Deep semantic analysis within ETAP-4 linguistic processor

Ivan Rygaev
irygaev@gmail.com
Laboratory of Computational Linguistics, IITP RAS

IITP Lab 11 Open Workshop
Moscow, January 2019

ETAP-4

Natural language processing

- Tasks
  - Machine translation
  - Sentiment analysis
  - Topic analysis (texts classification)
  - Text summarization
  - Information extraction
  - Question answering
  - Chat bots
- All the tasks require text understanding (to some extent)

Natural language processing

- Internal tasks
  - Text segmentation (tokenization, sentence boundary disambiguation, phrase chunking)
  - Morphological analysis (stemming, lemmatization, part-of-speech tagging)
  - Syntactic analysis (building a syntactic tree)
  - Semantic analysis (meaning representation)
  - Anaphora/coreference resolution
  - Named entity recognition
  - Word sense disambiguation

ETAP-4 linguistic processor

- ETAP-4 has been developed in our lab since 1980s
  - Available for free download since November 2019
- 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 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
- Synonymous paraphrasing
- UNL conversion and deconversion
  - Universal Networking Language
- Deep semantic analysis (for Russian only)
  - Logical form with inferences

Technical side

- ETAP-4 is a big software complex
  - Written in C++ for Visual Studio (Windows)
  - More than 50 thousand lines of code
  - 16 graphical applications
  - 13 console applications
  - 37 reusable libraries (dll)
- Tasks for further development
  - Porting to other platforms (Linux)
  - Creation of a multithreaded server version

Linguistic knowledge

- Rule-based (knowledge-driven) approach
  - With some statistical components
- Linguistic knowledge
  - Dictionaries
  - Grammar rules
- Separated from the software code
  - Transparent
  - The same code is reused for different languages

ETAP-4 resources

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

Combinatorial dictionary

- A lot of grammar knowledge is in the dictionary
  - Syntactic and semantic features
  - Governance patterns
  - Lexical functions
  - Dictionary rules for specific words processing
- Governance patterns (покупать – buy)

– Buyer – NOM	*Вася покупает*
– Product – ACC	*Покупает зонтик*
– Seller – preposition У	*Покупает у Пети*
– Price – prepositions ЗА, ПО	*Покупает за/по 10 рублей*

