A word about the world: semantic decomposition in commonsense reasoning

Ivan Rygaev

Laboratory of Computational Linguistics
Kharkevich Institute for Information Transmission Problems RAS, Moscow, Russia

Heinrich Heine Universität CRC991 Colloquium
Düsseldorf, November 22, 2018

ETAP-4

ETAP-4 linguistic processor

- ETAP-4 has been developed in our lab since 1980s
  - Available for free download since yesterday
- The goal
  - To build functional model of natural language
- Theoretical grounds
  - Igor Melchuk's meaning-text theory
  - Jury Apresjan's theory of integrated linguistic description

Theoretical grounds

- Meaning-text theory
  - Language maps meaning to text and vice versa
  - Several layers of representation – PhonR, MorphR, SyntR, SemR
  - Explanatory combinatorial dictionary
  - Lexical functions
- Integrated linguistic description
  - Dictionary and a grammar should produce a unified account covering the whole of language with no gaps

ETAP-4 main functions

- Comprehensive support for Russian and English
- Syntactic parsing
  - Building dependency trees
- Machine translation
  - On the level of deep syntactic structures or UNL
- UNL conversion and deconversion
  - Universal Networking Language
- Deep semantic analysis (for Russian only)
  - Logical form with inferences

ETAP-4 resources

- Combinatorial dictionary
  - More than 100 000 entries for Russian, English and UNL
- Transformation rules
  - For syntactic structure creation and modification
  - Written in a formal language FORET
- Ontology
- Repository of individuals
- Inference (semantic decomposition) rules

SynTagRus

- The largest syntactically annotated corpus for Russian
  - About 70 000 sentences
- A part of Russian National Corpus
- Annotations are made automatically by ETAP with manual verification and correction
- Used to create statistical parsers for Russian
- Statistics from SynTagRus is used in ETAP for syntactic disambiguation

Winograd Schema Challenge

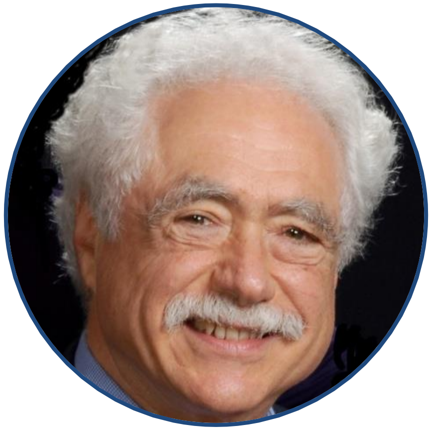
Winograd Schema Challenge

- A test for computer intelligence
- More convincing than the Turing test that machines can think
- Based on analysis of the short text of 1-3 sentences and a question on them
- Special type of anaphora resolution problem
- Linguistic features, collocation statistics, selectional restrictions do not help
- Some kind of world knowledge is required

Key people

Hector Levesque

Ernest Davis



Terry Winograd

Leora Morgenstern

Turing test criticism

- Turing test was formally passed by a chat-bot Eugene Goostman in 2014
- But does the chat-bot think?
- Is *conversation* the right way of evaluation?
  - Subjective
  - Encourage verbal acrobatics and trickery
- Turing Test requires *deception*
  - Must fool an interrogator that it is a person
  - Do we need this from an intelligent machine? For which purposes?

Winograd schemas

- Proposed by Hector Levesque in 2011
- The trophy doesn't fit in the brown suitcase because **it**'s too *big*. What is too *big*?
  - the trophy
  - the suitcase
- Joan made sure to thank Susan for all the help **she** had *given*. Who had *given* the help?
  - Joan
  - Susan
- Terry Winograd provided the first example in 1970

Winograd schema structure

- Anaphora resolution problem
- There are two potential antecedents in the sentence
- Linguistic features, collocation statistics and selectional restrictions do not help much
- Changing a special word in the sentence reverts the correct answer (*big -> small*)
- The trophy doesn't fit in the brown suitcase because **it**'s too *small*. What is too *small*?
  - the trophy
  - the suitcase

