

**NATIONAL RESEARCH UNIVERSITY
HIGHER SCHOOL OF ECONOMICS**

Faculty of Computer Science
Bachelor's Programme 'HSE University and University of London Double Degree
Programme in Data Science and Business Analytics'

Software Project Report (Final)

on the topic “Ontology-Controlled Parsing of Assignment Texts with Defaults”

Fulfilled by the Student:

group #БПАД 193


Signature

Dzhkha Anika

Surname, First name, Patronymic, if any

29.05.2021

Date

Checked by the Project Supervisor:

Neznanov Alexey Andreevich

Surname, First name, Patronymic (if any), Academic title (if any)

Job

FCS NRU HSE

Place of Work (Company or HSE Department)

Date May 27 2021

7 (seven)

Grade according
to 10-point scale


Signature

Moscow 2021

Contents

Contents	2
1. Key terms and definitions	4
2. Introduction	6
2.1. Short Description of Subject Field	6
2.2. Relevance of Problem	6
2.3 Objectives	6
2.4 Tasks	6
3. Review and Comparative Analysis of Sources and Analogues	7
4. Physics terms and important notions	7
4.1. Kinematic tasks	7
4.2. Sample of a Problem with Defaults	8
4.3. Defaults:	9
5. Review of task model	10
5.1. Work with JSON	10
5.2. NLP Technologies and Methods	11
5.3. Techniques and tools	12
5.4. Text parsing method - dependency tree	12
5.5. Excel tables	13
5.5.1. Table Description	13
5.5.2. Next Stage of Excel development:	15
6. Natasha tool and parsing model realization	16
6.1. Functionality	16
6.2. SlovNet, Razdel and other libraries	16
6.3. Using the functionality in my project	19
6.4. Description of the initial template of the morphological and syntactic parsing	20
6.4.1. Description of the morphological parsing model:	20
6.4.2. Description of the syntax parsing model:	21
6.5. Description of my model	22
6.6. Result of my project	24
7. Bibliography	25
8. Additional requirements of Mentor/Supervisor	26
9. Project Works' Calendar	27

1. Key terms and definitions

Computational linguistics (CL) – the scientific and engineering discipline concerned with understanding written and spoken language from a computational perspective, and building artifacts that usefully process and produce language.

Sections of CL:

- Syntax,
- Morphology
- Semantics

Natural language – any language used to communicate with people and not created purposefully.

Natural Language Processing (NLP) – a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data.

The development of a linguistic processor for some applied task involves a formal description of the linguistic properties of the processed text, which can be considered as a **text model**.

Applications of CL:

- **Information Retrieval**
- Information Extraction
- Knowledge Discovery
- Categorization
- Automatic text generation

Parsing, or syntactic analysis, - the process of analyzing sentence structure and representing it according to some syntactic formalism.

Tokenization - is NLP task which represents a way of separating a piece of text into smaller units called tokens.

Stemming – a method of preprocessing that gets rid of the unneeded parts of a word while keeping its root (original word “troubled” – stemmed word “troubl”).

Lemmatization - an algorithm that replaces a word by its most basic form (original “studies” – lemma “study”).

JSON - an open standard file format, and data interchange format, that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and array data types.

Tomita-parser - is a parser designed to extract structured data from natural language text. Facts are extracted using context-free grammars and keyword dictionaries.

Ontology - a way of showing the properties of a subject area and how they are related, by defining a set of concepts and categories that represent the subject.

2. Introduction

2.1. Short Description of Subject Field

The programming model is based on Python programming language using JSON techniques, Natasha parser instruments, and some other instruments, so that the model should be able to read the texts of training tasks, identify relationships between subjects in the presence of defaults, and solve physical problems using specified algorithms. For this project there is also an article with NLP tools and modelling methods overview and model descriptions.

2.2. Relevance of Problem

The development of tools for working with educational materials, on the one hand, and distance learning using the latest technologies, on the other hand, has updated the tasks of converting the text of the educational task into a model based on ontology and generating the text of the task from the model.

The project focuses on the important sub-task of converting the task text into a model, due to the fact that the majority of training tasks actively use defaults of various types: from the training topic to implicit references to implied conditions. This important subproblem rests simultaneously on the need for natural language text analysis and ontological modeling. The project is also part of another, more extensive project that is being conducted together with experts from the HSE Institute of Education.

2.3 Objectives

The main goal is to implement the transformation of texts of educational tasks in physics into a model based on specialized ontologies of the subject area and the educational process.

2.4 Tasks


- 1) Study and make an overview of materials about Natural Language Processing tools with a focus on identifying key phrases and identifying relationships between entities.
- 2) Make an overview of knowledge representation and modelling methods in the field of kinematics and dynamics.
- 3) Prototype the task text parser with access to ontologies and default descriptions.
- 4) Create a module for a software system to support the adaptive learning process, including a simulator for school physics.
- 5) Train the created module on kinematics and dynamics problems.

3. Review and Comparative Analysis of Sources and Analogues

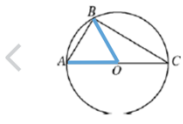
Currently we can have a look at the one interesting source that solves high school geometry questions: <https://geometry.allenai.org>.

Here we can see not only the answer for the particular tasks, but also the information that machine has extracted from the text and how it was reading and processing this information.

GeoS Demo — An End to End Geometry Problem Solver



In the figure above, triangle ABC is inscribed in the circle with center O and diameter AC. If $AB=AO$, what is the degree measure of angle $\angle ABO$?



