Natural Language Interpreter and Arithmetic Word Problem Solver

Jozef Franzen

Resum—Avui en dia, el domini de Processament del Llenguatge Natural pertany als camps més tractats de la Intel·ligència Artificial. En el context de la varietat immensa de les seves aplicacions es pot destacar, que la prova d'intel·ligència de màquines – el test de Turing – comporta la detecció de la intel·ligència justament mitjançant el xat fent servir el llenguatge per demostrar les capacitats mentals. En aquest sentit, doncs, l'anàlisi computacional de la comprensió i producció del llenguatge pot considerar-se d'importància especial. Aquest treball té com a objectiu entrellaçar per les seves sortides els resultats de l'anàlisi de llenguatge natural amb el punt característicament fort dels ordinadors – la manipulació amb números. Dit això, es poden delimitar dues tasques principals en que consisteix el present projecte. Per una banda s'hi té l'analitzador del Català encarregat d'esbrinar la representació sintàctica de la frase donada i per l'altra, el sistema per resoldre problemes aritmètics senzills que permet passar de la interpretació de les frases formant l'enunciat en el llenguatge natural a la solució del problema. Per acabar, s'inclou la discussió pel que fa als resultats obtinguts, possibilitats de millores en el futur i causes de deficiències detectades.

Paraules clau—llenguatge natural, processament del Català, anàlisi sintàctica, anàlisi semàntica, sistema per resoldre problemes, intel·ligència artificial simbòlica

Abstract—The field of Natural Language Processing (NLP) belongs nowadays to most studied and developing fields of Artificial Intelligence. Of countless applications of tasks of the NLP it could be particularly remarked that the intelligence test of a machine – Turing Test – involves detection of a human-like intelligence precisely through the language-based chat aimed to demonstrate sufficient mental capacities. In this sense, the computational analysis of language comprehension and production can thus be deemed of a prominent importance. This work has as its ultimate objective to combine for its outcomes results of the language parsing with notable strengths of the computers – manipulation of numbers. Therefore, two principal tasks of this project can be outlined. The parser of the natural language selected for this project – Catalan – is destined to find a syntactical representation of the given sentence, and the arithmetic word problem solver links up the established interpretation with resolution of an arithmetic word problem given in the natural language. Finally, the work concludes by discussion focused on analysis of results, opportune enhancements for the future work and possible ways how to address encountered issues and deficiencies.

Index Terms—natural language, processing of Catalan, syntactic parsing, semantic parsing, knowledge extraction, problem solver, symbolic artificial intelligence

1 Introduction

SINCE the dawn of Artificial Intelligence, the field of Natural Language Processing has attracted much attention as dealing with one of direct manifestations of the human intelligence – the capability to understand and produce utterances conveying meaning, abstract ideas or even committing actions affecting the reality. Hence, programming a computer, or any machine in general, in the way endowing it with the apparatus allowing it to communicate with humans has been the most challenging quest of the NLP yet from its very beginnings.

The utmost motivation behind this project has been initially driven by the interest to acquire knowledge about the state of the art in the language engineering actively by means of the project elaboration. The primary goal of a parser / interpreter of the natural language sought familiarisation with the basics of NLP in the first place, materialised by the implementation of a program transforming the input sentence to its corresponding syntactic representation. As such, the expected outcome shall be the syntactic tree providing information on parts of speech of each word in particular, identifying the function in the sentence performed by the word and determining relations between words whenever they form phrases implying relations such as grammatical agreement.

Once this stage worked with the allowable margin of error, data derivable from such a syntactic interpretation were supposed to be used for a further processing. This would ideally entail extraction of the relevant data to be computationally processed so as to get an objective, and objectively classifiable result in terms of its correctness.

There were several alternatives thought as for the posterior application of the developed parser. More concrete-

- *E-mail de contacte: jozef.franzen@gmail.com*
- *Menció realitzada: Computació*
- *Treball tutoritzat per: Dr Ramon Baldrich Caselles (Departament de Ciències de la Computació)*
- *Curs 2015/16*

ly, the first concept was based upon the idea of a **chatterbot** able to guide a conversation towards resolution of a problem. Despite its potential advantages whereof the most striking would clearly be the impression of talking to a computer, given the scope of this project, abilities of such a chatterbot could only be quite limited. As a consequence, a different alternative has finally had to be adopted. Among suggested solutions, a much **enhanced parser** or a simple virtual assistant / **question answering system** had been proposed. Whilst the idea of keeping the project within the purview of the parser was quickly discarded owing to the lack of more concrete outcomes, the question answering system was seriously considered. Though, because of the need to understand arbitrarily complex sentences commonly appearing in the intended source – an internet website – alongside other elements that would make the parsing much more difficult, such as tables, for example, the idea gradually evolved into the **arithmetic word problem solver**. This solution is particularly attractive for the conciseness and straightforwardness of the language used, common pattern followed by problem statements and the fact that such problems lead to the unique and objective result.

Considering concrete options for the implementation depending on the state-of-the-art tools for Natural Language Processing, before formulating objectives and describing the work methodology of this project, the state of the art shall be exposed in the next section. Only afterwards the whole work carried out for the development of established objectives will be explained in detail, followed by the summarised results. Ultimately, a discussion fusing the analysis of issues encountered with suggestions for the future work will be presented along with the evaluative conclusions to be drawn from the present project.

2 State of the Art

As comprising two actual tasks that can up to a certain extent be dealt with independently of each other, the state of the art ought to reflect both fields respectively.

2.1 Parsing of the Natural Language

As far as the parser is concerned, depending on the desired strategy and degree of control over the interpreter, several approaches to the automated parsing of natural language have been developed, mostly following the distinction of **symbolic** vs **sub-symbolic** AI. The most cutting-edge tools for symbolic Natural Language Processing can be focused on a specific language (e.g. KoNLPy [1] exclusively for Korean, using Python), provide support of several languages based on available prebuilt models (e.g. Apache OpenNLP [2] for Java, Stanford CoreNLP [3] using Java or spaCy [4] for Python) or make use of a grammar composed of production rules somewhat resembling grammars of programming languages, allowing the application to cover virtually any language (e.g. Natural Language ToolKit, NLTK [5] for Python).

In the context of its versatility, a tool like NLTK would definitely be preferable for a symbolic approach.

To provide a basic idea how natural language may be represented in a symbolic model specifically by means of a production-based grammar, in the grammar of a programming language non-terminals correspond to structures ranging from basic expressions, through higher-level structures such as function calls or loops up to classes. In contrast, in the grammar of a natural language non-terminals can represent structures ranging from affixes or special functional words, through phrases spanning across multiple words that carry a specific function up to the sentence as a whole. Then, just as terminals in the grammar of a programming language represent the limited and language-specific vocabulary of the lowest level, so do they represent in the grammar of a natural language its particular vocabulary – that is, the set of tokens found on the leaves of an eventual syntactic tree.

The sub-symbolic approach to the NLP is characterised by the artificial neural network serving to hold the knowledge of the parser, whereby no explicit grammar definition is necessary. On the other hand, a tagged training set is required for learning, what could ultimately end up being then the factor limiting the choice of a language to develop the parser for. One example of a framework centralised on the sub-symbolic NLP is the Deep Learning for Natural Language Processing, DeepNL [6] written for Python.

2.2 Automatic Word Problem Solving

For arithmetic problems are characteristic by the unambiguous procedure leading towards the sole correct result, the first endeavour to join the NLP with word problem solving dates back to 1964, when the program STUDENT capable of solving secondary school algebra word problems was developed by Bobrow [7]. This system employed a so-called **"relational model"** populated by data extracted from the statement of a story problem, where expressions in a "restricted subset of English" were parsed to identify variables, "substitutors" and operators denoting relations between objects forming the relational model to be expressed using a set of algebraic equations. Among further noteworthy works in this field chiefly from recent years, [8] and [9] can be referred to as dealing with simple arithmetic word problems involving addition and subtraction only by means of **verb categorisation** and **predefined problem types**, respectively. Of works tailored to a specific subset of problems, [10] is specialised onto automatic solving of problems looking for numbers fulfilling relations according to the statement of problem given in the natural language, whereas [11] covers four domains of word problems, namely investment, distance, projectile and percent. Finally, as the most multi-purpose and generic problem solver, the solution [12] developed by Kushman et al. should be discerned owing to only limitations imposed by **induced equation templates**.