Lexical functions

- Lexical function MAGN denotes high degree
- English

– MAGN (disease)	= grave
– MAGN (fog)	= heavy
– MAGN (control)	= strict

- Russian

– MAGN (болезнь)	= тяжелый
– MAGN (туман)	= густой
– MAGN (контроль)	= строгий

Lexical functions usage

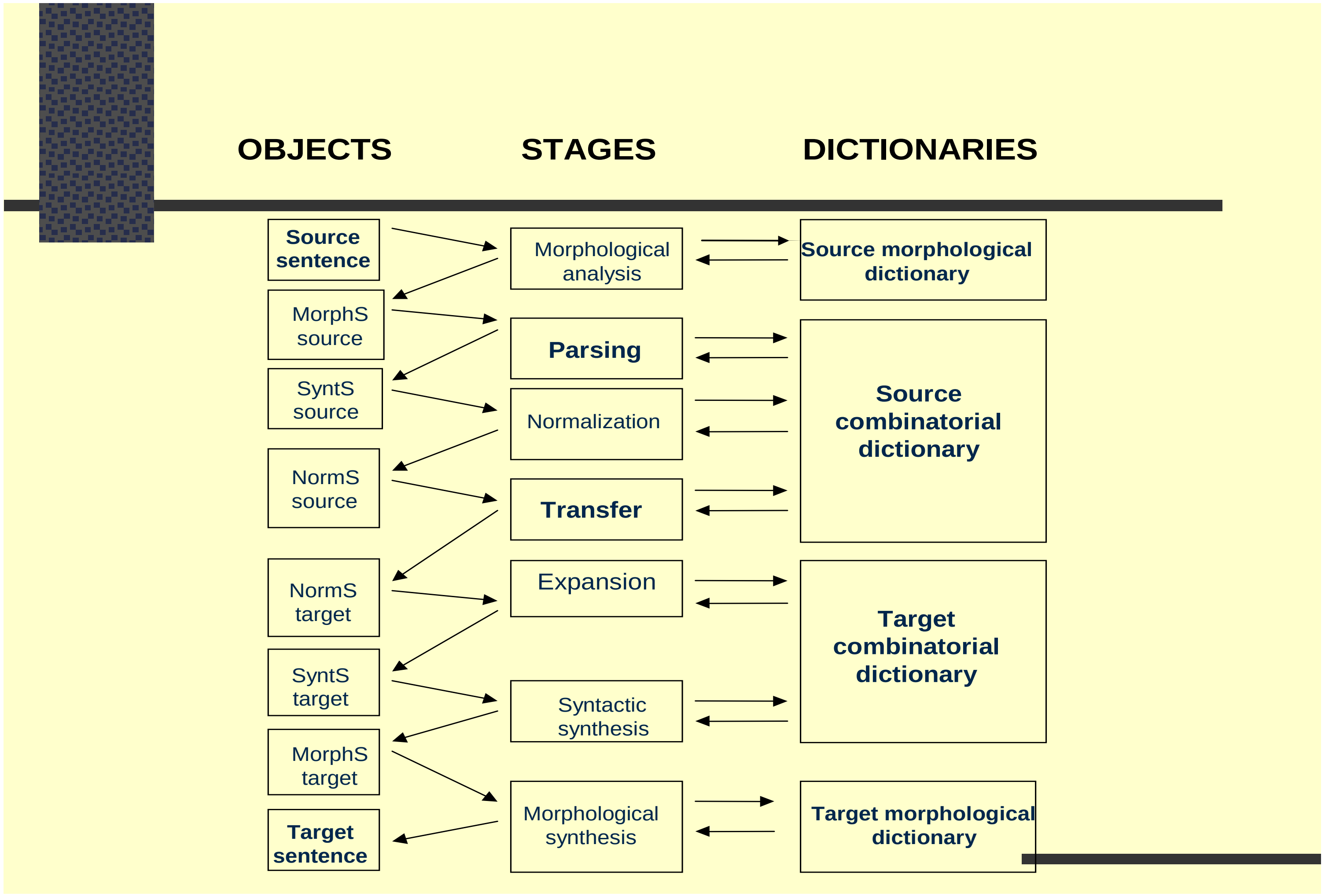
- Idiomatic translation
  - *тяжелая болезнь – grave (~~heavy~~) disease*
  - *густой туман – heavy (~~dense~~) fog*
- Lexical and syntactic disambiguation
  - *Draw a distinction – проводить различие*
  - *The president had the <u>support of the parliament</u>*
  - *The president expressed full <u>support of the parliament</u>*
- Synonymous paraphrasing
  - *He began to observe the rules*
  - *He stopped violating the rules*

FORET

- Language for syntactic transformations
  - Created in 1986. Still improving.
  - It is based on three-valued logic and contains:
    - Predicates to search for syntactic nodes
    - Instructions to modify the syntactic structure

```
CHECK
  1.1 DEP(X,Z,СОЧИН)
  2.2 DOM(X,*,ПРЕДИК)
DO
  1 ADD-NODE(Z1,БЫТЬ)
  2 ADD-FEAT(Z1,[НАСТ,НЕСОВ,ИЗЪЯВ,ЛИЧ])
  3 REPLACE-HEAD([X,*],[Z1,*])
  4 LINK-NODES(Z1,X,ПАСС-АНАЛ)
  5 MOVE-NODE-BEFORE-NODE(Z1,X)
  6 DEL-FEAT(X,[ЗЕРО])
```

Processing steps



Text segmentation

- Tasks
  - Split the text into sentences
  - Split each sentence into words
- Problems
  - Abbreviations with dots (*и т. д., и т. п.*)
  - Words with punctuation marks (*какой – то*)
  - Complex words with spaces (*ни за что, check out*)
  - Dates/numbers processing
  - etc.