(A) 15°

(B) 30°

(C) 45°

(D) 60°

(E) 90°

IsInscribedIn(ABC, circle):	0.71	Is(DiameterOf(circle), AC):	0.99
IsInscribedIn(ABC):	0.64	IsInscribedIn(ABC):	0.99
IsCenterOf(O, ABC):	0.41	IsDiameterLineOf(diameter, circle):	0.99
Is(What, MeasureOf(ABO)):	0.21	IsDiameterLineOf(AC):	0.99
IsDiameterLineOf(diameter, circle):	0.19	IsInscribedIn(ABC, circle):	0.99
IsCircle(circle):	0.03	IsDiameterLineOf(diameter):	0.99
Is(AC, diameter):	0.01	IsDiameterLineOf(AC, circle):	0.99
Is(diameter, AC):	0.01	IsCenterOf(O, ABC):	0.34

GeoS extracted the formulas above. Hover over each for more information.

Next

This system is a great sample and analogy in geometry, so this resource was taken into account.

However, the goal of this project is to work with problems in physics and with defaults.

4. Physics terms and important notions

4.1. Kinematic tasks

The following terms are components of the model of the kinematic tasks:

1) **World** - general description of the task context.

a. Scene - formalization of the visualization of the world in a specific delivery environment.

b. Force field - formalization of the effects on actors from the world as a whole.

2) Actor - moving or resting object. The actor can be a point or extended along some axes.

a. Binding to the scene.

3) Motion element (ME) - movement of a specific actor for a specific time with a specific law of motion (the law of motion is determined by the type of ME).

Description of ME:

a. Type of ME.

b. State of ME.

c. Conditions.

d. Formula.

e. Current activity (inclusiveness).

4.2. Sample of a Problem with Defaults

“Из одного города в другой вышел пешеход. Когда он прошел 27 км, вслед за ним выехал автомобиль со скоростью в 10 раз большей, чем у пешехода. Второго города они достигли одновременно. Чему равно расстояние между городами?”

Translation:

“A pedestrian walked from one city to another. When he walked 27 km, a car followed him at a speed 10 times faster than that of a pedestrian. They reached the second city at the same time. What is the distance between cities?”

Defaults are the written (or unwritten) words whose meaning should only be guessed from the context. In this task such defaults are: he (a pedestrian), followed him (from the same city), at a speed 10 times faster than that of a pedestrian (a speed of pedestrian is 10 times less than a speed of the car), they reached the second city at the same time (the car meet the pedestrian on $27 + n$ km and this is the distance between cities), and the type of this task - Uniform Rectilinear Motion

For a person who read this task it should be clear that:

1) World - Earth: a. Scene - road;

2) Actors are the pedestrian and the car

3) MEs : type - uniform rectilinear motion, state of the pedestrian $l = 27$ km, state of

the car $l = 0$ km, condition of finish = the car meet the pedestrian on $27 + n$ km and this is the distance between cities, condition 2 is that car velocity = 10 times of the pedestrian velocity, formula = condition of finish.

For a machine it is a complicated task to read without special techniques and instruments, so I need to create a parser that is able to perform such operations, extract needed facts and put it to some model.

4.3. Defaults:

The default I found personally by hand using the conditions of these physical problems. The defaults are different for each type of problem.

For example:

To determine the velocity of the flow of water, a float is lowered into a river, which in 50 seconds passes 60 meters between two milestones. Assuming the speed of the float is equal to the speed of flow, determine the speed of flow of the water.

The defaults for the problem:

(ME or actor's features) - Position_start = 0;

(type of ME) - uniform rectilinear

(measurement units) - expressed in SI(m/s)

(the movement direction) - (other features) - one way

And so, with these defaults you can already think much more broadly about problem solutions and point to solution patterns, for this I made up default types for different kinds of problems and later did the same for other data in the problems (See point 5.5.1 Page 12).