3 Objectives and Methodology

In the context of the state-of-the-art solutions and the interest of having more fine-grained control, the **symbolic approach** with Python+NLTK has been favoured over the Deep Learning for the implementation of the parser of the natural language. As to the language choice, in spite of the wide-spread availability of tools supporting English, for the particular setting of this project, specifics of Catalan language along with lack of similar solutions developed for it made it prevail over albeit opportune, but still more common choices such as Spanish or French.

3.1 Objectives

Inasmuch as the NLP-focused development presented an unknown harder to evaluate a priori, the following set of general objectives has been defined to formalise the goals of the realisation of the present project:

- Define a **computable representation** of the grammar of a natural language
- Provide **syntactic analysis** of a given statement with the outcome being the syntactic tree of the given statement
- Provide **semantic analysis** of a given statement with the outcome being the assignation of applicable grammatical categories
- Identify syntactic and semantic **irregularities** of the parsed statement
- Propose **corrections** to common mistakes provided that the parsed statement gives enough information to do so
- In the case if a word had not been found in the known corpus, as long as the internet connection is available, **load additional information** from an online dictionary, such as [13]
- Develop an interface for **problem-solving engine** capable of resolving arithmetic problems at the level of the primary school
- Implementation of **language production** to give the user response to the question that has been asked
- Implement a **graphical user interface** tailored to the needs of the system being developed

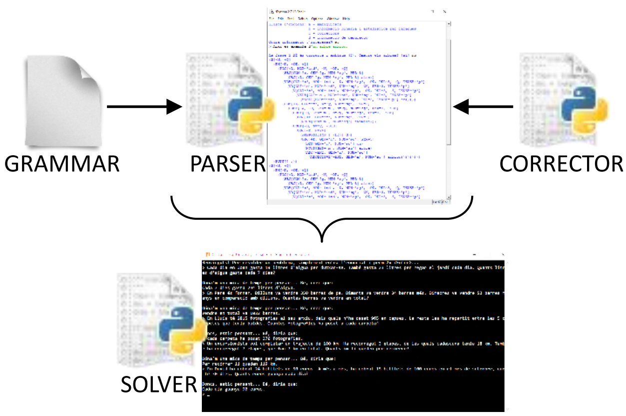


Figure 1: Global organisation of the proposed project

Even though initial objectives differed assuming a chatterbot was about to be developed, in either case there have been three main categories of objectives: **parser-related** including parser's extensions, **application-related** (initially a chatterbot, posteriorly the word problem solver) and the tentative **graphical user interface**. In virtue of the extent of definition of the objectives, they provided the basic – neither vague nor too restrictive – guidelines for the actual development of both stages of the project.

3.2 Methodology of Development

For this project it was decided not to follow any of common methodologies frequently used in the sector of software engineering. Instead, the project was developed and its progress tracked pursuant to the customised methodology based on the **iterative process** borrowed from most common methodologies of agile development.

A regular iteration was set to span over the period of **one week**, being each iteration characterised by a predefined set of desired **achievements**. By the end of every iteration a short report of progress reviewed whether all expected achievements had been accomplished, providing main relevant technical details. In the case any issues were encountered, they were pertinently explained along with an eventual workaround or solution, if found, including also the reasoning as to the possible impacts.

The main reason motivating this methodology consisted in the nature of the system to develop. Insofar as tasks related to the NLP presented unknown factors hard to predict as to their success or complexity, this had to be considered upon scheduling not completely determinable until some feedback was available. The iteration of one week was thus a fair compromise ensuring availability of **sufficient progress tracking** information and possible and plausible rescheduling when necessary.

4 Development of the Parser

Prior to taking the definitive decision in favour of Python and NLTK, functionalities of the latter were analysed to determine its suitability for the conceived project. Of the most important features, the suite of libraries encompassed within NLTK supports among others working with various **corpora**, **tokenisation**, stemming, part-of-speech tagging, or **parsing** of the given tokenised text. Besides, one of the featured corpora with built-in support, *cess_cat* is a corpus of Catalan language by Centre de Llenguatge i Computació of the Universitat de Barcelona.

Despite its support of a wide range of functionalities, the suitability of NLTK for parsing of Catalan underlay the condition of a **feasible representation** of its grammar. Unlike English being the most frequently analysed language, Catalan abound in its richness of verb forms, whereby in order to represent its entire verb inflection, full conjugation of each verb would need to be specified within the grammar for the parser to recognise any possible verb form. However, this approach implied having at least 50 entries per one non-defective verb (most of the verbs with exception of ones like *caler* or *dar*) independently of its regularity. Disregarding for now the irregular verbs, if the parser had to recognise solely 20 common regular verbs possibly following one and the same conjugation model, an amount of 1000 entries would still be required.

4.1 Tokenisation

So to address this drawback, the apparent strategy of splitting the verb stem from suffixes must have been adopted with the parser, and particularly the tokeniser, adapted correspondingly. The tokeniser is in charge of pre-processing the input sentence by converting it to atomic (lexical) **tokens** that must occur in the vocabulary, otherwise the parser throws an error. The simplest tokeniser could do enough just by dividing the sentence into particular **words** according to whitespace and punctuation, what is indeed the first step.

Owing to the support of multi-sentence input, in fact there are **multiple levels of tokenisation** before the input is actually parsed. Firstly, the processing starts by the global normalisation and identification of individual sentences. Secondly, it continues by normalisation on the scope of a sentence consisting in the differentiation of particular words. Lastly, further splitting of words that are considered to be composed of two atomic tokens is performed. The Appendix A1 illustrates the complete succession of steps involved in the processing of the input. For the sake of better expressiveness, though, the entire process comprising all levels of tokenisation will be described step-by-step in the following chapters and simultaneously illustrated on the next (erroneous) example:
Quant al seu desenvolupament, pensem que l'intel.ligència_artificial ha anat avançant molt ràpidament.

4.1.1 Normalisation and Sentence Tokenisation

The raw input from the user is first and foremost normalised so that it follows systematic rules in the use of **non-alphabetic characters** including underscores (to address how multi-word expressions are usually represented in corpora) or apostrophes of different forms, unifying both Right Single Quotation Mark (U+2019) and Apostrophe (U+0027) into the latter form. Other than that, the full stop appearing amidst two *l*'s is replaced to follow the Catalan standard *l·l* and runs of whitespace characters are replaced by a single space. Done this, the example turns into:
Quant al seu desenvolupament, pensem que l'intel·ligència artificial ha anat avançant molt ràpidament.

Once normalised, the **start of a sentence** is defined as the start of the user-supplied string or an uppercase letter coming after any of the usual sentence-ending marks (full stop, question mark and exclamation mark) or semicolon, followed by a space.

Subsequently, for each sentence in particular, **words** are differentiated in two phases. First, preposition+article contractions are split, converted to lowercase and then a simple splitting according to specified delimiters (space, dash and apostrophe) is performed. So, even each weak pronoun ends up as a separate token on its own.

After contractions of a preposition and article are split, the example gets transformed into:
Quant a el seu desenvolupament, pensem que l'intel·ligència artificial ha anat avançant molt ràpidament.

Delimiter-based splitting then results in the list:
["quant", "a", "el", "seu", "desenvolupament", ",", "pensem", "que", "l", "intel·ligència", "artificial", "ha", "anat", "avançant", "molt", "ràpidament", "."]

4.1.2 Word Tokenisation

Two classes of words are considered as composed, so they need to be further split into two atomic tokens; namely **verb forms** composed of a suffix attached to the stem and compound **adverbs** ending in *-ment*. However, as the information whether a word belongs to the class implying that it may be subject to such splitting is not available until the sentence has passed through the parser, each token is first searched in the parser's vocabulary. If found, the next token is tested, elsewise, it is attempted to find the correct cut and only if the word could not be divided, it is searched in the corpus processed as explained in the section 4.3.

