:X4; co_theme:X5; time: during E1)</td>
<td>E2:During E1 the first part of the way Sophie Amundsen walked with Joanna.</td>
<td>E2: E1 laikā ceļa pirmo pusi Sofija Amundsena gāja kopā ar Jūrunu.</td>
</tr>
<tr>
<td>They had been discussing robots.</td>
<td>X6: robots;</td>
<td>E3: Discussion (interlocutors: X1,X5; topic:X6; time: during E2)</td>
<td>E3:During E2 Sophie Amundsen and Joanna discussed robots.</td>
<td>E3: E2 laikā Sofija Amundsena un Jūruna apsprieda robotus.</td>
</tr>
<tr>
<td>the human brain was like an advanced computer.</td>
<td>X7:the human brain; X8: an advanced computer;</td>
<td>E5: Similarity (entity1:X7; entity2:X8)</td>
<td>E5:The human brain is similar to an advanced computer.</td>
<td>E5:Cilvēka smadzenes ir līdzīgas sarežģītam datoram.</td>
</tr>
</tbody>
</table>

**Question Answering (FN-CNL Formal Semantics)**

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Was Joanna at school?</td>
<td>Yes.</td>
</tr>
<tr>
<td>Is human brain similar to a computer?</td>
<td>I do not know, but Joanna thinks so.</td>
</tr>
</tbody>
</table>
Grammatical Framework (GF)

• A toolbox for creation of multilingual CNLs
  A. Includes a predefined syntactic interlingua (common "abstract grammar")
  B. Includes "resource grammars" for translating core syntax of English and 20 other languages to/from the "abstract grammar"
  C. Includes a parser and generator (largely equivalent to Prolog DCG for CFG)

• Limitation – a purely syntactical framework. Lexical and frame semantics must be coded into each GF-application from scratch
FrameNet (FN)

- Focused on lexical and frame semantics
  A. Identifies ~1000 frames: prototypical situations (coarse, language independent) with participating semantic roles (frame-elements, FE)
  B. Identifies lexical-units invoking each frame and their syntactical valence patterns for realizing FEs
  C. Based on true corpus evidence in several languages
- Limitation – not entirely formal. Difficult to map towards sentence grammar, except for verb core valences.
  Full-text FrameNet annotation is highly overlapping (not tree-like)

We are primarily interested in verb frames and their core FEs (non-core FEs are largely same for all verbs and thus can be treated as verb-independent modifiers)
GF Resource Grammar Library (RGL)

• GF facilitates **reusability** by splitting the grammar development in two levels:
  
  – General-purpose *resource grammars* cover morphological paradigms and the realization of syntactic structures
    
    • Require in-depth linguistic knowledge about each particular language, as well as in-depth GF knowledge
    
    • Implement a common **syntactic API**
  
  – Domain-specific *application grammars* (CNLs) are built on top of resource grammars
    
    • However, application grammar developers still have to manipulate with **syntactic constructors**
Proposal: FrameNet RGL

• Built on top of the syntactic GF RGL and FrameNet
  – A common **semantic interlingua** (API)
  – Provides **mapping** from semantic valences to their syntactic realization

• Application grammar (CNL) developers would manipulate with **semantic constructors** *(predicates)*
  – **Predicates**: domain and language independent
    • The **robust** verb frames
  – **Arguments**: domain and language specific
    • Frame **core** elements; remains the current approach
Use-Case: MOLTO Phrasebook

- Precise translation of standard touristic phrases
- Implemented in 15+ languages
- Defines ~40 categories and ~300 functions
  - 20+ “actions” ≈ frames (ALive, ALike, AWant, AWantGo etc.)