5. Review of task model

5.1. Work with JSON

This is how the task should be described in JSON based on the model that described earlier:

```
task1.js > No Selection
1 var data = {
2   rawText: `Расстояние между двумя населенными пунктами 120 км. Автобус преодолевает это расстояние,
3   двигаясь со средней скоростью 40 км/ч, а автомобиль – со средней скоростью 60 км/ч.
4   На сколько часов пассажиры автобуса находятся в пути дольше, чем пассажиры автомобиля?`,
5   actors: [
6     {
7       id: "uuid-a1",
8       name: "actor1",
9     },
10    {
11      id: "uuid-a2",
12      name: "actor2",
13    }
14  ],
15  actorModels: [
16    {
17      id: "uuid-blue-car",
18      name: "Машина синяя",
19      model: "BLUE_CAR.fbx",
20      type: [
21        {
22          id: "uuid",
23          name: "Любой наземный транспорт"
24        },
25        {
26          id: "uuid",
27          name: "Любой автомобиль"
28        }
29      ]
30    },
31    {
32      id: "uuid-blue-bus",
33      name: "Автобус синий",
34      model: "BLUE_BUS.fbx",
35      type: [
36        {
37          id: "uuid",
38          name: "Любой наземный транспорт"
39        },
40        {
41          id: "uuid",
42          name: "Любой автомобиль"
43        }
44      ]
45    }
46  ],
47  world: {
48    name: "Земля",
49    title: "EARTH",
50    dim: 1,
51    parameters: [
52      {
53        name: "g",
54        MU: {
55          name_short: "Н * м^2/кг^2"
56        }
57      }
58    ],
59    model: {
60      id: "uuid",
61      scene: "ROAD_TWO_WAY",
62      type: [
63        {
64          id: "uuid",
65          name: "2 параллели"
66        },
67        {
68          id: "uuid",
69          name: "Дорога"
70        }
71      ]
72    },
73    motionElements: [
74      {
75        name: "motion1",
76        actor: "uuid-a1",
77        model: "uuid-blue-car",
78        mainFormulas: [
79          {
80            formula: "motion1_position_x = motion1_startPosition_x + (time - motion1_startTime) * motion1_velocity * motion1_direction_x",
81            parameters: [
82              {
83                name: "motion1_position_x",
84                MU: {
85                  name_short: "км"
86                }
87              }
88            ]
89          }
90        ]
91      }
92    ]
93  }
94 }
```

```

task1.js > No Selection
93     MU: {
94       name_short: "KM"
95     },
96   },
97   {
98     name: "motion1_startPosition_x",
99     MU: {
100       name_short: "KM"
101     },
102   },
103   {
104     name: "time",
105     MU: {
106       name_short: "ч"
107     },
108   },
109   {
110     name: "motion1_startTime",
111     MU: {
112       name_short: "ч"
113     },
114   },
115   {
116     name: "motion1_velocity",
117     MU: {
118       name_short: "KM/ч"
119     },
120   },
121   {
122     name: "motion1_direction_x",
123     MU: {
124       name_short: "vector"
125     },
126   },
127 ]
128 },
129 ],
130
131 startFormulas: [
132   {
133     formula: "time = 0",
134     parameters: [
135       {
136         name: "time",
137         MU: {
138           name_short: "ч"
139         },
140       }
141     ]
142   }
143 ]

```

It is just a part of one described task, which will help me to understand how to connect the code with the text and task model. This example you could see on my github account. It should be extended and be more specific, for ex. there is nothing about defaults. And this almost handwritten (the ontology was used) model of the result that I should achieve using my parser.

5.2. NLP Technologies and Methods

Applied computational linguistics is largely equivalent with Natural Language Processing (NLP). The main challenge of the CL is the fact that Natural Language (NL) is a complex multi-level system of signs that has arisen for the exchange of information between people, developed in the process of human practice, and is constantly changing in connection with this activity. Another difficulty of developing CL methods (and the difficulty of studying NL within linguistics) is associated with the variety of natural languages, significant differences in their vocabulary, morphology, syntax, and different ways of expressing the same meaning.

The key challenge regarding the CL that we are going to consider here is **text processing**

with an emphasis on **identifying key phrases and identifying relationships** between them. This problem includes other well-known technologies of NLP, such as Text Mining (which is a part of Data Mining field), **Information Extraction** (IE), Information Retrieval(IR) and Knowledge Discovery(KD). In Text Mining we can use Categorization(also an application of CL) in many ways. For thematic categorization we focus on noun terms that can characterize a topic.

5.3. Techniques and tools

<https://github.com/AnikaJha/Ontology-controlled-parser-of-textbooks>

Text documents typically contain many words that are not needed to grasp the basic idea of the text, but what to do to eliminate useless data without losing semantics of text. To group similar words into one, we reduce words to their stem, or root form – this approach is called stemming (**or lemmatization**). For example, “walking”, “walk”, “walked”, and “walker” will all be reduced to the root word “walk”.

Stemming works faster than lemmatization, but this method often does not give satisfactory results for the Russian language. Lemmatization is a more smart operation and requires more computational resources, but it does not give perfect results, because Russian has a fairly large number of homonyms and even people find it difficult to identify the correct meaning of these words.

We can use spelling correctors and acronyms and abbreviation expanders to minimize extra info.

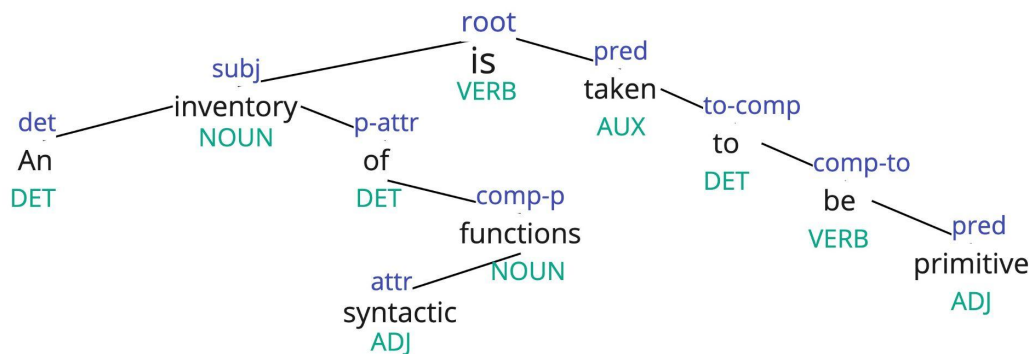
There is also such a problem as the **word sense disambiguation** (WSD) that deals with finding the most probable sense of a word with multiple meanings. **Tokenization** is one more common task which represents a way of separating a piece of text into smaller units called tokens.

5.4. Text parsing method - dependency tree

Dependency tree:

- 1)Dependency tree is a directed graph that satisfies the following constraints:
- 2)With the exception of the root node, each vertex has exactly one incoming arc.
- 3)There is a unique path from the root node to each vertex in V.