To determine the **correct cut**, all possible token pairs are generated and tested for the coverage in the vocabulary of the parser and whether the pair is **unifiable** based on the rules of the parser's grammar. If no cut could have been identified and the word is not found in the corpus, it is just to be added to a list of unknown words.

As a result, cutting the words to get atomic tokens covered in the parser's vocabulary yields in the case of the example picked for illustration the list:
["quant", "a", "el", "seu", "desenvolupament", ",", "pens", "em", "que", "l", "intel·ligència", "artificial", "ha", "anat", "avan", "çant", "molt", "ràpida", "ment", "."]

Upon closer observation it can be seen that *desenvolupament* as a noun has correctly not been split in spite of ending in *-ment*, whereas *ràpidament*, again correctly, has. Also, regularly conjugated verb forms *pensem* and *avançant* have been correctly split into a fixed stem and suffix.

4.2 Implementation of the Parser's Grammar

The grammar of the parser serves as the **rule base** describing all valid structures, much like the grammar of a programming language defines syntactic and semantic rules compulsory for any program to obey. If and only if the given input can be expressed in terms of these rules, it can be considered valid. There are three types of grammars supported by NLTK that ought to be highlighted; context-free grammars (with extension cfg) as the most basic, probabilistic grammars (pcfg) optionally assigning probabilities to the rules and feature-based grammars (fcfg) where any non-terminal can be attributed a set of arbitrary parameters. For the sake of synthetisation of a computational expression of the grammar of Catalan, the **feature-based grammar** was used.

The grammar of a natural language as understood by the NLTK comprises the set of **production rules** in the form LHS -> RHS. Any unquoted string is interpreted as a non-terminal symbol, whereas anything on the right-hand side introduced in single or double quotes is a terminal symbol. If the rule involves terminal symbols, it is designated as **lexical** for the set of all terminals of the grammar forms the **vocabulary** of the instantiated parser. In addition, a grammar rule can syntactically consist of multiple **alternatives** separated by a vertical bar. On the other hand, the production rules of grammars in NLTK do not feature built-in support for optional symbols, what would result in the need of stating explicitly every variant of otherwise still the same rule. Hence, the implemented parser **extends the syntax** of production rules allowing for multilevel, possibly nested **optional symbols** introduced in parentheses. In order to comply with the standard syntax, then, the grammar featuring optional symbols is compiled to create a standard feature-based grammar file, deploying rules to include and exclude symbols appearing in the parentheses.

For the actual details how the grammar of Catalan has been implemented, as it falls outside the scope of this work, the reader is referred to the document of **Grammar Specification**. Among linguistic sources used for the development of the grammar, [14] and [15] shall be particularly cited. It should be noted, though, upon conception of the grammar it had to be taken into account that the NLTK does not include tools that could handle particularities like stem alternations (e.g. *sortir* – "to leave" → stem *surt-* when stressed) or different orthography conditioned by the environment (cf. *menjo* vs *mengem*, "I eat" vs "we eat"). As a consequence, the first conjugation class (*-ar* verbs), for instance, had to be divided into 8 subclasses so that what is supposed to be the verb stem remains unchanged throughout the conjugation. Nevertheless, this occurs across all the grammar obviously without being limited to verbs of the first conjugation only.

4.3 Corpus Data Inclusion

In order to circumvent the need for the grammar to include all the acceptable vocabulary, although the initial objectives as detailed in the section 3.1 counted on the integration of an online dictionary, owing to availability of an offline alternative – the corpus *cess_cat* – the latter was given the preference. Thereby, this corpus has been processed to speed up the word lookup, creating thus the **external dictionary** file serving as the last source to resort to before the word is labelled as unknown, effectively preventing the sentence in question from being parsed.

Owing to the fact that certain closed grammatical word classes (e.g. pronouns) have already been mostly incorporated in the grammar, the processing of the corpus involves filtering out all the words except for nouns, names, adjectives and verbs supposed to represent the vast majority of words missing in the parser's intrinsic vocabulary. The transition from the inherent format of word categorisation tags stored in the corpus to grammatical category attributes utilised in the grammar is then realised by establishing **correspondence** to match the categorisation of words in the corpus with categories differentiated for each word class of interest. For example, the word *xiularan* ("they will whistle") is accompanied by the string "vmif3p0" wherefrom corresponding attributes VERB[PER=3, NUM=pl, MOD=ind, TENSE=f] can be derived. Just as all the verbs are classified likewise, so does every other part of speech share a common pattern.

Whereas the used corpus *cess_cat* comes bundled with an information claiming that it accounts for roughly 500,000 words, the actual processed dictionary finally contains 35,411 lexical entries (keys). Forasmuch as some may be associated with more than a single word class at the same time, a total of 38,609 entries were exported in the dictionary stored in a file apart.

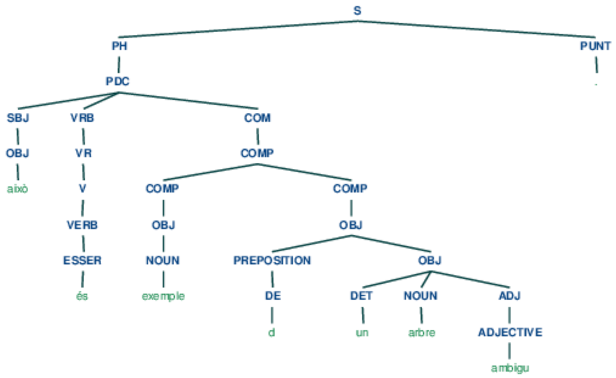
4.4 Output of the Parsing

Reached this point, the parser can analyse the **tokenised input**. Provided that all tokens could have been identified and appropriate rules found in the grammar, the **parsing** consisting in a simple library function call yields a pair (sentence, list of syntactic trees) for each detected sentence with the list of **syntactic trees** possibly containing more than one tree if multiple suitable interpretations have been found. Yet when this happens, it is often due to ambiguities not caused by genuine syntactic ambiguities reflected through several possible interpretations, but rather ambiguities not caught by fairly permissive rules of the grammar. Contrariwise, if no possible interpretation has been found at all, the list of syntactic trees may be empty, or alternatively if the sentence could not be parsed owing to unknown words, the list of words that were not identified is returned instead.

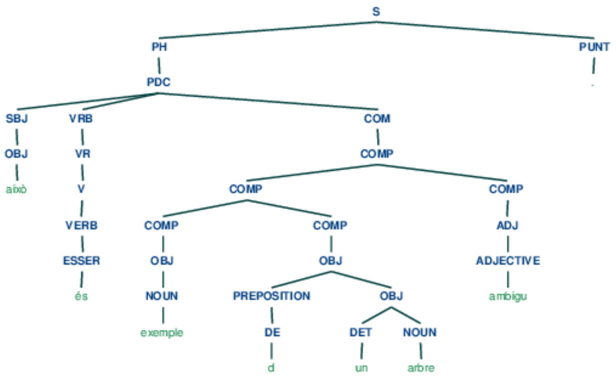
The parser offers four **modes**, each associated with a specific letter that can be used at any time to turn the particular mode *x* on (+*x*) or off (-*x*):

a information regarding **ambiguities** in the sentence
b **batch** mode when no resulting trees are printed
c enables **corrections** to be proposed

d **debug** mode making also that detailed information about the progress of the parsing is output



(a)



(b)

Figure 2: Two syntactic trees (simplified) returned for the sentence "*Això és exemple d'un arbre ambigu.*" The tree above (a) attaches the word *ambigu* to develop *un arbre*, whereas the tree given below (b) links that word to *exemple*. For more complicated sentences, such as the first sentence of the Abstract where the parser yields 36 syntactic trees, many of the resulting trees are actually redundant.