- I live in Belgium[
- I want a pizza[
- [Do] you like this pizza[?]
- I want to go to a museum[

---

Abito in Belgio[[] (Italian)

PSentence : Phrase
  
SProp : Sentence
  
PropAction : Proposition

Residence IMale : Person
  \Belgium : Country

Resident
Phrasebook: Abstract & Common Concrete

**cat** -- abstract categories

- Action
- Country
- Person

**fun** -- abstract functions

- **IMale** : Person
- **Belgium** : Country
- **ALive** : Person -> Country -> Action

**lincat** -- category linearization types

- Action = **Cl**; -- clause
- Country = **NP**;
- Person = {name: **NP**; isPron: **Bool**; poss: **Quant**};

**lin** -- function linearization rules

- **IMale, IFemale** = mkPerson Syntax.i_Pron;
## RGL API: Syntactic Constructors

<table>
<thead>
<tr>
<th>Function</th>
<th>Arguments</th>
<th>Value</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>mkC1</td>
<td>NP  VP</td>
<td>C1</td>
<td><em>she always sleeps</em></td>
</tr>
<tr>
<td>mkC1</td>
<td>NP  V2  NP</td>
<td>C1</td>
<td><em>she loves him</em></td>
</tr>
<tr>
<td>mkC1</td>
<td>NP  VV  VP</td>
<td>C1</td>
<td><em>she wants to sleep</em></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mkVP</td>
<td>VP  Adv</td>
<td>VP</td>
<td><em>to sleep here</em></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mkNP</td>
<td>Det  CN</td>
<td>NP</td>
<td><em>the old man</em></td>
</tr>
<tr>
<td>mkNP</td>
<td>PN</td>
<td>NP</td>
<td><em>Paris</em></td>
</tr>
<tr>
<td>mkNP</td>
<td>Pron</td>
<td>NP</td>
<td><em>we</em></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mkCN</td>
<td>N</td>
<td>CN</td>
<td><em>house</em></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mkAdv</td>
<td>Prep  NP</td>
<td>Adv</td>
<td><em>in the house</em></td>
</tr>
</tbody>
</table>
Phrasebook: English

Belgium = mkNP (mkPN "Belgium")
Museum = mkPlaceKind "museum" "at" ; -- {name:NP ; at:Adv ; to:Adv}
Pizza = mkCN (mkN "pizza")

-- Cl -> NP VP // VP -> VP Adv // Adv -> Prep NP
ALive pers country = mkCl pers.name
    (mkVP (mkVP (mkV "live"))) (mkAdv SyntaxEng.in_Prep country));

-- Cl -> NP V2 NP
ALike pers item = mkCl pers.name (mkV2 (mkV "like")) item ;

-- Cl -> NP V2 NP
AWant pers obj = mkCl pers.name (mkV2 (mkV "want")) obj ;

-- Cl -> NP VV VP // VP -> VP Adv
AWantGo pers place = mkCl pers.name SyntaxEng.want_VV
    (mkVP (mkVP IrregEng.go_V) place.to) ;
Observations

• When one gets used to..
  – the syntactic API
  – the best practices, trade-offs and workarounds

• ..it becomes a rather routine work (for a native or fluent speaker) to “copy-paste-edit” the clause level patterns
  – among functions, applications and (partly) even languages
  – providing a miniature domain-specific framenet for each application

• But beware of “exceptions”: verb-dependent realizations of clauses (e.g. love vs. like in Russian Phrasebook)
  – Я[ NOM] люблю тебя[ ACC] (I love you)
  – Я[ NOM] нравлю эту пиццу[ ACC] (*I am liked by this pizza*)
  → Мне[ DAT] нравится эта пицца[ NOM]
  → I like this pizza)
FrameNet Data

- E.g. the **Residence** frame:
  - Core elements: Resident, Co_resident, Location
  - Lexical units: *camp, dwell, inhabit, live, lodge, occupy, ..., stay*

- E.g. the lexical entry **Residence.live**:

<table>
<thead>
<tr>
<th>FE</th>
<th>Total</th>
<th>Pattern</th>
<th>Total</th>
<th>Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resident</td>
<td>143</td>
<td>NP.Ext (90%)</td>
<td>98</td>
<td>Location</td>
</tr>
<tr>
<td>Co_resident</td>
<td>14</td>
<td>PP.De p(86%)</td>
<td>71%</td>
<td>PP.De p</td>
</tr>
<tr>
<td>Location</td>
<td>131</td>
<td>PP.De p (81%)</td>
<td>7</td>
<td>Resident Co_resident</td>
</tr>
</tbody>
</table>

with
among
9
3

in
on

71
8

...
Assumptions

• There is a common syntactic realization of a frame that is reused by most verbs that evoke the frame
  – There can be alternative realizations that are specific to particular verbs or groups of verbs (systematic exceptions)
  – Prepositions, in general, do not depend on the frame, but often there is a dominant preposition per frame element (if it is realized as a PP)