Commonsense knowledge

- People are good on Windograd Schemas
- Tests show 91-92% correct answers.
- What is required to get the right answer?
- Understanding of the verb ‘fit’
  - if A fits into B then A must be smaller than B.
- Understanding of the connective ‘because’
  - Changing it to ‘in spite of’ also reverts the answer.
- Implicit information must be extracted from the text to pass the test

WSs preparation

- The wrong answer need not be logically inconsistent:
- Tom threw his bag down to Ray after **he** reached the *top* of the stairs. Who reached the *top* of the stairs?
  - Tom
  - Ray
- Alternate special word need not be the opposite:
- The man couldn't lift his son because **he** was so *weak/heavy*. Who was *weak/heavy*?
  - the man
  - the son

WSs preparation

- WS must not be ‘too obvious’:
- The women stopped taking the pills because **they** were *pregnant/cancerogenic*.
  Which individuals were *pregnant/cancerogenic*?
  - the women
  - the pills
- Selectional restrictions help:
  - Only women can be pregnant, not pills
  - Only pills can be cancerogenic, not women
- The first sentence can be totally ignored

WSs preparation

- WS must not be ambiguous for humans
- Frank was *jealous* when Bill said that **he** was the winner of the competition. Who was the winner?
  - Frank
  - Bill
- Frank was *pleased* when Bill said that **he** was the winner of the competition. Who was the winner?
  - Frank
  - Bill
- It is not unreasonable that Bill's victory pleased Frank

Competition

- The first competition was held in July 2016 at IJCAI conference in New York
- It was organized in two rounds:
    1. Sentences from real texts (children's literature) rather than constructed ones. They exhibited all the properties of WS but did not have an alternative variant.
    2. Actual constructed WSs with an alternative variant
- Motivation for two rounds:
    - Not to reveal WSs to contestants who are not ready yet
    - Increase relevance of the test by using real examples

Competition

- There were 60 questions in the first round and 60 in the second one.
  - To proceed to the second round a contestant had to score at least 90% correct in the first one.
- None of the solutions achieved that score
  - The second round was not held
- The big prize was offered to the team who would achieve at least 90% in both rounds
  - Three smaller prizes were offered to the top programs achieved at least 65% in the first round

Competition results

- Six solutions of four teams where presented:

Contestant	Number correct	Percentage correct
Patrick Dhondt	27	45%
Denis Robert	19	31.666%
Nicos Issak	29	48.33%
Quan Liu (1)	28	46.9% (48.33)*
Quan Liu (2)	29	48.33% (58.33)*
Quan Liu (3)	27	45% (58.33)*

- Random answering could yield 45%

Results assessment

- None of the solutions got over the 65% threshold to receive even the smaller prize
- Four of the six programs showed scores around the chance level or even worse
- The next test had been scheduled for AAAI-2018 (Feb), but it was cancelled
  - Several participants dropped out at the last minute
- Text understanding by machines is an unsolved task yet

SemETAP

SemETAP semantic text analyzer

- SemETAP
  - A part of ETAP-4 linguistic processor
  - Translates an original sentence to a language-independent semantic representation in a formal language
  - Applies logical rules (semantic concept decomposition) to infer new knowledge
- Semantic representation
  - Based on Semantic Web standards (OWL, RDF, SPARQL)
  - Can be seen as a semantic graph or a formula in predicate logic