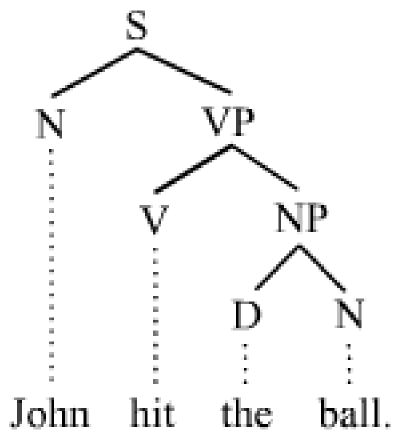
Morphological analysis

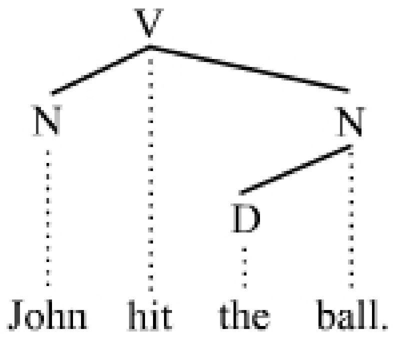
- For each word
  - Find all possible lemmas from the dictionary
  - Along with POS and morphological features
- For the word ‘стали’:

3.1	СТАЛЬ	S, РОД, ЕД, ЖЕН, НЕОД
3.2	СТАЛЬ	S, ДАТ, ЕД, ЖЕН, НЕОД
3.3	СТАЛЬ	S, ПР, ЕД, ЖЕН, НЕОД
3.4	СТАЛЬ	S, ИМ, МН, ЖЕН, НЕОД
3.5	СТАЛЬ	S, ВИН, МН, ЖЕН, НЕОД
3.6	СТАНОВИТЬСЯ1	V, ПРОШ, МН, ИЗЪЯВ, СОВ
3.7	СТАНОВИТЬСЯ2	V, ПРОШ, МН, ИЗЪЯВ, СОВ
3.8	СТАТЬ1	V, ПРОШ, МН, ИЗЪЯВ, СОВ

Syntactic structure

- Syntax is about:
  - How words are connected to each other within a sentence
  - Syntactic structure is always a tree
- Two types of trees
  - Constituent tree
  - Dependency tree
- Example
  - *[John [hit [the ball]]]*
  - *John ←hit ↷ the ← ball*
- ETAP uses dependency trees