There is a top-level node - the predicate. From it you can reach any word. Each word depends on the other, but only one. The dependency tree looks something like this:



An inventory of syntactic functions is taken to be primitive

miro

In this tree, the edges are signed by some special type of syntactic relation. Dependency grammar analyzes not only the fact of a relationship between words, but also the nature of that relationship. For example, "is taken" is almost one verb form; "inventory" is the subject for "is taken. Accordingly, we have an edge from "is" in both directions. They are not the same relation, they are of a different nature, so they must be distinguished.

Here and below we consider simple cases where sentence members are present rather than implied. There are structures and marks to deal with omissions. Something appears in the tree that has no surface expression: words. But this is the subject of another study, and we still need to focus on our own.

5.5. Excel tables

5.5.1. Table Description

Tasks_kinematics

As said above, I use an excel table to input and output data to and from the project. More specifically: The table is used to train the parser on the input data.

The first important thing I get from the table are the "what to find" references and description of the properties of motion elements .To do this, I reviewed and manually

processed a large number of examples of kinematics problems, and described the most popular types of actors, highlighted the properties of motion elements and options for what to find. To do this, I reviewed and manually processed a large number of examples of kinematics problems, and described the most popular types of actors (they are often repeated), highlighted the properties of ME and variants of the names of what to find. I did not describe some obvious things in order to focus on the important.

Равномерное прямолинейное движение			Элемент движения (motion element)					ответ		
	Actors		Тип	Состояние начала	Условия окончания	Формула	Тек активность	откуда искать вопрос	что найти	
Автомобиль движется со скоростью 15 м/с. Выразите эту скорость в км/ч, дм/мин.	Автомобиль		по умолчанию	v_start = 15 м/с	null			выразите	эту скорость	скорость
Для определения скорости течения воды в реку опущен поплавок, который за 50 с проходит 60 м между двумя весами. Принимая скорость поплавка равной скорости течения, определите скорость течения воды.	поплавок	течение	по умолчанию		t_end=50с; s_end=60м	s=vt		определите	скорость	скорость
Самолет пролетает 100 км за 5 мин. Определите скорость самолета в м/с и км/ч.	Самолет		по умолчанию					определите	скорость	скорость
Что имеет большую скорость: самолет, пролетающий за час 1200 км, или пуля винтовки, вылетающая со скоростью 760 м/с?	самолет	пуля винтовки	по умолчанию					что имеет	большую скорость	скорость
За сколько минут плывущий по реке плот пройдет расстояние 150 м, если скорость его движения 0,5 м/с?	плот		по умолчанию	v_start=0,5м/с	s=150м			За сколько	минут	время
Расстояние между двумя населенными пунктами 120 км. Автобус преодолевает это расстояние, двигаясь со средней скоростью 40 км/ч, а автомобиль - со средней скоростью 60 км/ч. На сколько часов пассажиры автобуса находятся в пути дольше, чем пассажиры автомобиля?	Автобус	автомобиль	по умолчанию					на сколько	часов	время
С некоторого момента времени парашютист стал спускаться равномерно со скоростью 5 м/с. Двигаясь с такой скоростью, он за 5 мин достиг земли. Какой путь преодолел парашютист за это время?	парашютист		равномерно					какой	путь	путь
Автобус в течение первого часа двигался со скоростью 60 км/ч, а в течение второго часа - 80 км/ч. На сколько километров больше составил путь автобуса за второй час движения, чем за первый?	Автобус		по умолчанию					на сколько	километров	путь
Пешеход за минуту делает 100 шагов. Определите скорость движения пешехода, считая длину шага равной 80 см.	Пешеход							определите	скорость	скорость
Автомобиль двигался со скоростью 40 км/ч в течение 30 мин, а следующие 0,5 часа со скоростью 60 км/ч. Какой путь прошел автомобиль за все время движения?	Автомобиль							Какой	путь	путь
Мотоцикл за первые два часа проехал 90 км, а следующие 3 часа двигался со скоростью 50 км/ч. Какой была скорость мотоциклиста на первом участке пути? Какой путь он прошел за все время движения?	Мотоцикл		по умолчанию	s_start=90км; t_start=2ч;	t_end=t_start+3ч; v_end=50км/ч;			Какой	скорость	скорость, путь

The next important step in this table is the description of the defaults in the tasks. For example, during the description of the properties of the elements of motion, I found that the default can be set that we are considering the problem of uniform rectilinear motion, if there is no indication of accelerations or other indications. Also, an important observation was that the number of elements of motion is a default that we usually define ourselves by.