Text understanding

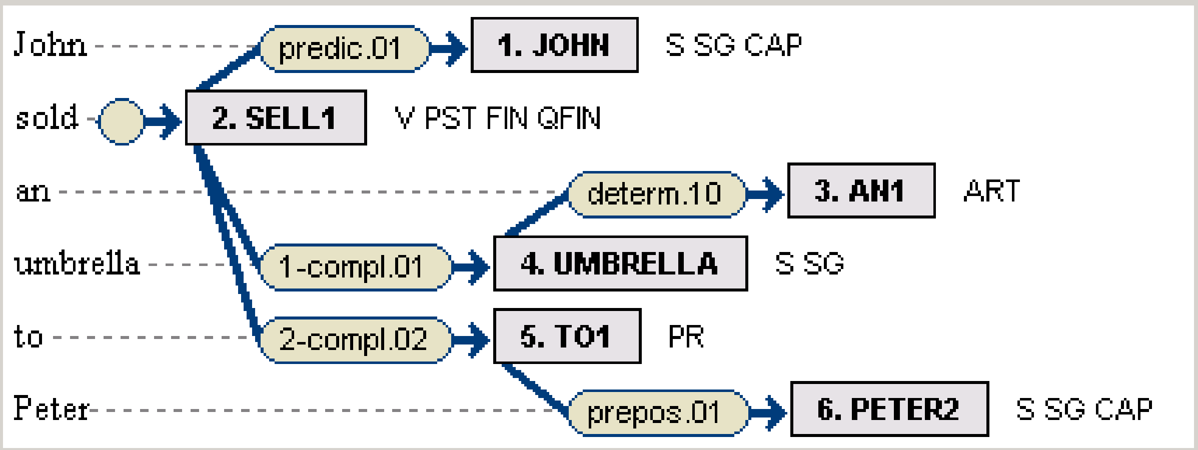
- The ultimate goal
  - To achieve near-human understanding of the text
- How can we measure understanding?
  - The amount of inferences that can be made out of the text
- How can we test inferences?
  - Questioning
- SemETAP is able to answer questions for which there is no direct answer in the original text

An example

- Input sentence:
  - *John sold an umbrella to Peter*
- Questions (easy for humans):
  - *Who bought the umbrella? (Peter)*
  - *What did John give to Peter? (the umbrella)*
  - *What did John get? (money)*
  - *Who owns the umbrella? (Peter)*
- Where is the knowledge?
  - In the meaning of the words *sell*, *buy*, *give*, *get* and *own*.

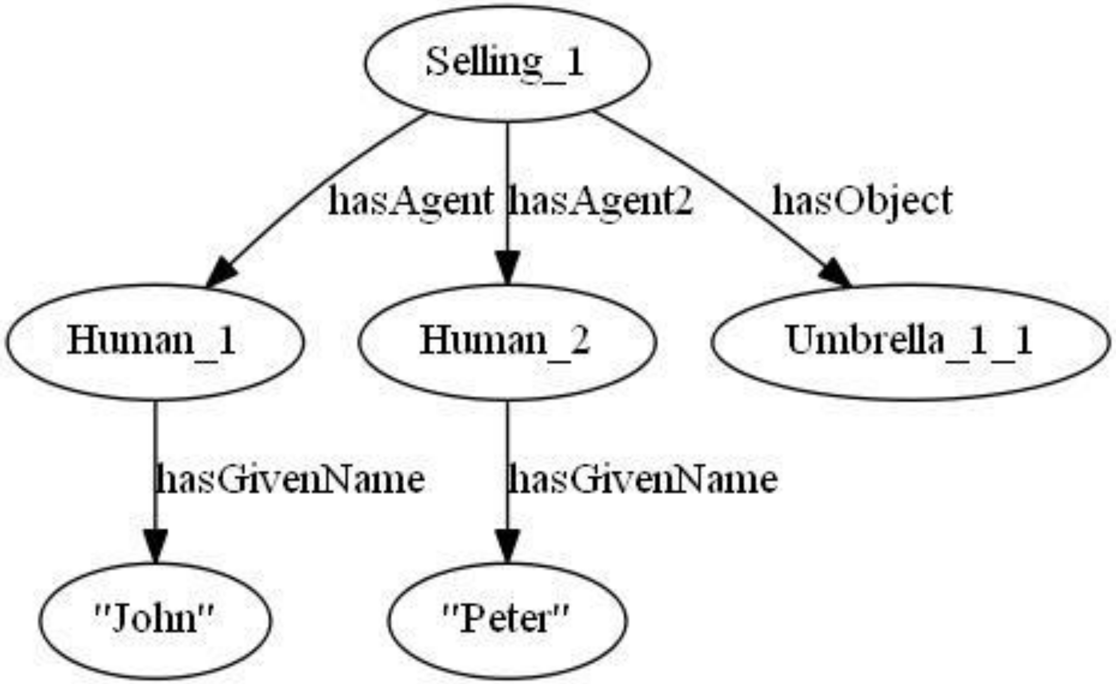
Semantic analysis steps

- Syntactic tree



- Basic semantic structure
  – Words are translated to semantic concepts and syntactic relations to semantic roles (roughly)
- Enhanced semantic structure
  – Semantic concept decomposition apply to extend the semantic graph