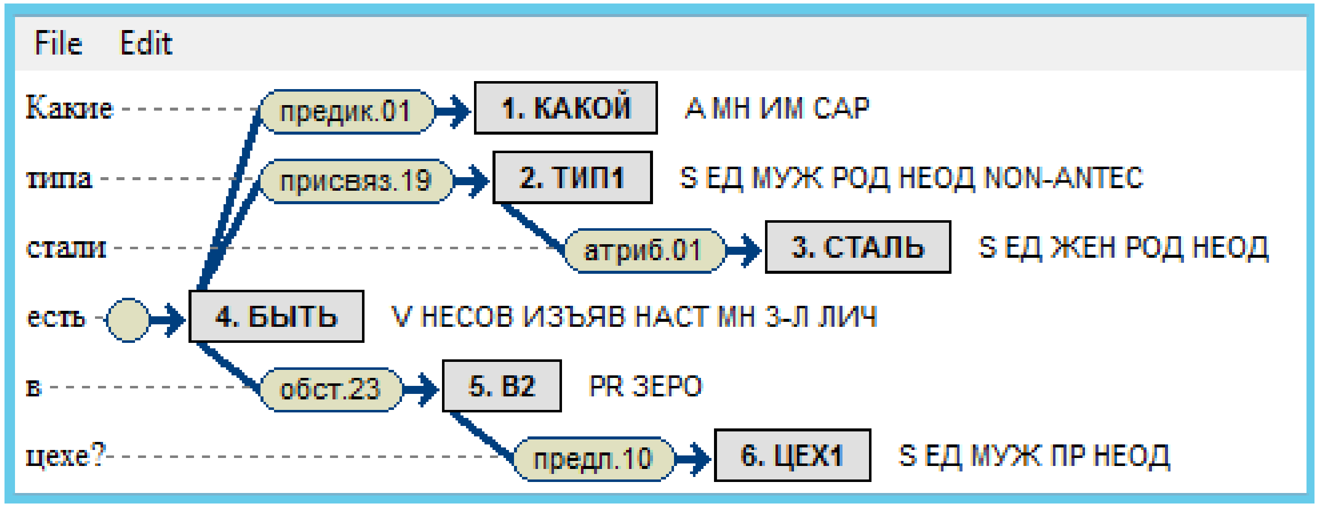
Syntactic analysis

- Steps
  - Identify pairs of words potentially connected to each other
  - Build a full tree out of the pairs of connected words
  - There might be multiple trees successfully built
  - Select the most plausible one
  - Here is where the statistics helps
- Example
  - *Какие типы стали есть в цехе?*

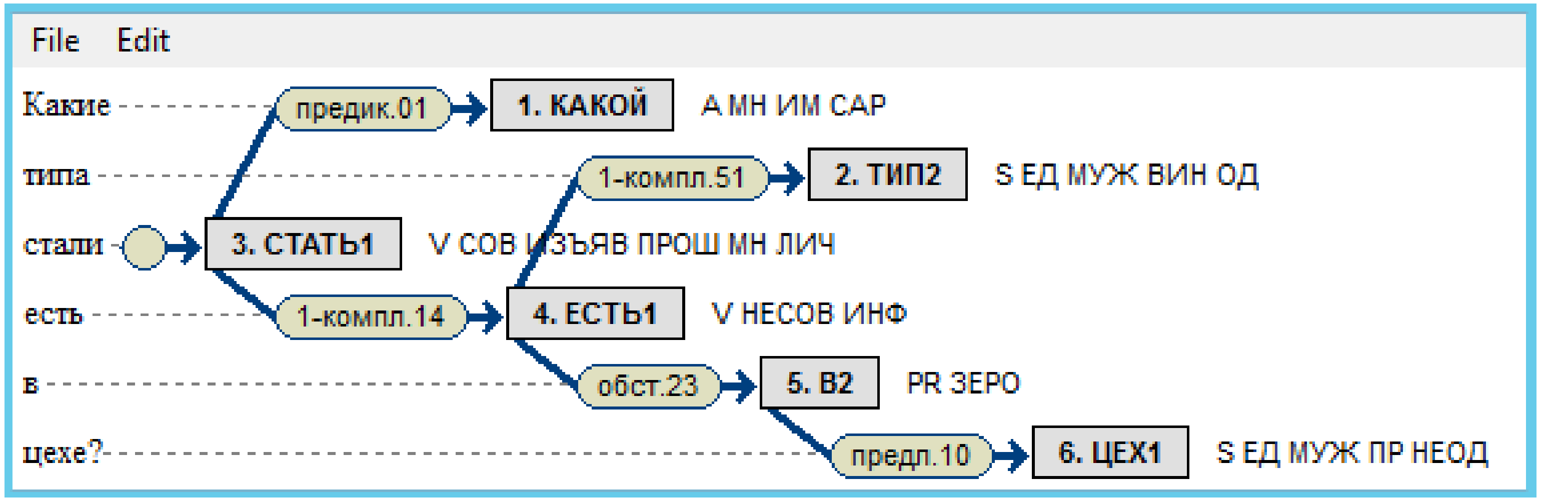
СТАЛЬ	ТИП1	АТРИБ
СТАНОВИТЬСЯ1	ТИП1	ПРЕДИК