Равномерное прямолинейное движение	element)		ответ			умолчания			
	Формула	Тек активность	откуда искать вопрос	что найти		ЭД(или свойства актора)	тип эд	единицы измерения	другое
Автомобиль движется со скоростью 15 м/с. Выразите эту скорость в км/ч, дм/мин.			выразите	эту скорость	скорость	Position_start = 0; направление - одно(вектор+)	равномерное прямолинейное	выразить не в СИ	
Для определения скорости течения воды в реку опущен поплавоч, который за 50 с проходит 60 м между весами. Принимая скорость поплавок равной скорости течения, определите скорость течения воды.	$s=vt$		определите	скорость	скорость	Position_start = 0;	равномерное прямолинейное	выразить в СИ(м/с)	$t1=t2=answer$
Самолет пролетает 100 км за 5 мин. Определите скорость самолета в м/с и км/ч.			определите	скорость	скорость	Position_start = 0;	равномерное прямолинейное		ответ - актор
Что имеет большую скорость: самолет, пролетающий за час 1200 км, или пуля винтовки, вылетающая со скоростью 760 м/с?			что имеет	большую скорость	скорость	Position_start = 0;	равномерное прямолинейное		
За сколько минут плывущий по реке плот пройдет расстояние 150 м, если скорость его движения 0,5 м/с?			За сколько	минут	время	Position_start = 0;	равномерное прямолинейное	выразить не в СИ	
Расстояние между двумя населенными пунктами 120 км. Автобус преодолевает это расстояние, двигаясь со средней скоростью 40 км/ч, а автомобиль - со средней скоростью 60 км/ч. На сколько часов пассажиры автобуса находятся в пути дольше, чем пассажиры автомобиля?			на сколько	часов	время		равномерное прямолинейное		пассажиры автобуса>автобус; пассажиры автомобиля = автомобиль
С некоторого момента времени парашютист стал спускаться равномерно со скоростью 5 м/с. Двигаясь с такой скоростью, он за 5 мин достиг земли. Какой путь преодолел парашютист за это время?			какой	путь	путь	это время= $t_{end}-t_{start}$	равномерное прямолинейное		
Автобус в течение первого часа двигался со скоростью 60 км/ч, а в течение второго часа - 80 км/ч. На сколько километров больше составил путь автобуса за второй час движения, чем за первый?			на сколько	километров	путь	$s2=s_{end}-s1$	равномерное прямолинейное		
Пешеход за минуту делает 100 шагов. Определите скорость движения пешехода, считая длину шага равной 80 см.			определите	скорость	скорость	$s_{end}=k \cdot t$	равномерное прямолинейное		
Автомобиль двигался со скоростью 40 км/ч в течение 30 мин, а следующие 0,5 часа со скоростью 60 км/ч. Какой путь прошел автомобиль за все время движения?			Какой	путь	путь			разные единицы измерения	
Мотоцикл за первые два часа проехал 90 км, а следующие 3 часа двигался со скоростью 50 км/ч. Какой была скорость мотоциклиста на первом участке пути? Какой путь он прошел за все время движения?			Какой	скорость	скорость, путь	Position_start = 0; 2 ЭД; $v_{start}=v_{end}$; направление вектор +			
Поезд в течение одного часа шел со скоростью 20 м/с, затем еще 3 ч со скоростью 36 км/ч, а длина последнего участка пути составила 20 км. Какой путь прошел поезд?			Какой	путь	путь	3 ЭД;		разные единицы измерения	

5.5.2. Next Stage of Excel development:

The tables describe three types of physics problems:

Uniform rectilinear

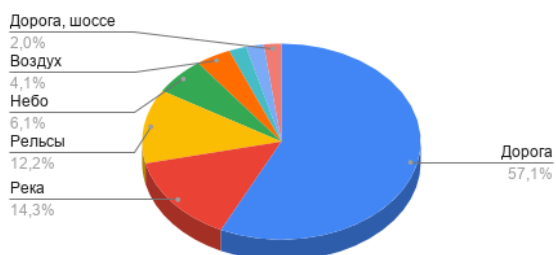
Motion with acceleration

Equal-velocity motion.

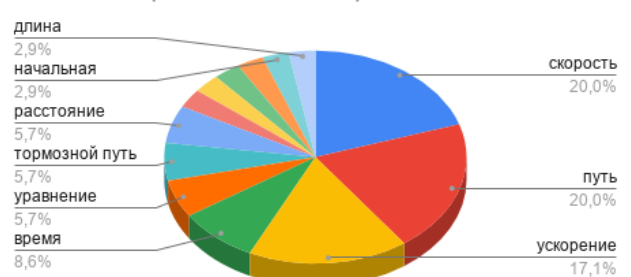
I have updated and completed the tables. I made Scenes and Actors, Words Indicators, which indicate the beginning of a question. Possible unknowns and most popular defaults.

Then I made summary tables of this data(for 60 problems) that provide statistics of the most popular data.

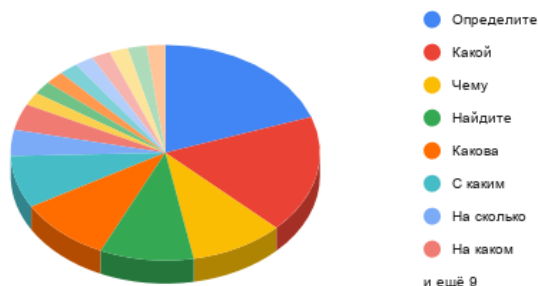
Сцена (scene)



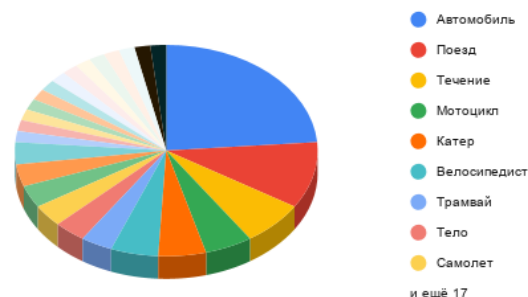
Что найти (the X - unknown)



Откуда искать вопрос (Start of the question)



Акторы (Actors)



It gave me an understanding of the statistics of building tasks based on data.

6. Natasha tool and parsing model realization

6.1. Functionality

Natasha is a set of quality open source tools for natural Russian language processing (NLP). Natasha library unites them under one interface, and solves the basic tasks of natural Russian language processing: segmentation into tokens and sentences, morphology and syntax analysis. All solutions show top results in news topics, and are fast on CPUs.