Basic semantic structure

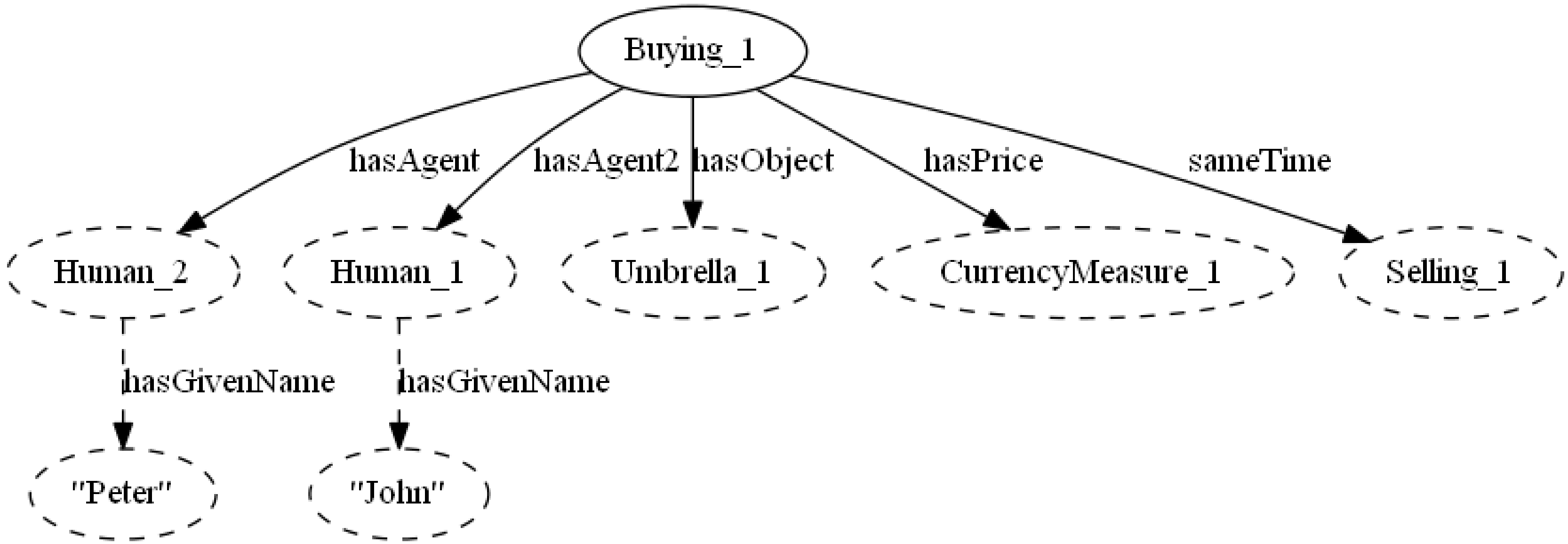



```
hasGivenName (Human_1, "John")
hasAgent (Selling_1, Human_1)
hasAgent2 (Selling_1, Human_2)
hasGivenName (Human_2, "Peter")
hasObject (Selling_1, Umbrella_1)
```

Decomposition rule example

```
Rule Selling:
      Selling(?selling) ->
      hasAgent(?selling, ?seller), Agent(?seller),
      hasAgent2(?selling, ?buyer), Agent(?buyer),
      hasObject(?selling, ?thing), Thing(?thing),
      hasPrice(?selling, ?money), Money(?money),
      hasSyncEvent(?selling, ?buying),
      Buying(?buying),
            hasAgent(?buying, ?buyer),
            hasAgent2(?buying, ?seller),
            hasObject(?buying, ?thing),
            hasPrice(?buying, ?money),
            hasSyncEvent(?buying, ?selling).
```

Enhanced semantic structure




```
hasAgent (Buying_1, Human_2)
hasAgent2 (Buying_1, Human_1)
hasObject (Buying_1, Umbrella_1)
hasPrice (Buying_1, CurrencyMeasure_1)
hasSyncEvent (Buying_1, Selling_1)
hasSyncEvent (Selling_1, Buying_1)
```