4.5 Mistake Corrector

In conformity with the objectives defined in the section 3.1, to **propose corrections** to common mistakes, a special corrector script was developed. Because the correction relies upon categorisation of words into parts of speech, the information that is available once the sentence has been successfully parsed, the corrector cannot handle mistakes causing the sentence being declared as incorrect. Besides, when this happens, there is no information concerning the issue why the sentence could not be interpreted. Thus, in order to try somehow to correct it, unless a strategy sophisticated enough was employed, the brute force would require keeping on varying certain grammatical categories subject to grammatical agreement for opportune words of the sentence until a valid alternative is found. Still though, this approach would not guarantee finding a working correction either, therefore the idea of such a potent error corrector must have been discarded.

Instead, the corrector script was designed to address chiefly cosmetic irregularities related to **apostrophising**, including probably one of most peculiar areas of Catalan grammar – **weak pronouns**. For any given sentence, if the correction mode is enabled, past its successful parsing the attempt to correct the sentence is made and if the result differs from the input, the correction is proposed. The corrector is able to form the correct combination of any (up to four) given weak pronouns regardless of the order in which they have been given, taking into account the position (proclitic/enclitic), verb beginning/ending as applicable, impossible combinations and even the distinction between Catalan and Valencian. Unless specified otherwise, assuming Catalan standard by default makes *li* is replaced by *hi* when accompanied by 3rd person direct object pronoun. Ultimately, for the apostrophising of articles, the preposition *de* as well as weak pronouns, particularities of cases where they ought to take the apostrophe are respected, except for exceptions. Of the sources useful upon implementing the mistake corrector, [16], [17] and [18] are considered of most important contribution.

5 Implementation of the Problem Solver

Before proceeding to the implementation of the problem solver, **sample problems** have been selected to guide the implementation and, obviously, for posterior testing. The selection borrowed from sources [19], [20], [21], [22], and [23] can be matched to the type of problems as used in [12] by Kushman et al., involving addition and subtraction of terms mostly obtained by pairwise multiplication of two related numbers. This was taken into consideration in the preliminary design of the problem solver.

Subsequently, selected problems had to be **rephrased** (cf. "substitutors" as seen in the Bobrow's work [7]), substituting for instance *cada setmana* by *cada 7 dies*, or *quants diners* by *quants euros*. Doing this pursues the purpose of mimicking the background knowledge implicit for humans, yet still hard to infer for a machine.

5.1 Interpretation of the Statement of Problem

By analogy to human understanding and taking advantage of the ease of detection of the items of interest, the first step in the solution of an arithmetic word problem consists in the **interpretation**. Computationally, this entails recognition and extraction of data (henceforth denoted as "items") that are to be manipulated with for the objective of problem resolution to be attained.

Hence, for each problem the interpreter starts with an empty **knowledge base**, record of sentences along with their syntactic trees and the reference to the type of the last added item for occasions when the repeated reference using *cadascun* or *cadascuna* is encountered. Then, each sentence forming the given story is processed word-by-word, looking for two particular indicators – **numerals** and **words beginning with *cada*** perceived to specify the unitary quantifier per subordinate unit.

When a **numeral** is found, considering the very first word of the sentence if starting with *quant* as a special case of numeral equal to zero, if the next word has been classified as a noun, **a new item** of that type and respective number is inserted under the corresponding key in the knowledge base. Otherwise the type of the last added item is used as the item type and insertion key. Euros are deemed a special unit since they are usually directly related with other items appearing in the same sentence. In other words, for *10 bitllets de 20 euros* banknotes and euros are correctly linked up without need to have it paraphrased like *10 bitllets, on cadascun val 20 euros* (using *cada* normally needed so as to get items tied together).

In order to distinguish between addition (by default) and subtraction based on **verb categorisation** (cf. [8] by Hosseini et al.), for each item the action undergone is stored. The value to fill in is the non-auxiliary verb of the i^{th} verb phrase for i^{th} item detected in the sentence as long as i does not exceed the number of verb phrases in the sentence. A special action "DE" is attributed to an item if the previous item originates in the same sentence and coincides in its type and the item in question comes preceded by the preposition *de* found anywhere earlier in the sentence, for these circumstances point at the subset relation that is supposed to trigger subtraction, too.

Further criterion determining the arithmetical operation to apply is the **compatibility** of items in question. In order to estimate whether two items are compatible, it is needed to store an attribute retrieved based on words following the number and item type; possibly an adjective, either coming immediately after (e.g. *13 correus electrònics*) or after a succeeding verb (e.g. *3 són domèstics*), or any complement in general (e.g. *6 galetes de xocolata*). Through the scrutiny, several types of complements were excluded as misleading, namely constituents of eventual items (numerals and words beginning with *cada*) and subordinate clauses.

When the assigned attribute equals *Més* – "More" or *Menys* – "Less," the item's number is regarded as **relative** with respect to an antecedent. In such case, the sentence the antecedent should stem from is identified according to the magnitude of set intersection of words found in the analysed sentence and those occurring in the sentence under consideration. When the sentence having most in common with the analysed one is discerned, the item of the matching type proceeding therefrom is retrieved and number of the "dependent" item adjusted accordingly.

Once a **word starting with *cada*** is found in the analysed sentence, for *cadascun* or *cadascuna* it is automatically assumed to refer to the type of the last added item with the **unitary quantifier** of 1 (e.g. *4 bombons, dels quals cadascun conté 2 nous*). Elsewise, if *cada* is followed by a numeral, the unitary quantifier is set to the specified number of units being the next word. If no numeral follows (e.g. *cada dia*), the unitary quantifier will in turn refer to 1 given unit. Finally, the quantifier is copied to every item found in sentence, so the example about bonbons would result in 2 items added: 4 bonbons per 1 bonbons and 2 nuts per 1 bonbons.

5.2 Resolution of the Question

After the knowledge base had been built, the problem solver is ready to **resolve the question** asked in the statement of problem. At present, the compulsory question must begin with any gender- and number-specific variant of the interrogative pronoun *quant* so that an item with the number equal to zero is instantiated.

Hence, within the first step the item Q whose number equals to zero is searched and retrieved along with a list of all items of the matching type subsequently looped over. For each item i except the one initially having the number equal to zero ($i \neq Q$) destined to accumulate the result, if the corresponding unitary quantifier is greater than 1, it is automatically used as a factor multiplying the item's number. In the other case, if there is a subordinate unit associated with the item, a related item of the type consistent with the sought unit is chosen, using its number to multiply the number of the item i. Note that there may be several levels of indirection if the related item has its own subordinate unit.

Two types or units are considered consistent providing that holds:

$$^2/_3 \leq relation(i,j) \begin{cases} 1 \text{ if } i = j \\ 0 \text{ if } i[end] = j[end] \land i \neq j \\ \max(l \mid i[:l] = j[:l]) \;/\; \text{minlen}(i,j) \end{cases}$$

where:
- *x[end]* represents the last character of x
- *x[:y]* represents the first y characters of x
- *minlen(x, y)* returns length of the shorter argument

In terms of **verb categorisation**, there are two categories of verbs that imply the addition by default to be replaced by subtraction:

- subtraction-employing verbs, considering as a member any verb starting with *baix-*, *des-*, or *ven-*, positive also for the special "subset" action "DE"
- negation-employing verbs, considering as a member any verb starting with *qued-* or *torn-*

When it comes to **putting the items together**, if the action associated with the item Q (what is being asked about) falls within negation-employing verbs, every item is multiplied by -1 prior to being accumulated (added or subtracted) until any non-initial item to be added is encountered. If the action of any item falls within subtraction-employing verbs, the number is about to be subtracted. Otherwise, the number is added by default.

When the items are actually being put together, disregarding the negation that may apply, **addition** normally takes place for items providing that their action does not entail subtraction and they are compatible with the item in question (*Q*). Then, **subtraction** takes place for any items whose action entails subtraction whether they are compatible with *Q* or not. As a consequence, incompatible items can be subtracted, but not added to be included in the solution.

Yet, there are several **particular types of problems** for which the aforedescribed generic strategy will not work. The most complicated type of problems is where the information is not properly linked or relies on data other than relation between subordinate units and item types.

When **no concrete number** has ever appeared before the word *cadascun/a*, the resulting item may have a unitary quantifier while lacking the subordinate unit to seek. In such case, the unit shall be completed automatically according to the type of the item in question (*Q*).