Natasha is similar to other combinatorial libraries: SpaCy, UDPipe, Stanza. SpaCy initializes and calls models implicitly, the user passes text to NLP function, and gets a fully parsed document. It is also an analogue of the tomite parser.

6.2. SlovNet, Razdel and other libraries

The Razdel library, part of the Natasha project, divides the Russian-language text into tokens and sentences.

The speed and quality of Razdel is comparable or higher than other open source solutions for Russian.

Segmentation parsers	Errors on 1000 tokens	Время обработки, секунды
NLTK	130	3.1
Moses	11	1.9
RuTokenizer	15	1.0
Razdel	7	2.6

- ❑ Why do you need **Razdel** at all, when a similar quality is given by baseline with regularochka and there are plenty of ready-made solutions for the Russian language? Actually, Razdel is not just a tokenizer, but a small rule-based segmentation engine. Segmentation is a basic task, often encountered in practice. For example, there is a judicial act, you need to isolate the operative part of it and divide it into paragraphs. Naturally, ready-made solutions do not know how to do that.
- ❑ **SlovNet** is a Python library for deep-learning based NLP modeling for Russian language. Library is integrated with other Natasha projects. Natasha Slovnet's project deals with training and inference of modern models for Russian-language NLP. The library contains high-quality compact models for extracting named entities, parsing morphology and syntax. The quality on all tasks is comparable or superior to other

open-source solutions for Russian on news texts.

- ❑ **Yargy-parser** is an analogue of Yandex Tomita-parser for Python. Entity extraction rules are described using context-free grammars and dictionaries. The Yargy Parser is the most comprehensible and flexible tool at the moment when it is necessary to create new unique rules for data extraction.

- ❑ **Tomita Parser** is a tool for extracting structured data (facts) from natural language text. Fact extraction is performed using context free grammars and keyword dictionaries. (In the minimum configuration, the parser inputs the text to be analyzed, as well as the dictionary and grammar. The volume of the dictionary and the complexity of the grammar depend on the purposes of analysis: they can be both very small and huge. The grammar file consists of templates written in the internal Tomita-parser language/formalism. These templates describe, in generalized form, the chains of words that can be encountered in a text. In addition, the grammars define exactly how the extracted facts should be presented in the final output. But there is a problem that Tomita Parser does not build properly from the official Yandex website. The problem is still not solved, which is why I switched to Natasha and Yargy Parser)

- ❑ **Ipymarkup** is a primitive library, needed for highlighting substrings in text and making a visualization. Installation instructions and examples of usage are in the Ipymarkup repository. The library is similar to displaCy and displaCy ENT, invaluable for debugging grammars for Yargy parser.

- ❑ **Navec** library is a part of the Natasha project, a collection of pre-built embeddings for the Russian language. According to intrinsic quality metrics, they slightly fall short of the top RusVectores solutions, but the size of the archive with weights is 5-6 times smaller (51MB), the dictionary is 2-3 times larger (500K words).

6.3. Using the functionality in my project

Natasha's libraries are the basis of my code. I use the libraries to do tokenization, morphology and syntax analysis, and for lemmatization.

Tokenization is the splitting of text into smaller parts, tokens. Tokens include both words and punctuation marks. Quite often the task is to present the text as an array of meaningful words. Then after tokenization we can make a cleanup for punctuation marks and irrelevant words (for example, prepositions).

The screenshot shows the PyCharm IDE interface. The main editor window displays a Python script named 'main.py'. The code defines a function 'sentenize' which takes a string 'text' as input and returns a list of tokens. It uses 'tokenize.sent_tokenize' to split the text into sentences and 'tokenize.word_tokenize' to split each sentence into words. A list comprehension is used to flatten the nested lists of tokens from all sentences. The variable 'text' is assigned a Russian sentence about water flow speed determination. The output of the function is printed. The Run console at the bottom shows the command executed and the resulting list of tokens, confirming successful execution.

```
pythonProject6 main.py  
from razdel import sentenize, tokenize  
  
text = 'Для определения скорости течения воды в реку опущен поплавок, который за 50 с проходит 60 м между двумя вежами. Принимая скорость поплавок равной скорости тече  
chunk = []  
for sent in sentenize(text):  
    tokens = [_.text for _ in tokenize(sent.text)]  
    chunk.append(tokens)  
print(chunk[:2])
```

Run: main -
[['Для', 'определения', 'скорости', 'течения', 'воды', 'в', 'реку', 'опущен', 'поплавок', ',', 'который', 'за', '50', 'с', 'проходит', '60', 'м', 'между', 'двумя', 'веж']]
Process finished with exit code 0

“Для определения скорости течения воды в реку опущен поплавоч, который за 50 с проходит 60 м между двумя вехами. Принимая скорость поплавок равной скорости течения, определите скорость течения воды.” - We can see how this sentence is divided into smaller components, tokens. This brings us to the syntactic and morphological parsing step.

For **Morphological** and **Syntactic parsing** I use the SloveNet and Razdel libraries.

In them I use the functions:

- ❑ `load()` - Loading from a data file
- ❑ `navec()` - pre-trained embedding for splitting words into trees for parsing
- ❑ `markup()` - a tool to represent a program language in html format, made for parsing.