Question processing steps

- Basic semantic structure of the question is built
- Individuals corresponding to wh-words are marked in a special way
- The semantic structure is used as a pattern for SPARQL query
- The query is run against the enhanced semantic structure of the text
- Returned wh-word individuals are displayed to the user

Answer example

- *Who bought the umbrella?*

```
// Asking for: ?agent_1, ?degreeAttribute_1
// 1 answer:
// ?agent_1 = (Human #2 hasGivenName "Peter")
// ?degreeAttribute_1 = MaximalDegree
```

Etalog

Neo-Davidsonian semantics

- N-place predicate:

  `Selling (Human_1, Umbrella_1, Human_2, _)`

- Binary predicates:

  `Selling (Selling_1)`

  `hasAgent (Selling_1, Human_1) ...`

- Advantages:
  - Allows attaching adjuncts (time, location) to the event
  - Unexpressed arguments (price) can be safely ignored
  - Better resembles the syntactic structure
  - Compatible with Semantic Web standards – RDF and OWL

Disadvantages

- Unsorted list of triples is hard to read or type manually
- One variable is repeated multiple times (error prone)

```
Rule Selling:
     Selling(?selling) ->
     hasAgent(?selling,?seller), Agent(?seller),
     hasAgent2(?selling,?buyer), Agent(?buyer),
     hasObject(?selling,?thing), Thing(?thing),
     hasPrice(?selling,?money), Money(?money),
     hasSyncEvent(?selling,?buying), Buying(?buying),
     hasAgent(?buying,?buyer),
     hasAgent2(?buying,?seller),
     hasObject(?buying,?thing),
     hasPrice(?buying,?money),
     hasSyncEvent(?buying,?selling).
```

Etalog

- Etalog emerged as a language for inference rules
- Inference rules are
  - Mostly concept decomposition rules
  - Similar to word definitions but in a formal language
  - Created manually by linguists
- Motivation
  - Make the language more natural
  - Reduce the variable usage
  - Reduce the usage of special symbols

SPO notation

- Triples are written in SPO notation without brackets:

  `~~hasAgent(?selling, ?seller)~~`

  `?selling hasAgent ?seller`

- Features
  - Similar to RDF/Turtle
  - Resembles natural language
- Unary predicates are also written without brackets:

  `~~Agent (?seller)~~`

  `Agent ?seller`

Complex propositions

- Triples with the same subject can be joined together:

  `?event hasAgent ?x hasObject ?y =`

  `?event hasAgent ?x, ?event hasObject ?y`

- Features
  - Complex proposition with a single usage of the variable
  - Borrowed from RDF/Turtle but no separators are needed
  - Resembles the behavior of prepositions in natural language:
  - *go from Moscow to Düsseldorf*

Implicit variables

- Single-use variables can be omitted altogether:

```
Buying hasAgent ?x hasObject ?y =
Buying ?b hasAgent ?x hasObject ?y
```

- Features
  - Implicit variable must be preceded by a class name
  - Variable name is generated internally
  - It is not shown to the user
  - Implicit variable can be the head of a complex proposition

Nested expressions

- Complex proposition in the place of a variable:

```
?selling hasAgent (Agent ?seller) =
?selling hasAgent ?seller, Agent ?seller
```

- Subject variable can be omitted as well:

```
?selling hasAgent (Agent)
```

- Features
  - Bracketed proposition "returns" its subject variable
  - Tree-like graph can be written without mentioning variables at all

Inverse relations

- Native support for inverse relations:

```
?seller isAgentOf ?selling =
?selling hasAgent ?seller
```

- Features
  - Makes the object the subject
  - It can be combined further to form a complex proposition

```
Agent ?seller isAgentOf
(Selling hasObject (Umbrella))
```

  - Resembles the relative clause construction:
  - *An agent who sold the umbrella*

Rule in Etalog

```
Rule Selling: // Sale
    Selling ?selling ->
    ?selling
        hasAgent (Agent ?seller) // seller
        hasAgent2 (Agent ?buyer) // buyer
        hasObject (Thing ?thing) // product
        hasPrice (Money ?money) // money
        hasSyncEvent
        (    // Sale is defined through Purchase
            Buying
                hasAgent ?buyer
                hasAgent2 ?seller
                hasObject ?thing
                hasPrice ?money
                hasSyncEvent ?selling
        ).
```