Syntactic analysis

- *Какие типы стали есть в цехе?*



```
File   Edit
Какие   предик.01   1. КАКОЙ   A МН ИМ САР
типа    присвяз.19  2. ТИП1    S ЕД МУЖ РОД НЕОД NON-ANTEC
стали   атриб.01    3. СТАЛЬ   S ЕД ЖЕН РОД НЕОД
есть                4. БЫТЬ    V НЕСОВ ИЗЪЯВ НАСТ МН 3-Л ЛИЧ
в       обст.23     5. В2      PR ЗЕРО
цехе?   предл.10    6. ЦЕХ1    S ЕД МУЖ ПР НЕОД
```



```
File   Edit
Какие   предик.01     1. КАКОЙ    A МН ИМ САР
типа    1-компл.51    2. ТИП2     S ЕД МУЖ ВИН ОД
стали                 3. СТАТЬ1   V СОВ ИЗЪЯВ ПРОШ МН ЛИЧ
есть    1-компл.14    4. ЕСТЬ1    V НЕСОВ ИНФ
в       обст.23       5. В2       PR ЗЕРО
цехе?   предл.10      6. ЦЕХ1     S ЕД МУЖ ПР НЕОД
```

SynTagRus

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

Winograd Schema Challenge

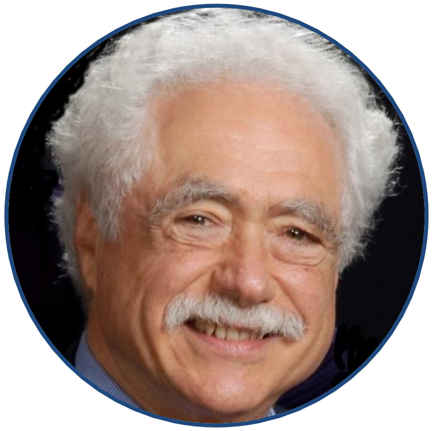
Winograd Schema Challenge

- 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 were 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

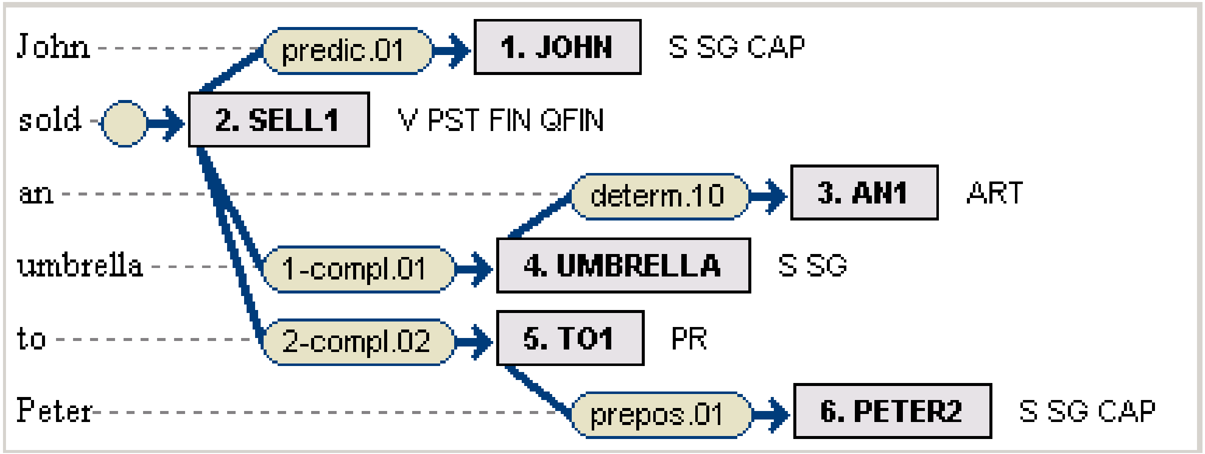
- The ultimate goal
  - To achieve near-human understanding of the text
- How can we measure understanding?
  - The amount of inferences that can be made out of the text
- How can we test inferences?
  - Questioning
- SemETAP is able to answer questions for which there is no direct answer in the original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