• It is possible to choose a default lexical unit (LU) per frame to be used in the linearization, if a specific verb is not provided
  – The most general and/or the most frequently used LU

• In the CNL settings, it is often sufficient that only core valences (according to FrameNet) are available
Semantic vs. Syntactic API

- **ALive** \( p \ co = \)
  - \( \text{mkCl} \ p.\text{name} \ (\text{mkVP} \ (\text{mkVP} \ (\text{mkV} \ "live") \ (\text{mkAdv} \ \text{in} \_\text{Prep} \ co))} \)
  - \( \text{mkCl} \ p.\text{name} \ (\text{mkVP} \ (\text{mkVP} \ (\text{mkV} \ "abitare") \ (\text{mkAdv} \ \text{in} \_\text{Prep} \ co))} \)
  - **Residence** \( p.\text{name} \ NIL \ co \)
    - Resident  Co_resident  Location

- **ALike** \( p \ it = \)
  - \( \text{mkCl} \ p.\text{name} \ (\text{mkV2} \ (\text{mkV} \ "like") \ it) \)
  - \( \text{mkCl} \ it \ (\text{mkV2} \ (\text{mkV} \ (\text{piacere} \_64 \ "piacere") \ \text{dative}) \ p.\text{name}) \)
  - **Experiencer\_focus** "piacere" \( p.\text{name} \ it \ NIL \ NIL \)
    - LU  Experiencer  Content  Event  Topic

- **AWantGo** \( p \ pl = \)
  - \( \text{mkCl} \ p.\text{name} \ \text{want} \_\text{VV} \ (\text{mkVP} \ (\text{mkVP} \ \text{IrregEng} \_\text{go} \_\text{V}) \ pl.\text{to}) \)
  - \( \text{mkCl} \ p.\text{name} \ \text{want} \_\text{VV} \ (\text{mkVP} \ (\text{mkVP} \ \text{LexiconIta} \_\text{go} \_\text{V}) \ pl.\text{to}) \)
  - **Desiring** \( p.\text{name} \ (\text{Motion} \ \text{go} \_\text{V} \_s \ NIL \ NIL \ pl.\text{name}) \)
    - Experiencer  Event  LU  Theme  Source  Goal
FrameNet RGL: Implementation

Residence  res  NIL  loc =  -- NP  NIL  NP
    mkCl  res  (mkVP  (mkVP  (mkV  "live"))
            (mkAdv  in_Prepp  loc));  -- "in": 67%

Residence  lu  res  NIL  loc =  -- V  NP  NIL  Adv
    mkCl  res  (mkVP  (mkVP  (mkV  lu))  loc);

Residence  lu  res  co_res  loc =  -- V  NP  Adv  Adv
    mkCl  res  (mkVP  (mkVP  (mkVP  (mkV  lu))  co_res)  loc);

Experiencer_focus  lu  exp  cont  NIL  NIL  =  case  lu  of  {
    "amare"  =>  mkCl  exp  (mkV2  (mkV  "amare"))  cont  ;
    "piacere"  =>
        mkCl  cont  (mkV2  (mkV  (piacere_64  "piacere"))  dative)  exp  ;
} ;
### FrameNet RGL API: Semantic Constructors

<table>
<thead>
<tr>
<th>Function (FN frame)</th>
<th>Arguments</th>
<th>Value</th>
<th>Mapping to FEs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Residence</td>
<td>NP   NIL  NP</td>
<td>Cl</td>
<td>Resident</td>
</tr>
<tr>
<td></td>
<td>Str  NP   NIL  Adv</td>
<td>Cl</td>
<td>Co_resident</td>
</tr>
<tr>
<td></td>
<td>Str  NP   Adv  Adv</td>
<td>Cl</td>
<td>Location</td>
</tr>
<tr>
<td>Experiencer_focus</td>
<td>NP   NP   NIL  NIL</td>
<td>Cl</td>
<td>Experiencer</td>
</tr>
<tr>
<td></td>
<td>Str  NP   NP   NIL  NIL</td>
<td>Cl</td>
<td>Content Event Topic</td>
</tr>
<tr>
<td>Motion</td>
<td>NP   NP   NP</td>
<td>Cl</td>
<td>Theme Source Goal</td>
</tr>
<tr>
<td></td>
<td>Str  NP   NP   NP</td>
<td>Cl</td>
<td>Source Goal</td>
</tr>
<tr>
<td></td>
<td>Str  NIL  NIL  NP</td>
<td>VP</td>
<td>Theme Source Goal</td>
</tr>
<tr>
<td>Desiring</td>
<td>NP   VP</td>
<td>Cl</td>
<td>Experiencer Event</td>
</tr>
<tr>
<td></td>
<td>Str  NP   VP</td>
<td>Cl</td>
<td>Experiencer Event</td>
</tr>
</tbody>
</table>