- ❑ map - connects a word with its morphological description

```
/Users/anikajha/PycharmProjects/pythonProject6/venv/bin/python /Users/anikajha/PycharmProjects/pythonProject6/main.py
[[ 'Для', 'определения', 'скорости', 'течения', 'воды', 'в', 'реку', 'опущен', 'поплавок', ',', 'который', 'за', '50', 'с', 'проходит', '60', 'м', 'между', 'двуамя', 'вехами', '.' ]]
```

(“slovnet_morph_news_v1.tar” and “slovnet_syntax_news_v1.tar”) trained on the news bulletins. For my project I will need to rewrite or add notes of the current model template more for the physical quantities, because this model still gives us some mistakes like ‘50 c’ = NUM + ADP(preposition), however ‘c’ in this text is in the context ‘seconds’.

1.1)Amortize Navec to make trees of relations words to each other

- 1.2) Import syntax for morphological parsing.
- 2) Split the sentence into tokens and identify words.
- 3) Load a model from news bulletins, parser runs tokens through the model and reveals the morphological parsing of each.
- 4) Output words from the map with their morphological parsing.

```
from razdel import sentenize, tokenize
from navec import Navec
from ipymarkup import show_dep_ascii_markup as show_markup
from slovnet import Syntax, Morph

text = 'Для определения скорости течения воды в реку опущен поплавок, который за 50 с проходит 60 м между двумя вежами. Принимая скорость поплавок'
chunk = []
for sent in sentenize(text):
    tokens = [_.text for _ in tokenize(sent.text)]
    chunk.append(tokens)
print(chunk[:2])

navec1 = Navec.load('navec_news_v1_1B_250K_300d_100q.tar')
morph = Morph.load('slovnet_morph_news_v1.tar', batch_size=4)
morph.navec(navec1)

markup = next(morph.map(chunk))

words, deps = [], []
for token in markup.tokens:
    print(f'{token.text:>20} {token.tag}')
```

6.4.2. Description of the syntax parsing model:

- 1) from the Partition library, import a token to parse sentences into words, prepositions, etc. and setinize(to run the tokens).

- 1.1) Amortize Navec to make trees of relations words to each other

- 1.2) Import syntax for parsing.

- 1.3) Import ipymarkup to output html parsing model.

- 2) Split a sentence into tokens and identify words.
- 3) Load the model from news bulletins, parser runs the tokens through the model and identifies the parsing of each sentence.
- 4) We output the words through ipmarkuo with their syntactic parsing of sentences (already assembled back), remembering each step (from sentence to sentence) and outputting html model of word relations in a convenient form (Relationship lines do not intersect, which

```
pythonProject6 main.py
1 from razdel import sentenize, tokenize
2 from navec import Navec
3 from ipymarkup import show_dep_ascii_markup as show_markup
4 from slovnet import Syntax
5
6 text = 'Для определения скорости течения воды в реку опущен поплавок, который за 50 с проходит 60 м между двумя вежами. Принимая скорость поплавок'
7 chunk = []
8 for sent in sentenize(text):
9     tokens = [_.text for _ in tokenize(sent.text)]
10    chunk.append(tokens)
11    print(chunk[:2])
12
13 navec1 = Navec.load('navec_news_v1_1B_250K_300d_100q.tar')
14 syntax1 = Syntax.load('slovnet_syntax_news_v1.tar')
15 syntax1.navec(navec1)
16
17 markup = next(syntax1.map(chunk))
```

makes the parsing extremely pleasing to the eye)

6.5. Description of my model

1) I used a new parser, namely the Yargy parser, to work with my tables and extract the main questions from the whole task.

I made my list from the input data to the dictionary and through the parser I checked for them in the text to extract necessary data(Actors, main questions, type of question)

View of my model:

```
from yargy import Parser, rule
from yargy.predicates import gram, dictionary
import csv

startQWords = []
with open('StartWords.csv', newline='') as File:
    reader = csv.reader(File)
    for row in reader:
        startQWords.extend(row)

actors = []
with open('actors.csv', newline='') as File:
    reader2 = csv.reader(File)
    for row in reader2:
        actors.extend(row)

R_1 = rule(dictionary(startQWords), gram('VERB'), gram('NOUN'), gram('NOUN'))
R_2 = rule(dictionary(startQWords), gram('VERB'), gram('NOUN'))
R_3 = rule(dictionary(startQWords), gram('NOUN'), gram('NPRO'), gram('VERB'))
```

```
R_4 = rule(dictionary(actors))
parser = Parser(R_1) or Parser(R_2)
parser3 = Parser(R_3)

def find_x_parser(txt):
    for match in parser.findall(txt):
        print([x.value for x in match.tokens])

    for match in parser3.findall(txt):
        print([x.value for x in match.tokens])

def find_actor(txt):

text = 'Мотоцикл за первые два часа проехал 90 км, а следующие 3 часа двигался со скоростью 50 км/ч. Какой была '\
'скорость мотоциклиста на первом участке пути? Какой путь он прошел за все время движения? '
```

Firstly, I created a csv file and put the word indicators in it. After that I made it as a sheet and went through it using a parser and its rules, for example gram(VERB) gram(NOUN) gram(NOUN) in order to extract the question completely. After that I output it as a sentence.

In this way several main questions were extracted from the problem.

(There is only one problem, that the parser does not work with two-component words such as: how much, how long, etc.