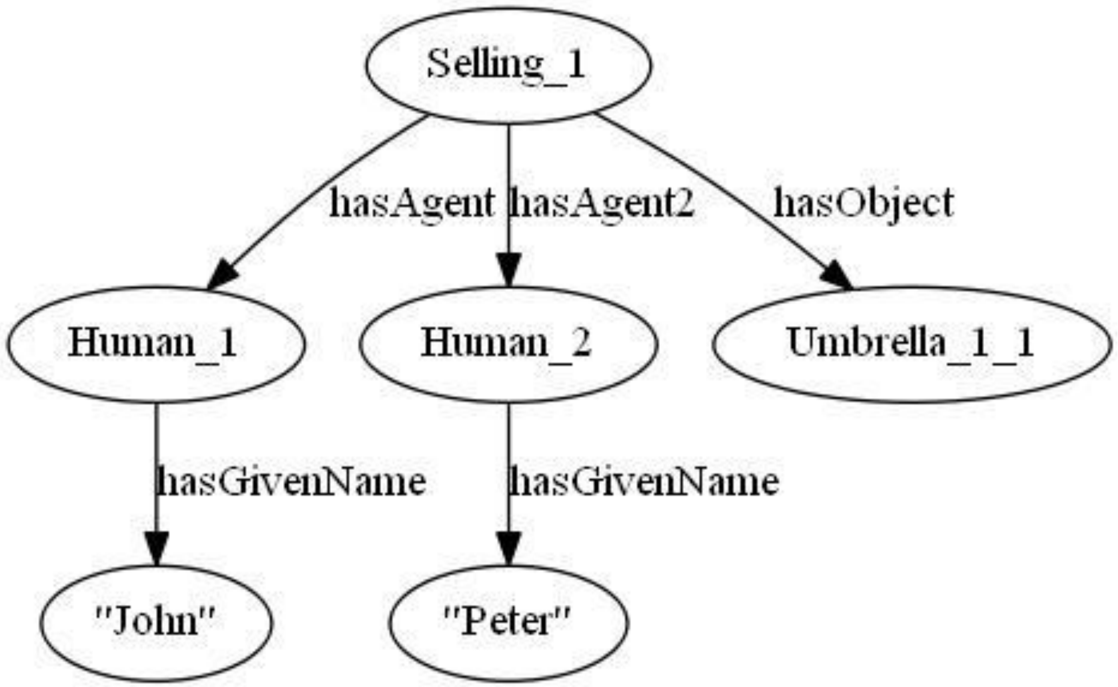
Semantic analysis steps

- Syntactic tree



- Basic semantic structure
  - Words are translated to semantic concepts and syntactic relations to semantic roles based on the dictionary and the ontology
- Enhanced semantic structure
  - Inference rules are applied 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)
```

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

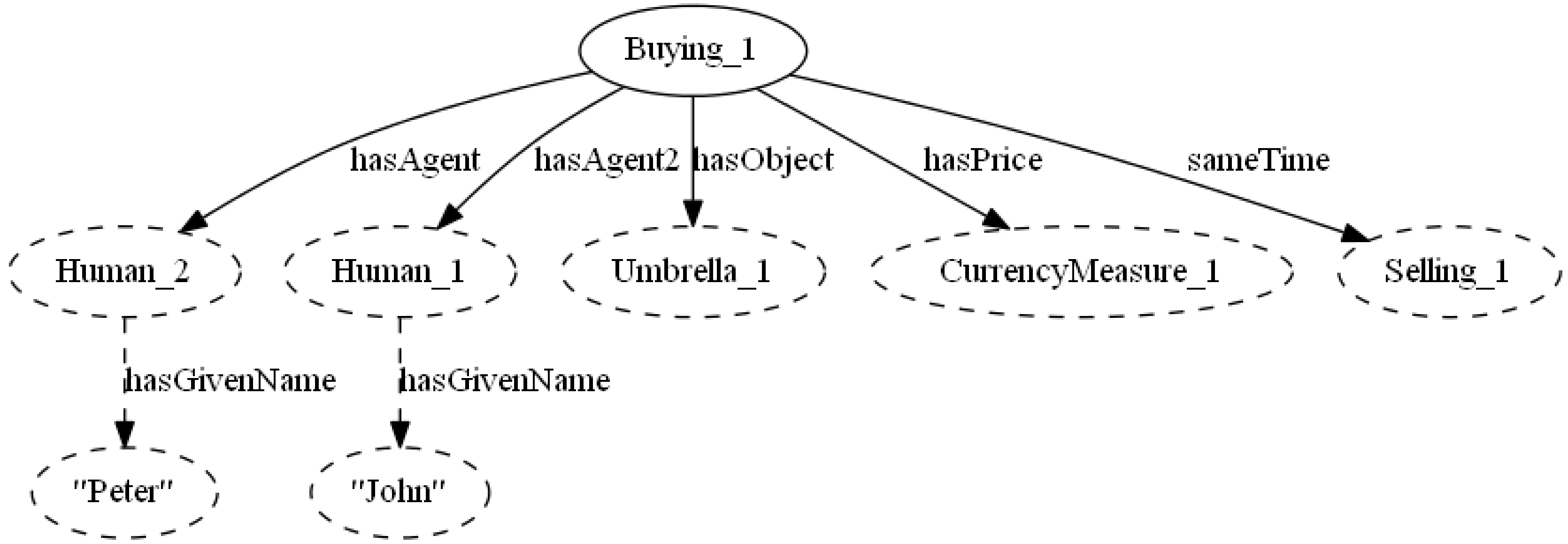
RDF model

- RDF is a Semantic Web standard
  - All data is expressed in the form or triples
  - [subject, predicate, object]
  - Subject denotes a resource, which is identified by IRI (URL)
  - Object can be another resource or a literal
  - *[Bill_Gates, born, "October 28, 1955"]*
  - *[Bill_Gates, married, Melinda_French]*
  - RDF graph is a conjunction of triples
- SPARQL is a SQL-like language to query the graph

Decomposition rule example

```
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
           ).
```

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
```

Case study

- *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*

Case study

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

Rule features

- Declarative rules
  - In the form of implications (if A then B)
  - Similar to Prolog
  - But Prolog does not allow new variables in the rule consequent
  - Datalog is fully declarative variant of Prolog
  - Datalog$^{\pm}$ is an extension of Datalog which allows rules with new variables (so called existential rules or TGDs)
  - Our rules are Datalog$^{\pm}$ compatible

Rule features

- 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

Etalog

- Datalog$^{\pm}$ (semantically) compatible language
  - Supports only unary and binary predicates (RDF model)
- Inference rules written by linguists
  - Requires minimal mathematical background
- Simplifications to make it close to natural language