...
FrameNet RGL API: Next Step

<table>
<thead>
<tr>
<th>Function (FN frame)</th>
<th>Arguments</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Residence</td>
<td>Resident, Location</td>
<td>Cl</td>
</tr>
<tr>
<td>Str Resident, Location</td>
<td></td>
<td>Cl</td>
</tr>
<tr>
<td>Str Resident, Co_resident, Location</td>
<td></td>
<td>Cl</td>
</tr>
<tr>
<td>Experiencer_focus</td>
<td>Experiencer, Content</td>
<td>Cl</td>
</tr>
<tr>
<td>Str Experiencer, Content</td>
<td></td>
<td>Cl</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motion</td>
<td>Theme, Source, Goal</td>
<td>Cl</td>
</tr>
<tr>
<td>Str Theme, Source, Goal</td>
<td></td>
<td>Cl</td>
</tr>
<tr>
<td>Str Goal</td>
<td></td>
<td>VP</td>
</tr>
<tr>
<td>Desiring</td>
<td>Experiencer, Event</td>
<td>Cl</td>
</tr>
<tr>
<td>Str Experiencer, Event</td>
<td></td>
<td>Cl</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FrameNet RGL: Ongoing Work

• Language-specific realization of CNL clauses can be hidden behind the language-independent FrameNet API
  – Frames can be specified already in the common concrete syntax
  – Resulting grammars would more generic and easier to extend

• Language-specific FrameNet resource grammars can be acquired semi-automatically from FrameNet data that include mapping to syntactic patterns and statistics from FrameNet-annotated corpora

• Abstract syntax trees could be annotated with FrameNet
  – Parsing with FrameNet RGL = semantic parsing (semantic role labeling)
<table>
<thead>
<tr>
<th>NL text</th>
<th>Objects</th>
<th>FN Events</th>
<th>GF-EN Paraphrase</th>
<th>GF-LV Paraphrase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sophie Amundsen was on her way home from school.</td>
<td>X1:Sophie Amundsen; X72:home; X73:school; X3:way;</td>
<td>E1:Self_motion( self_mover:X1; source:X73; goal:X72; path:X3)</td>
<td>E1:Sophie Amundsen moved from school to home.</td>
<td>E1:Sofija Amundsena pārvietojās no skolas uz mājām</td>
</tr>
<tr>
<td>She had walked the first part of the way with Joanna.</td>
<td>X4: the first part of X3; X5:Joanna;</td>
<td>E2: Self_motion( self_mover:X1; path:X4; co_theme:X5; time:during E1)</td>
<td>E2:During E1 the first part of the way Sophie Amundsen walked with Joanna.</td>
<td>E2: E1 laikā ceļa pirmo pusi Sofija Amundsena gāja kopā ar Jūrunu.</td>
</tr>
<tr>
<td>They had been discussing robots.</td>
<td>X6: robots;</td>
<td>E3: Discussion (interlocutors: X1,X5; topic:X6; time:during E2)</td>
<td>E3:During E2 Sophie Amundsen and Joanna discussed robots.</td>
<td>E3: E2 laikā Sofija Amundsena un Jūruna apsprieda robotus.</td>
</tr>
<tr>
<td>the human brain was like an advanced computer.</td>
<td>X7:the human brain; X8: an advanced computer;</td>
<td>E5: Similarity (entity1:X7; entity2:X8)</td>
<td>E5:The human brain is similar to an advanced computer.</td>
<td>E5: Cilvēka smadzenes ir līdzīgas sarežģītam datoram.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>QuestionAnswering (FN-CNL FormalSemantics)</th>
<th>Was Joanna at school?</th>
<th>Yes.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Is human brain similar to a computer?</td>
<td>I do not know, but Joanna thinks so.</td>
</tr>
</tbody>
</table>
FN-CNL Formal Semantics