The next step was for us not only to understand the essence of the question, but also to understand the actors in the task, for this I used the same structure as for the definition of the essence of the question, I also created a csv file with actors and ran them through text using the list, so we get the final actors(i.e. facts extraction, information retrieval - Yargy Parser)

```
startQWords = []
with open('StartWords.csv', newline='') as File:
    reader = csv.reader(File)
    for row in reader:
        startQWords.extend(row)

actors = []
with open('actors.csv', newline='') as File:
    reader2 = csv.reader(File)
    for row in reader2:
        actors.extend(lower(row))

R_1 = rule(dictionary(startQWords), gram('VERB'), gram('NOUN'), gram('NOUN'))
R_2 = rule(dictionary(startQWords), gram('VERB'), gram('NOUN'))
R_3 = rule(dictionary(startQWords), gram('NOUN'), gram('NPRO'), gram('VERB'))
R_4 = rule(dictionary(actors))
R_5 = rule(dictionary(type_dictionary))
```

```
Type of the task is uniform rectilinear

Актор 1 is Мотоцикл

Motion elements of the actor:
путь 90 км
время 3 часа
скорость 50 км/ч

The main question(s) of the text is(are)
['Какой', 'была', 'скорость', 'мотоциклиста']
['Какой', 'путь', 'он', 'прошел']
```

As you can see, the parser can also highlight the type to define, for this I used the condition of the dictionary with acceleration or not.

Notes:

- 1) I only considered straight line movement, I did not consider circular and other types of movement, it is very similar and is added approximately in the same ways.
- 2) Also in the model you can remove the anaphoric connection, but at the moment it is not so important (For example, in one sentence the author uses the word Motorcyclist, and in another sentence replaces it with he, the code will think that there are two actors, when this is not the case and will give different versions of questions, actors, etc.)
- 3) What's more, I should have made a more coherent synonym series, but so far I haven't used it in my project making them more simplified and settling on singular and declension by case.
- 4) Another challenge was to find the connection between the actor and the Motion Element. By this link it is clear what are the parameters of EP and it allows to solve the problem successfully, but the task becomes more complicated when there are several actors then it is necessary through parsing to look for their motion elements, so I simplified the task and output all MEs for one actor and can't output the corresponding links between actor and ME

6.6. Results of my project

- 1) I did the research and created an overview about NLP, russian parsers and kinematic tasks that can be very useful for parser development.
- 2) Gathering statistics. In the course of the work I made statistics on the tables of the given actors, movement elements of their properties and defaults. Thanks to work with tables it is easier to work with the parser in the project and as thanks to statistics it is possible to reveal some aspects to which understanding of conditionalities of tasks is simplified.
- 3) I created the ontology-controlled parser that can read specific kinematics tasks from physics and it can extract the type of task, actors, motion elements defined by numbers and short names, and the main questions of the task.
- 4) I presented the model of the kinematic tasks, so that my parser could be developed further in the future
- 5) The parser is able to present some templates for the kinematics tasks, one just need to write the text of the task (if needed, you can use mine task source named “Task descriptions.csv” in the github)

7. Bibliography

A Brief Survey of Text Mining: Classification, Clustering and Extraction Techniques [В Интернете] / авт. Mehdi Allahyari Seyedamin Pouriyeh, Mehdi Assef, and others. - <https://arxiv.org/pdf/1707.02919.pdf>.

Computational Linguistics [В Интернете]. - <https://plato.stanford.edu/entries/computational-linguistics/#GoaComLin>.

Information extraction in text mining [В Интернете] / авт. Mulins Matt. - 2008 г. - https://cedar.wvu.edu/cgi/viewcontent.cgi?article=1003&context=computerscience_stupubs#:~:text=Where as%20Data%20Mining%20extracts%20patterns,news%20articles%2C%20or%20business%20reports..

ONTOLOGIES IN COMPUTER SCIENCE [В Интернете] / авт. Man Diana. -

<http://www.math.ubbcluj.ro/~didactica/pdfs/2013/didmath2013-06.pdf>.

Tomita-parser [В Интернете]. - <https://yandex.ru/dev/tomita/>.

Сборник Задач По Физике [В Интернете] / авт. сост. Е.Г. Московкина В.А. Волков. - <https://static.my-shop.ru/product/pdf/389/3887605.pdf>.

Автоматическая обработка текстов на естественном языке и анализ данных [В Интернете] / авт. Большакова Е.И. Воронцов К.В., Ефремова Н.Э., Клышинский Э.С., Лукашевич Н.В., Сапин А.С.. - https://www.hse.ru/data/2017/07/22/1173852769/NLP_and_DA.pdf.

Анализ Инструментов Для Нормализации Слов [В Интернете] / авт. Александр Богуренко Павел и Теплярев. - <https://newtechaudit.ru/normalizaciya-slov/>.

Синтаксические Парсеры Для Русского Языка [В Интеренете] /

<https://habr.com/ru/company/sberbank/blog/418701/>

Проект Natasha. Набор Качественных Открытых Инструментов Для Обработки Естественного Русского Языка (NLP) [В Интеренете] /

<https://habr.com/ru/post/516098>

8. Additional requirements of Mentor/Supervisor

9. Project Works' Calendar

30.11.2020 - 30.02.2021 - The start of researching part with observing instruments, cases and other important details. Meeting with experts every Friday.

22.02.2021 - CP1

30.02.2021 - Work with JSON and parsers, task base.

13.03.2021 - Making a table of physical problems for the template

20.03.2021 - Studying the Yargi Parser and the Natasha Project Library(Because of the Tomite Parser flaw)

30.03.2021 - Learning the SlovNet and Razdel templates for syntactic and morphological parsing

20.04.2021 - making parsing patterns, pattern conditioning

29.04.2021 - Checkpoint 2. Making a report on the work done.

15.05.2021 - Connecting Xl tables with a finished template for parsing, checking and improving errors

20.0.2021 - Completion of a complete table with all these tasks, the errors are minimized

30.05.2021 - Bug fixes, end of project