Likewise, it may happen that items of a certain type **cannot be distinguished** according to the information extracted upon interpreting the statement of problem. A classic example here is the subject-reliant distinction (e.g. *una moto ha fet 5 voltes i un cotxe n'ha fet 8*). For such problems with two item types, three items and the fourth to follow an analogical relation allowing for getting two proportional pairs, an optimal sentence pairing is sought in terms of the number of shared alphabetic tokens. Once two sentences having most in common are matched, the other pair is unambiguously determined, permitting thus to quantify the relation between types and units using the pair where both items are known and apply it so that the other pair agrees. Notwithstanding, this approach is a bit limited as one item per sentence is expected so that the pairing can be found more easily without having to take into account a possibility of several items eventually appearing in one and the same sentence.

Ultimately, when the relation in question is the **inverse proportionality** involving two pairs of items both to amount to the equal product, none of the approaches described up to this point was able to yield any other answer but zero since an intermediate result is apparently to have already been computed. To proceed, the total number of items of the type consistent with the *Q*'s subordinate unit (the common product) is thus computed first in a recursive call with the knowledge base adjusted accordingly. That is, the item *Q* in question is removed and the one linked with it by the subordinate unit is substituted by a new item with the number equal to zero. Only then the resulting product is divided by the unitary quantifier of the item in question (*Q*).

The last step of the previous example generalises to any type of problems where the question asks about number supposed to cover a scope different from the one data given in the story are relative to. In such case, the final **result is adjusted** in relation to the number of its subordinate units appearing elsewhere in the statement of problem.

Once the solution has been enumerated, the answer to the question asked in the statement of problem is formulated. In principle, segments preceding and following the main verb phrase should switch places and the result that has been found substitute the interrogative *quant*. Nonetheless, this **sentence formation** has the main drawback in its dependence on the syntactic tree that can possibly be misleading. Moreover, the reconstruction of the main verb phrase from tokens rarely cause that no sentence can be formed; yet even in spite of this, the solution is still printed out.

6 Results

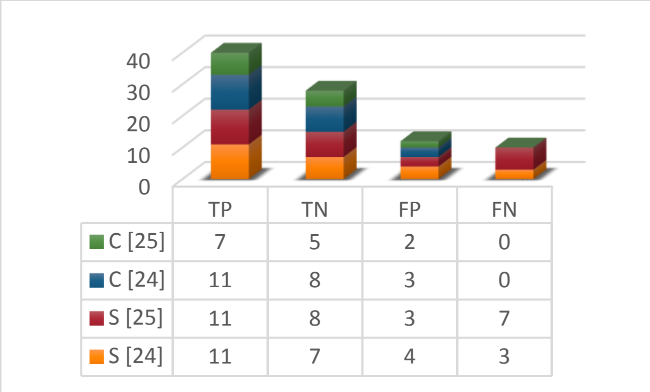
Given the structure of the present project, each part was assessed separately, so shall the results be presented as well. As for the results of parser due to particularities of how its performance was measured, the parser's assessment methodology shall be explained in more detail.

6.1 Parser Assessment Methodology and Results

The parser has been exhaustively tested on a set of 50 sentences, 25 borrowed from a children's book [24] by J. Grave and the other 25 from tests of Catalan as a foreign language [25] at the B2 level of the Common European Framework of Reference for Languages. A representative sample of test sentences is given in the Appendix A3.

Because many sentences required just minor adjustments to pass, apart from numbers of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN), numbers of corrected true positives (CTP), corrected true negatives (CTN) and corrected false positives (CFP) were recorded in addition to data normally gathered. In the context of lack of the negative evidence, for each sentence deemed correct an error has been introduced to generate sentences that ought to be classified as invalid. Hence, aforementioned data correspond to:

- **TP:** correct original sentence classified as correct
- **TN:** original sentence with an introduced error classified as invalid
- **FP:** original sentence with an introduced error classified as valid
- **FN:** even corrected variant of the sentence classified as invalid (absorbing hypothetical CFN)
- **CTP:** corrected variant of the sentence classified as correct
- **CTN:** corrected variant of the sentence with an introduced error classified as invalid
- **CFP:** corrected variant of the sentence with an introduced error classified as valid



	TP	TN	FP	FN
C [25]	7	5	2	0
C [24]	11	8	3	0
S [25]	11	8	3	7
S [24]	11	7	4	3

Figure 3: Results of testing of the parser. Data labelled by S (Strict) take into account original sentences only, whereas data labelled by C (Corrected) include corrected sentences. Number in the square bracket refers to the source of pertinent 25-sentence dataset.

In terms of metrics, there are two ways how to interpret the collected data depending upon where numbers representing corrected alternatives belong to. Assuming CTP⊂TP, final numbers of true/false positives/negatives correspond to column-wise sums of data presented above. When CTP⊄TP in turn holds, column-wise sums of "strict" terms ignoring the corrected data apply with the exclusive exception of false negatives equal then to FN+CTP for no corrections surmised to have been made.

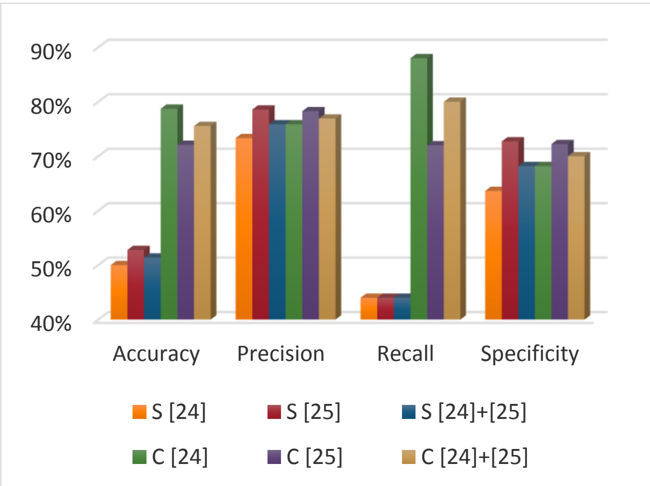


Figure 4: Results of testing of the parser expresed using metrics: accuracy, precision, recall and specificity. Columns labelled by S (Strict) presuppose CTP⊄TP, whereas columns labelled by C (Corrected) presuppose CTP⊂TP.

The most important **conclusion** from the realised tests concerns drawbacks of the current classifier driven by the parser's grammar. These results form above all a primordial basis for **further improvements of the grammar**. It is safe to affirm, though, that the parser is well-targeted, albeit most likely too restrictive. Whilst some restrictions could be easily relieved, there has been a certain type of fixed constructions once less control is imposed over the accuracy diminishes. Among most striking examples the mitigation of requirements for a noun to be preceded by a determiner or for a verb to be introduced by a preposition unless standing as the subject had most serious impacts.

6.2 Assessment of Performance of the Solver

The Ground Truth that guided the preliminary design of the word problem solver served as a basis for its performance estimation. The full text of all 20 problems forming the Ground Truth along with comments on incorrect results is provided in the Appendix A4. Notwithstanding the presumption of overfitting conceivable in the context of a success rate of 18 out of 20 (⇒ 90%), results of performed tests succinctly illustrate the logic attributable precisely to computers.

By no means should this deem the arithmetic word problem solver as going beyond its limited capabilities. Though, to draw a partial **conclusion** regarding the developed solution, in the context of problem types other recent works had focused on, it could cover slightly wider variety of problems mainly owing not to sticking to a presumed subclass or pattern of problems, not speaking about the fact that it should be the **first such system developed for Catalan language**.

7 Conclusions

Despite the issues that have been detected, independently of whether incidental to the developed parser of Catalan or arithmetic word problem solver, the project has successfully been realised in its entirety. As to the objective of the implementation of a graphical user interface, as it would not add any new functionality to the final solution, it was decided to leave it for future work.

Of the most relevant issues, the singular room should be dedicated in the future to a better implementation of the grammar that ought to be preferably organised in a multilevel pyramid allowing for better and more reasonable parsing. At the same time, such an enhancement would definitely potentiate also the extraction of relevant data pursuing a higher level of the natural language understanding. Last but not least, the non-ideal way of dealing with punctuation should be equally improved.