• Semantic gap between NL and FN-CNL is bridged by
  A. FrameNet annotation
  B. Anaphora tracking for objects
  C. Chronology tracking

• The key observation: verb frames are largely universal – domain-specific are only objects (NPs)
  – Decoupling via subPropertyOf mechanism:
    “I went to kitchen” / “I went to Zurich”  
    household domain  travel domain
Commercial Transaction Scenario

The event type known as Commercial Transaction is a small abstract bit of history in which:

- **Stage 1**: B wants G, owns M; S wants M, owns G.
- **Transition**: by explicit or implicit contract -
  (a) B gives M to S;
  (b) S gives G to B
- **Stage 2**: B owns G; S owns M

The verb *pay* evokes this frame; in particular, it evokes the bit about the buyer handing the money over to the seller. The question

*How much did Will pay for his laptop?*

invites us to have in mind an instance of this frame in which

Will is B,
the laptop is G,
the seller, S, is irrelevant, and
the requested information concerns M.

---

A slide from C.J. Fillmore presentation at FrameNet Masterclass (TLT-8, Milano, 2009)
Formalizing FN-CNL Semantics

• Formal FN-CNL semantics based on
  A. Frames implemented as stepwise procedures creating and deleting RDF-triples (micro-relations)
  B. Anaphors are typed in domain ontology
    • Domain-ontology may include inference rules/procedures going beyond DL limitations
  C. The discourse is history of all steps in time, modelled as a sequence of RDF named-graphs
Sophie Amundsen was on her way home from school. She had walked the first part of the way with Joanna. They had been discussing robots.
Self_motion frame implementation

- Prolonged events implemented as two steps:
  1. STAGE1 + TRANSACTION
  2. STAGE2
- Splitting of prolonged events in two steps enables implementation of “during” temporal relation

selfMotion(1,GR:N,SelfMover,Goal,Source,Path,Cotheme) :-
  rdf_assert(SelfMover, movesAlong,Path,GR:N),
  rdf_assert(Path, from,Source,GR:N),
  rdf_assert(Path, to,Goal,GR:N),
  rdf_assert(Cotheme,boundTo,SelfMover,GR:N).

selfMotion(2,GR:N,SelfMover,Goal,Source,Path,Cotheme) :-
  rdf_retractall(SelfMover,movesAlong,Path,GR:N),
  rdf_retractall(Cotheme,boundTo,SelfMover,GR:N),
  rdf_retractall(SelfMover,near,Source,GR:N),
  rdf_assert(SelfMover, near,Goal,GR:N).
Sophie’s world domain-ontology
(enables additional inferences over discourse)

Plural properly shall be treated as subclasses (with existential quantification) or whole class (with universal quantification). This primarily applies to ontological sentences covered by ACE/OWL.
Question answering over discourse: SPARQL+Reasoner

- Question in FN-CN
  - “Was Joanna(x5) at school(x73)?”

- Question in SPARQL
  - SELECT ?step
    WHERE { GRAPH ?step {x5 near x73}}

- Answer
  - ?step = “b:10”
  - This means “Yes, there was step b:10 when Joanna was at school”

C:\pellet-2.3.0>pellet query --query-file gbquery.txt rez.owl >reasongb.txt

Query Results:

step
---------------------------------------------
b:10

This Pellet example is constructed, because since StarDog introduction Pellet does not support NamedGraphs in SPARQL
Conclusions and Future work

• FN-CNL is manually tested on few examples, full evaluation requires implementation:
  – Mapping between FN-CNL abstract-syntax and multilingual concrete syntaxes in GF can be semi-automatically generated from FrameNet data (moderate amount of work)
  – Implementation of formal semantics requires defining RDF-triple interpretation procedures for all FrameNet frames (large amount of work)
Backup slides
We are mostly interested in verb Frames and their Core FEs
(non-core FEs are largely same for all verb frames and thus can be treated independently of specific verbs)
Size of FrameNet 1.5 (basis of FN-CNL)
RDF NamedGraphs sequence (discourse)