  - Atoms are written in SPO notation
  - Complex predicates allow to drop repeated subject
  - Complex predicate can be used in place of a variable
  - Variable name can be omitted if it is used only once
- Improves readability and decreases potential errors

Reasoner

- We use RDFox (from Oxford)
  - Very efficient in-memory RDF storage and reasoner
  - Optimized for chase
  - Supports existential rules and custom filters
  - Supports equality (EGDs) through the use of sameAs predicate (either with UNA or 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 if one is 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

Full basic semantic structure

- *John sold an umbrella to Peter*

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

Plausible expectations

- Invited inferences (implicatures)
  - Inferences which are most likely true based on the all the information we have so far
  - *John was allowed to smoke => John smoked (probably)*
  - 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
  - *Where did he buy an umbrella? Probably, from Peter.*
  - *John bought an umbrella. Who owns it? Probably, John.*
- Disjunctive uncertainty
  - Desired: allow disjunction in the rule consequent
  - *X wants Y* – Y is either an event that X wish to happen or a thing that X want to have.

Desired features

- Negation
  - Supported: on the level of individuals – Negation concept
  - Desired: on the level of propositions/atoms/triples
  - *John bought the umbrella not from Peter*
- Universal quantifier
  - Desired: at least in the context of negation
  - *John did not bought an umbrella [from anybody]?*
  - *Did John both un umbrella from Peter?*
- Plurality support
  - *Three men came*

Desired features

- Limitations
  - Limited expressiveness of RDF model
  - Complexity of reasoning algorithms

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

Thank you
for your attention!