Semantic structure representation

- Semantic structure is a set of triples
  - Can be expressed in Etalog
  - But it is generated automatically
  - How to choose the appropriate expression?
- Our solution
  - Do not invert any relations
  - Find the closest head following reverse direction of arrows
  - In case of loops choose the head alphabetically
  - Group all nodes by the closest head
  - Express each group as a single complex proposition

Semantic structure representation

- *John sold an umbrella to Peter*

```
Selling #1
  hasObject (Umbrella #1)
  hasAgent (Human #1 hasGivenName "John")
  hasAgent2 (Human #2 hasGivenName "Peter")
```

- Shortened variable names

```
Selling #1 =
Selling ?selling_1
```

  – Excludes names duplication
  – But keeps the class assignment explicit

Full semantic structure

- *John sold an umbrella to Peter*

```
EpistModality #1_1
  hasDegree MaximalDegree
  hasExperiencer UtteranceSpeaker
  hasTime (TimeInterval #1_1 before SpeechTimeInterval)
  hasObject
   (Complete #1_1 hasObject
       (Selling #1_1
          hasObject (Umbrella #1_1)
          hasAgent (Human #1_1 hasGivenName "John")
          hasAgent2 (Human #1_2 hasGivenName "Peter")
       )
   ).
```

Reasoning

Rule application process

- Conjunctive existential rules compatible with Datalog$^{\pm}$
  - Almost necessarily contain new variables in the rule consequent
  - Applied in the forward chaining manner (chase)
  - A rule is internally split into a number of smaller implications
  - Each implication checks the existence of one or more individuals and creates them only if they are missing
  - Some individuals can be found, others will be added
  - This happens invisibly for linguists who create the rules
  - Linguists can concentrate on the concept decomposition and ignore technical aspects of the rule application

Reasoner

- We use RDFox (from Oxford)
  - Very efficient in-memory RDF storage and reasoner
  - Optimized for chase
  - Meets all the requirements mentioned above
  - Supports equality (EGDs) through the use of sameAs predicate (both with UNA and no UNA)
  - Special mode of query execution combines a group of sameAs individuals into one individual.
  - Each Etalog rule is compiled into a number of RDFox Datalog$^{\pm}$ rules

Termination guaranteed

- Functional properties
  - Do not add an individual for the property value to the semantic structure if one already exists
- In more complex cases
  - Do not add exactly the same subgraph that already exists
  - Not a perfect rule and sometimes can create duplicates
  - Desired: merging of non-contradicting subgraphs/individuals (coreference resolution)
- Depth limit
  - Do not apply a (RDFox) rule if no variable from the rule antecedent maps to an original individual from the text

Facticity

- Truth
  - Semantic structure is a conjunction of atoms
  - But not every atom represents a true fact
- Epistemic modality
  - True facts are marked using epistemic modality – a degree of speaker/hearer's confidence in the proposition
  - Modality is assigned to an event, not to a proposition
  - The whole event is either true (took place) or false (did not take place)
- Negation applies to events in a similar fashion

Plausible expectations

- Invited inferences (implicatures)
  - Inferences which are most likely true based on the all the information we have so far
  - *John was able to leave the country*
  - Modelled using epistemic modality of medium degree
- Non-monotonic logic
  - Expectations can be confirmed or cancelled by a subsequent discourse
  - Made hidden when a corresponding confirming or disproving epistemic modality of maximal degree exists