As regards the drawbacks of the problem solver, provided more systematic understanding of the problem statements, it could be elaborated to much better reliability for much wider spectrum of problems. However, more sophisticated techniques, algorithms and strategies in general would be indispensable, not speaking about the effort that would be required to achieve such a degree.

All in all, this work has definitely met its goals not only towards the solid familiarisation with the state of art in the Natural Language Processing brought beyond the simple parsing up to limited understanding and problem resolution. By virtue of its focus on a natural language it even managed to lead forward to a significant knowledge improvement in the language used for the development of the very project – Catalan.

Given its outcomes, double effect and even issues that must have been coped with, this work as a whole has to a great extent contributed to the future interest in related works to be carried out in this field, for which the invaluable knowledge acquired by realisation of this project shall definitely be worth all the effort put forth towards its successful accomplishment.

Acknowledgements

I would hereby like to recognise the extraordinary effort, help and contribution of Dr Ramon Baldrich Caselles who tutored this scholarly work. His encouraging comments, corrections and recommendations made this work followed against the odds the right direction and after all attained all the objectives defined.

Just as I thus want to sincerely thank my tutor, so would I like to thank my sister and mother for their ideas and continuous support that made the whole process much more enjoyable.

Bibliography

[1] L. Park, “KoNLPy: Korean NLP in Python,” 2015. [Online]. Available: http://konlpy.org/en.

[2] Apache Software Foundation, “Apache OpenNLP,” 2015. [Online]. Available: http://opennlp.apache.org/.

[3] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard and D. McClosky, “The Stanford CoreNLP Natural Language Processing Toolkit,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Baltimore, MD, 2014.

[4] spaCy GmbH, “spaCy: Industrial-strength Natural Language Processing,” 2016. [Online]. Available: https://spacy.io/.

[5] S. Bird, E. Klein and E. Loper, Natural Language Processing with Python, 1st ed., O'Reilly Media, 2009.

[6] G. Attardi, “DeepNL: a Deep Learning NLP pipeline,” in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, NAACL 2015*, Denver, CO, 2015.

[7] D. G. Bobrow, “Natural Language Input for a Computer Problem Solving System,” Massachusetts Institute of Technology, Cambridge, MA, 1964.

[8] M. J. Hosseini, H. Hajishirzi, O. Etzioni and N. Kushman, “Learning to Solve Arithmetic Word Problems with Verb Categorization,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014.

[9] B. Amnueypornsakul and S. Bhat, “Machine-Guided Solution to Mathematical Word Problems,” in *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing*, Phuket, Thailand, 2014.

[10] S. Shi, Y. Wang, C.-Y. Lin, X. Liu and Y. Rui, “Automatically Solving Number Word Problems by Semantic Parsing and Reasoning,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015.

[11] K. Morton and Y. Qu, “A Novel Framework for Math Word Problem Solving,” *International Journal of Information and Education Technology,* vol. 3, no. 1, pp. 88-93, February 2013.

[12] N. Kushman, Y. Artzi, L. Zettlemoyer and R. Barzilay, “Learning to Automatically Solve Algebra Word Problems,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, MD, 2014.

[13] Universitat Pompeu Fabra, “Lèxic Obert Flexionat de Català,” Universitat Pompeu Fabra, [Online]. Available: http://ca.oslin.org/.

[14] Wiktionary, a wiki-based Open Content dictionary, “Appendix: Catalan verbs,” [Online]. Available: https://en.wiktionary.org/wiki/Appendix:Catalan_verbs.

[15] P. Fabra, Gramàtica Catalana, Barcelona: Institut d'Estudis Catalans, 2006.

[16] Wikipedia, The Free Encyclopedia, “Regles d'apostrofació del català,” [Online]. Available: https://ca.wikipedia.org/wiki/Regles_d%27apostrofaci%C3%B3_del_catal%C3%A0.

[17] Wikipedia, The Free Encyclopedia, “Catalan personal pronouns,” [Online]. Available: https://en.wikipedia.org/wiki/Catalan_personal_pronouns.

[18] Wikipedia, The Free Encyclopedia, “Numerals en català,” [Online]. Available: https://ca.wikipedia.org/wiki/Numerals_en_catal%C3%A0.

[19] Matemàtiques, “5è Primària - Problemes 1,” [Online]. Available: https://sites.google.com/site/matematiques5eprimaria/problemes/problemes-1. [Accessed 2 May 2016].

[20] Matemàtiques, “5è Primària - Problemes 2,” [Online]. Available: https://sites.google.com/site/matematiques5eprimaria/problemes/problemes-2. [Accessed 2 May 2016].

[21] Escola Betània - Matemàtiques Cicle Superior, “Cinquè de Primària - Problemes de l'1 al 50,” [Online]. Available: https://sites.google.com/site/matematiques5/problemes/problemes51-50.pdf. [Accessed 2 May 2016].

[22] Escola Betània - Matemàtiques Cicle Superior, “Cinquè de Primària - Problemes del 50 al 100,” [Online]. Available: https://sites.google.com/site/matematiques5/problemes/problemes5del50al100.pdf. [Accessed 2 May 2016].

[23] Escola Betània - Matemàtiques Cicle Superior, “Sisè de Primària - Problemes de l'1 al 80,” [Online]. Available: https://sites.google.com/site/matematiques6/problemes-6/blogdeproblemes6a.pdf. [Accessed 2 May 2016].

[24] J. Grave, Les Aventures d'en Nono, Barcelona: Col·lectiu Negres Tempestes, 2009.

[25] Institut Ramon Llull, “Certificats Català Bàsic,” [Online]. Available: http://www.llull.cat/catala/aprendre_catala/certificats_examens_basic.cfm.

APPENDIX

A1. Workflow of the Parser

The presented Figure A1.1 guides throughout the parsing process starting by tasks related to the initialisation of the parser, continuing by reading and processing the user input and ending by the parsing itself plus eventual error correction. Function calls marked in green indicate the termination point of the particular function.