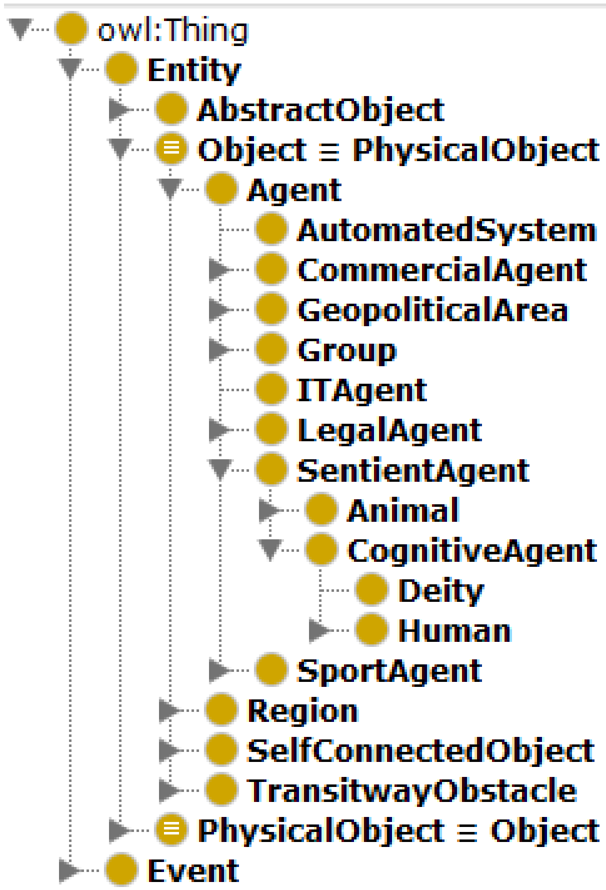
Desired Features

- Probabilistic uncertainty
  - Supported: on the level of individuals – epistemic modality
  - Desired: on the level of propositions/atoms/triples
- Disjunctive uncertainty
  - Desired: allow disjunction in the rule consequent
- Negation
  - Supported: on the level of individuals – Negation concept
  - Desired: on the level of propositions/atoms/triples
- Universal quantifier
  - Desired: at least in the context of negation

Ontology

Classes

- Classes are organized into a subsumption hierarchy in the ontology
- Classes inherit all the properties from their superclasses



Relations/properties

- Event argument properties
  - hasAgent, hasObject, hasRecepient, etc
  - Serve as labels only, do not have their own inferences
- Time interval relations
  - before, during, meetsTemporally, etc.
  - Transitivity rules are written in Etalog:

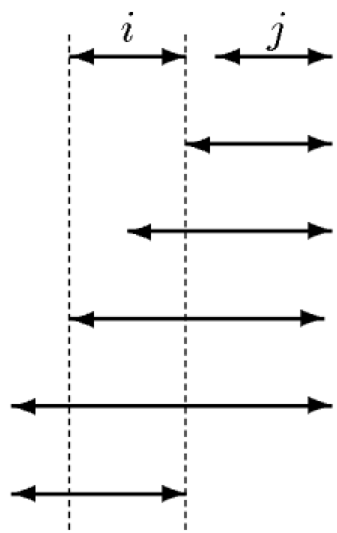
```
Rule TimeDuringMeets:
  ?a during ?b, ?b meetsTemporally ?c -> ?a before ?c.
```

- Implicative relations
  - hasSyncEvent, hasResult, hasPrecondition, etc
  - Expand epistemic modality and time in different contexts

Time

- We adopt Allen's temporal interval logic

Relation	Inverse
$Before(i, j)$	$After(j, i)$
$Meets(i, j)$	$MetBy(j, i)$
$Overlaps(i, j)$	$OverlappedBy(j, i)$
$Starts(i, j)$	$StartedBy(j, i)$
$During(i, j)$	$Contains(j, i)$
$Finishes(i, j)$	$FinishedBy(j, i)$



  – Each event is connected by hasTime relation to its TimeInterval
  – Transitivity rules form the composition table

Implicative verbs

- Implicative verbs transfer their facticity status to their complement (Karttunen 1971)
  - *John forced Mary to stay home => Mary stayed home*
- Different groups of verbs behave differently
  - *John **didn't** force Mary to stay home => ?*
  - *John prevented Mary from leaving => Mary **didn't** leave*
  - *Mary was able to leave => ?*
  - *Mary was **not** able to leave => Mary **didn't** leave*

Russian implicative verbs

- Verb behavior depends also on their aspect
  - *John achieved an increase in sales => The sales increased*
  - *John **was achieving** an increase in sales => ?*
- Also we want to capture plausible expectations
  - *John was allowed to smoke => **Probably**, John smoked*
- Need to analyze and describe for each verb:
  - in each of 4 contexts: polarity+/-, perfect+/-
  - specify one of 3 implication types: strict, plausible or none.
- Solution: implicative relations

Implicative relations

- Logical and temporal relation in which an event stands to another event from its decomposition
  - They have natural meaning on their own
  - The inference logic is encoded into the decomposition of the relation itself, no need to repeat for each verb

Main event	not started	not completed	started	completed
hasSpeakerCommitment	started	started	started	started
hasSyncEvent	not started	not completed	started	completed
hasSubEvent	not started			completed
hasPrecondition			started	started
hasResult				started
hasPreventedEvent		completed	not started	not started

Case study

- *Месси не смог спасти матч. Кто проиграл?*
  Messi could not save the match. Who was defeated?
  - Dictionary: ‘смог’ –> Succeed
  - Dictionary ‘спасти матч’ – Inhibiting hasObject (Defeat)
  - Syntax: ‘не смог спасти матч’ –> Negation hasObject (Succeed hasObject (Inhibiting hasObject (Defeat)))
  - Rule: Succeed hasObject ?event hasSyncEvent ?event =>
  - Negation hasObject (Inhibiting hasObject (Defeat))
  - Rule: Inhibiting hasObject ?event hasPreventedEvent ?event =>
  - Defeat

Case study

- *Месси не смог спасти матч. Кто проиграл?*
  Messi could not save the match. Who was defeated?
  - *Messi could not save the match =>*
  - *does not take place that Messi saved the match =>*
  - *does not take place that Messi prevented the defeat =>*
  - *does not take place that the team was not defeated =>*
  - *(Messi's) team was defeated*

Winograd Schema Challenge

Solution for Winograd schemas

- Is in progress (for Russian)
- Idea
  - Build two semantic structures – one for each potential antecedent of the pronoun
  - See which semantic structure is more consistent
- Two types of consistency
  - Two different parts of the sentence lead to the same inference with the same polarity
  - Two different parts of the sentence lead to the inference of an event with the same class and same arguments (polarity does not matter)

Solution for Winograd schemas

- The problem is to identify to which type a case belongs
- The trophy doesn't fit in the brown suitcase because **it** is too *big/small*. What is too *big/small*?
  - the trophy
  - the suitcase
- The man couldn't lift his son because **he** was so *weak/heavy*. Who was *weak/heavy*?
  - the man
  - the son

Information structure

Information structure

- Etalog is able to capture some elements of IS
- *The writer burned the novel that he had written*

```
Burning #1
   hasAgent (Writer #1)
   hasObject
     (Novel #1
         isObjectOf
           (Writing #1
               hasAgent ?writer_1
           )
     )
```

Information structure

- *The novel was burned by the writer who had written it*

```
Burning #1
  hasObject (Novel #1)
  hasAgent
    (Writer #1
       isAgentOf
        (Writing #1
           hasObject ?novel_1
        )
    )
```

Information structure

- The same situation but two different messages:
  - *What did the writer do?*
  - *Who burned the novel?*
- The two Etalog expressions
  - Represent exactly the same set of triples
  - Etalog syntax mirrors the syntax of the sentences
  - The expressions are tree-like, resembling the syntactic tree of the sentences
  - They can capture (some) communicative differences between sentences

Interlingua

- On the one hand
  - Etalog expressions are language-independent
  - They contain no language-specific features
- On the other hand
  - They contain enough information to restore the syntax of the sentence (to some extent)
- Hence
  - Etalog can serve as a language-independent interlingua
  - An intermediate semantic representation to translate from one natural language to another

Referring expressions for answers

- Question answering system:
  - *After a pass by Kerzhakov into the penalty area Arshavin with a brilliant shot in the fall hammers the ball into the net.*
  - *Which team does Arshavin play for?*
  - `(FootballTeam isObjectOf (PlaysFor hasAgent (Human hasName "Kerzhakov")))`
  - *The team which Kerzhakov plays for (The same team with Kerzhakov)*
- The information structure is also captured:
  - The expressions is tree-like with the found individual being the head of the phrase

Thank you
for your attention!