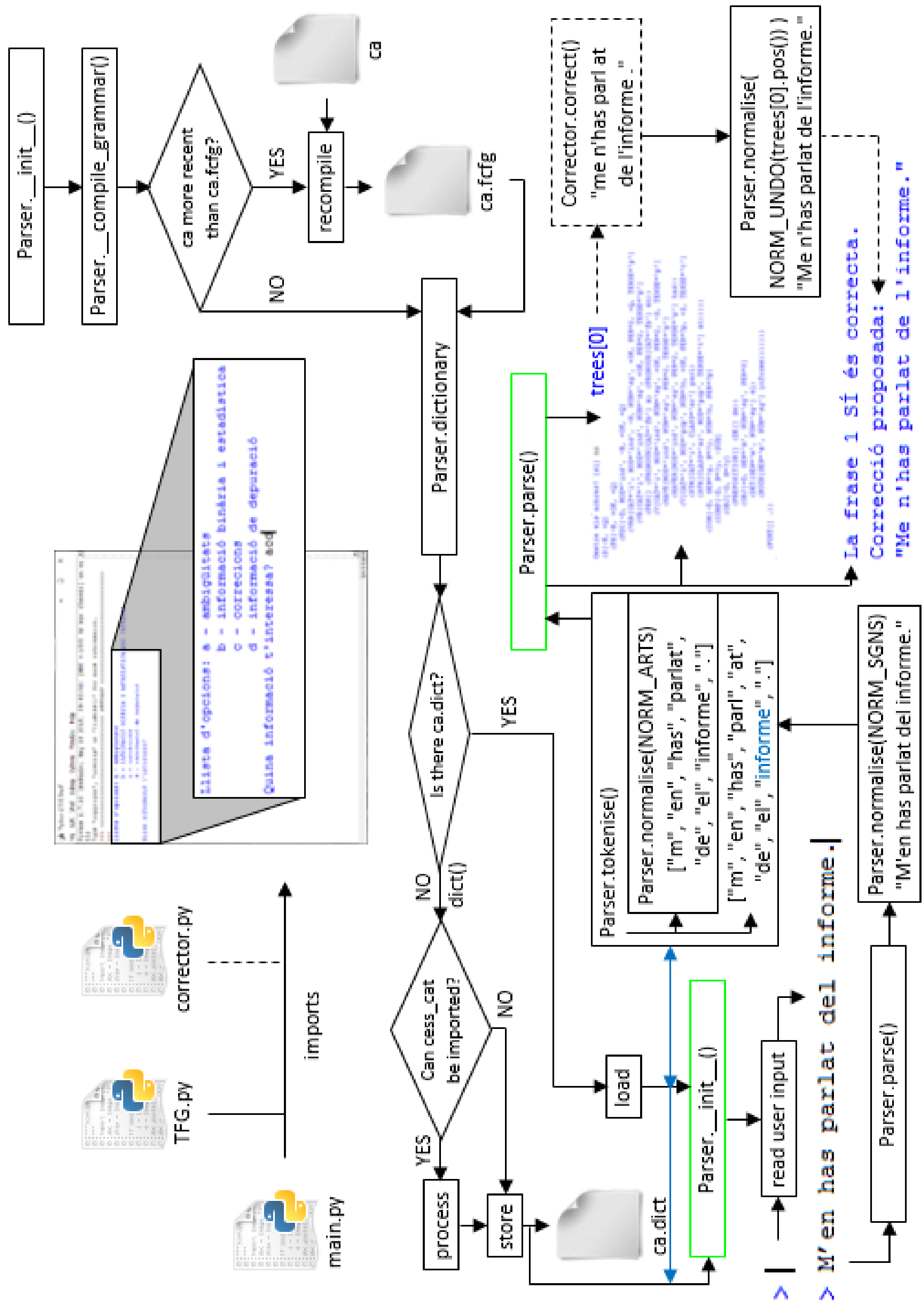


Figure A1.1: Workflow of the parser including most important function calls and stages of the data processing

A2. Workflow of the Problem Solver

To recapitulate the basics of processes involved in the interpretation of sentences forming the statement of problem and subsequent steps toward resolution of the given problem, the Figure A2.1 proposes an overall view onto the sequence of function calls with associated data leading up to the formulation of the final answer.

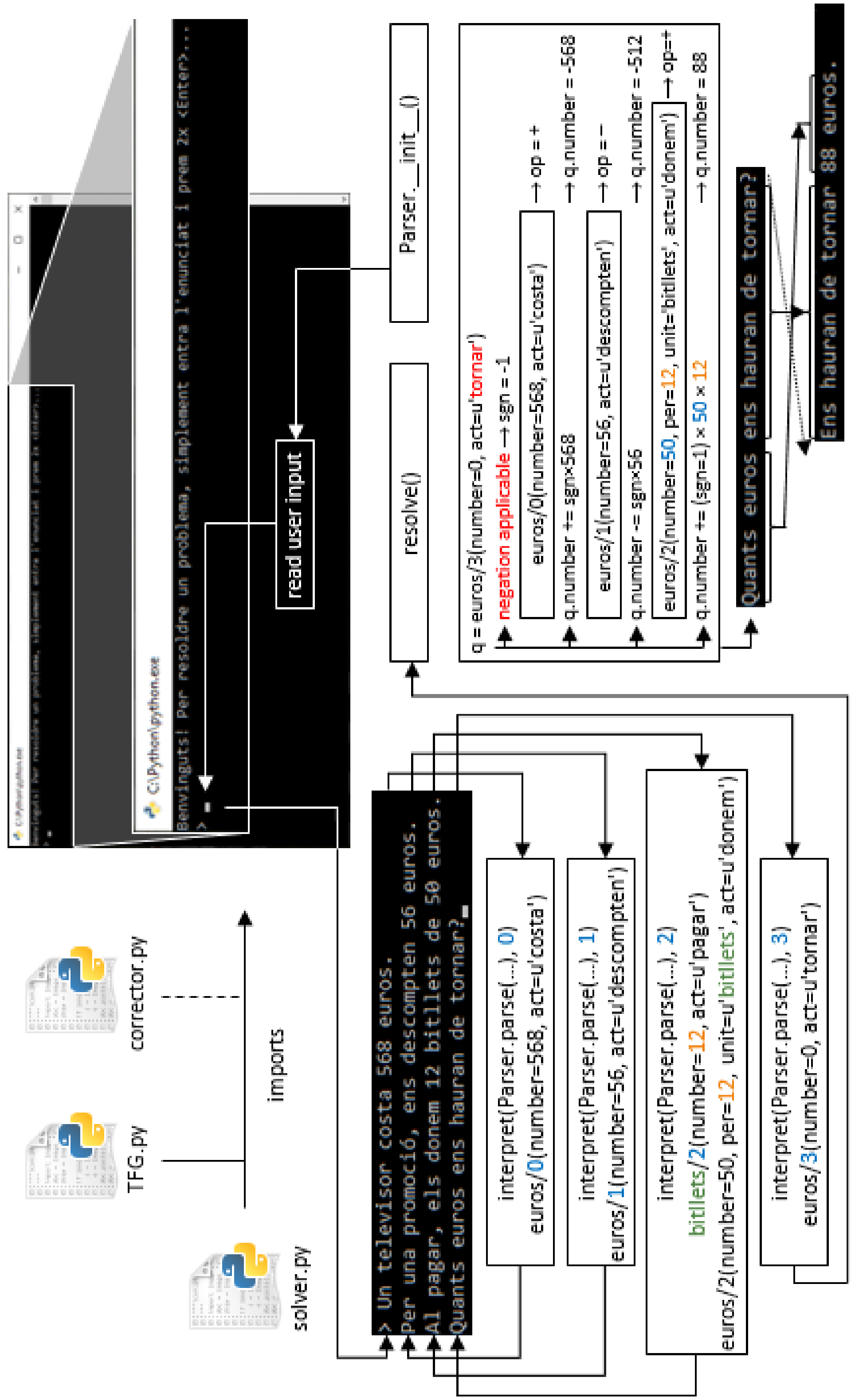


Figure A2.1: Succession of steps involving statement and data processing seeking the problem resolution

A3. On the Drawbacks of the Parser

In the interest of providing more illustrative perspective over the set of sentences used to test the parser, most representative falsely classified samples for each positive and negative evidence are presented along with a short comment on possible causes of misclassification. The full test set is to be found in the Report of Progress I.

Table A3.1: Summary of principal issues of the parser

ID	Sentence	Result
4+	A poc a poc tot allò va anar adquirint vida i animació. *Apart from the phrase "a poc a poc," the main problem would be the segment "vida i animació," since coordinated phrases involving parts of speech other than adjectives and adverbs are not supported.*	FN
5–	Què se n'he fet dels seus pares? *Although this sentence is senseless, its structure is grammatical (cf. "Què te n'he dit dels seus pares?"), so the misclassification is due to semantics.*	CFP
7–	Va mirar al seu voltant, però ningú no es manifesta. *This happens because it is beyond parser's capacities to check for correct agreement of verbal tenses, especially when an analogical construct might be grammatical.*	FP
9–	Asseu-te en aquesta taula, que les meves filles t'ha preparat un obsequi que satisfarà la teva gana. *Considered correct because the agreement control is limited within subordinate clauses, besides, the grammar does not recognise subjects within such clauses.*	FP
11+	En Nono es va aturar intimidat, mirant curiosament la dama. *The main disqualifier is the lack of rule enabling past participle to be used as an adjective, unless it is specifically declared so or imported from the corpus.*	FN
13–	Això és el que es demana als habitants del món de la qual véns. *Erroneously classified as correct owing to the inability of the parser to decide whether the relative pronoun has any suitable antecedent or not.*	CFP
18+	Els que anaven arribant col·locaven els seus fruits sobre fruiters de la mateixa porcellana que els plats. *There are two main problems. Firstly, "fruiters" is seen as an adjective and secondly, the comparison introduced with "que" cannot be recognised.*	FN
22–	Amb bona voluntat, es poden arreglar tot. *This reflects the main issue owing to variable position of the subject – to address this in most cases, subjects standing in non-standard positions are classified as objects which do not trigger the verb-subject agreement.*	CFP
26+	Un altre aspecte a destacar és que pot ser practicada per tot tipus de persones. *This case is misclassified because "és" as a copula usually requires the grammatical agreement that the parser wants here between incompatible "aspecte" and "practicada" irrespective of the fact that they are not related, only seeming so because of the dropped subject.*	FN

Table A3.1 (continued)

ID	Sentence	Result
29+	Volen conèixer paisatges de la geografia catalana, però sense haver de caminar gaire. *The structure of composed sentences does not allow an incomplete sentence (the second one lacks an action-bearing verb) to participate in such a composition.*	FN
30–	Consisteix en activitats guiats per aplicar tècniques d'orientació. *This erroneous classification occurs due to lack of restriction that would control what the adjectives are attached to.*	FP
32–	Per això aquest any l'Ajuntament a través de les oficines de la Guàrdia Urbana va recuperar aquesta iniciativa, que ja ha donat molt bons resultats en l'any anterior. *Controlling the correct usage of verb tenses according to adverbs of time is simply beyond parser's capacities.*	CFP
33–	Les persones que no puguin moure el seu vehicle, poden anar una oficina de la Guàrdia Urbana amb els documents del cotxe. *Obviously, the parser does not have capacities to distinguish constructions with a missing preposition. In fact, "anar" could be substituted by a verb which does not require any preposition at all.*	FP
44–	Durant els mesos d'estiu, us proposem que vingueu amb els més petit de la casa a sopar a l'Observatori de Barcelona. *The deficiency that causes this sentence is falsely classified is the object preceded by a preposition that can be split from following adjectives.*	CFP
46+	S'hi ha adaptat molt bé, però sempre es queixa que té pocs diners perquè la vida és molt cara. *In addition to issues seen before, the problem here is due to the adverb of time standing where it is not expected.*	FN
48+	Qui la rebi té la possibilitat de gastar-se els diners en diferents espectacles musicals i teatrals al llarg del cap de setmana. *The initial "qui" has nothing to attach to and since sentences beginning with a subordinate clause or standalone subordinate clauses serving as a subject are not permitted, this is not valid. Besides, the word "cap" is not known as a noun, but only as forming the preposition "cap a" or the determiner "cap."*	FN
50+	Mentrestant, els més menuts podran divertir-se amb els diferents tallers de dolços i pastissos preparats en especial per a ells. *This sentence exhibits the same issue as found before where adjective preceded by a preposition is seen as incorrect unless there is a noun it develops.*	FN

A4. On the Ground Truth Problems

To begin with, the complete list of problems selected to form the Ground Truth is introduced. Each problem is catalogued using a unique number and for each one the expected result is indicated. If the obtained result differs from the correct one, the former is marked in red and the latter in green, plus an explanatory comment is given.

Table A4.1: Summary of problems of the Ground Truth

№	Problem	Result
1	En un autobús, hi ha 65 persones. A la parada següent, hi pugen 14 persones. N'hi baixen 28 persones. Després, hi pugen 15 persones. Quantes persones continuen dins de l'autobús?	66
2	Cada dia en Joan gasta 19 litres d'aigua per dutxar-se. També gasta 23 litres per regar el jardí cada dia. Quants litres d'aigua gasta cada 7 dies?	294
3	La Mercedes ha comprat 7 quilos de carn a 9 euros per quilo. A més a més, ha comprat 5 quilos de botifarres a 4 euros per quilo. Quants euros ha gastat en total?	83
4	En Pere és forner. Dilluns va vendre 350 barres de pa. Dimarts va vendre 34 barres més. Dimecres va vendre 52 barres menys en comparació amb dilluns. Quantes barres va vendre en total?	1032
5	La Gemma va comprar 45 peces de tela. Cadascuna té 108 metres. Ja n'ha venut 3827 metres. Quants metres li queden per vendre?	1033
6	Cada dia, la Maria rep a l'ordinador 163 correus electrònics, dels quals 11 són estrangers. Quants correus electrònics domèstics rep cada 7 dies?	1064
7	La Nuria ha comprat 37 caixes de galetes per a un campament. En cada caixa, hi ha 50 galetes de xocolata. Cada caixa també conté 36 galetes sense xocolata. Quantes galetes ha comprat en total?	3182
8	Una escola ha comprat 12 ordinadors. També han comprat 8 equips de música. Cada ordinador val 1200 euros. Cada equip de música val 200 euros. Quants euros ha costat tota la comanda?	16000
9	En Ramon tenia 50 euros. Va comprar 4 carpetes, de les quals cadascuna va costar 4 euros. També ha comprat 2 llibretes. Cada llibreta va costar 2 euros. Quants euros li va costar la compra? *This problem nicely illustrates the limitations in understanding. The information about the amount of money held is redundant, but as the corresponding item is fully compatible with what it has been ask about, there is nothing else the program can do as it has no reason to disregard this item. On the other hand, if the question is "Quants euros li queden al Ramon?" the program handles this information properly in order to give the correct answer "Al Ramon li queden 30 euros."*	70 i/o 20
10	Un dipòsit, la capacitat del qual és 900 litres, s'ha omplert en 20 minuts. Quants litres raja l'aixeta utilitzada cada minut?	45
11	En una botiga, venen jerseis, dels quals cadascun es ven a 27 euros. Si els venen tots, en tindran 1242 euros. Quants jerseis tenen?	46

Table A4.1 (continued)

№	Problem	Result
12	Una moto ha fet 5 voltes al circuit. Un cotxe n'ha fet 8. La moto ha recorregut en total 21805 metres. Quants metres ha recorregut el cotxe?	34888
13	Un edifici té de 8 pisos en total. A cadascun dels pisos, hi ha 4 apartaments. Cada apartament té 8 finestres. Quantes finestres té l'edifici?	256
14	La Pilar té 4 capses de barres de pa. Cada capsa conté 44 barres de pa. A cada safata, la Pilar hi posa 16 barres de pa. Quantes safates necessitarà la Pilar?	11
15	En Lluís té 2815 fotografies al seu arxiu, dels quals n'ha desat 965 en capses. La resta les ha repartit entre les 5 carpetes que tenia buides. Quantes fotografies ha posat a cada carpeta?	370
16	Un televisor costa 568 euros. Per una promoció, ens descompten 56 euros. Al pagar, els donem 12 bitllets de 50 euros. Quants euros ens hauran de tornar?	88
17	En una capsa de bombons, hi ha 20 bombons de xocolata blanca. També hi ha 15 bombons de xocolata negra. Quants bombons hi ha en 12 capses? *In a way, the incorrect result got for this problem just follows the rigid logic of a machine unendowed with pragmatics. Indeed, there is no word about <u>each</u> box of bonbons, it is only known that in one box there are 20 white chocolate bonbons and that there are also 15 dark chocolate bonbons. Since all other boxes may be empty, it is only known for sure that there are 35 bonbons. Replacing "una capsa" by "cada capsa," the correct calculation 20×12+15 = 255 follows, rephrasing just the second sentence to read "En cada capsa, també hi ha 15 bombons de xocolata negra.", again, the correct calculation 20+15×12 = 200 follows as well. With both "corrections" applied, the solver achieves to come up to the correct amount of 420. Furthermore, just for curiosity, when the problem asks about bonbons of a specific colour, the distinction is taken into consideration, too.*	35 i/o 420
18	En una biblioteca, hi ha 12 prestatgeries. Cadascuna conté 11 prestatges. En cada prestatgeria, hi caben 24 llibres. Quants llibres hi ha a la biblioteca?	288
19	Un excursionista vol completar un trajecte de 180 km. Ha recorregut 3 etapes, de les quals cadascuna tenia 18 km. També ha recorregut 7 etapes, que fan 7 km en total. Quants km li queden per recórrer?	119
20	En David ha cobrat 24 bitllets de 50 euros. A més a més, ha cobrat 15 bitllets de 100 euros en el mes de setembre, que té 30 dies. Quants euros guanya cada dia?	90