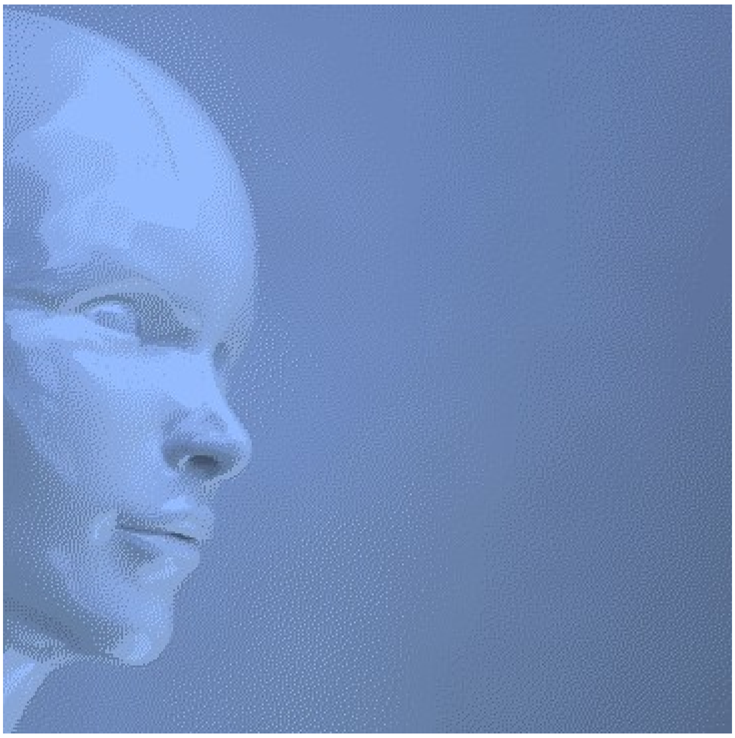
Scientific challenge:

Beat my natural language reasoner



Knowledge technology is based on applying smart algorithms to keywords, by which the natural meaning of non-keywords is ignored.

Non-keywords provide information to our brain about the structure of the sentence. By ignoring the self-organizing function of non-keywords, this field got stuck with "bags of keywords" and unstructured texts. Even IBM's Watson is unable to organize its knowledge base autonomously. Instead, Watson needs raw processing power “to find a needle in the haystack of unstructured texts” (quote from IBM). And at Cycorp, experts are structuring knowledge manually for decades already.

Predicate Logic describes the natural meaning of verb “is/are”, like: “Every father is a man” and “John is a father” from which can be concluded “John is a man”. However, Predicate Logic doesn't describe for example the natural function of verb “has/have”. As a result, knowledge containing this verb to be programmed directly into a reasoner, like: has_son(john,paul).

Programming directly into a reasoner – like has_son(john,paul) – is not a generic solution (science), but a specific solution (engineering). Because you need to program each and every noun directly into the reasoner (has_daughter, has_father, has_mother, and so on), and for each and every new language. As a consequence, there is no technique available to convert a sentence like “Paul is a son of John” in a generic way to “John **has** a son, called Paul”, by which the keywords “Paul”, “son” and “John” don't have to be programmed into the reasoner upfront. It is just the first example of this challenge.

My fundamental approach shows that verb “has” is complementary to verb “is”. So, verb “has” can also be used in predicate logic. In order to utilize the naturally intelligent function (natural meaning) of non-keywords, I have defined natural intelligence first. Then I have identified a few Natural Laws of Intelligence in grammar. By implementing these laws of nature as a set of structuring algorithms, my system is able to structure the knowledge of the system autonomously.

Rules of the game

- There are 5 blocks to beat the most basic techniques of my system. Your implementation should deliver the results of at least one block listed below;
- Your implementation should not have any prior knowledge. Instead, the system should derive its knowledge directly from the input sentences listed below. Preferable: The nouns and proper nouns of the listed examples are unknown upfront;
- Your implementation should be implemented as generic as can be, in such a way that all examples of this challenge can be integrated into one single system. (The reasoning screen shots of my natural language reasoner illustrate how multiple reasoning constructions reinforce each other);
- Your implementation should be published as open source software, so it can be checked. (Also my system is open source: http://mafait.org/download/);
- Your implementation should be accepted by a scientific committee (conference or journal);
- In case your results are slightly different, you need to explain why you have chosen differently;
- It is an on-going challenge, until my system has been beaten on all blocks;
- I am the judge.

Your rewards

- A small gesture from me: € 200 for each block;
- You will be the first one to have these results implemented in a scientifically accepted way, because I didn't use any technique known in this field.

You can contact me via the contact page of my website, or via LinkedIn.

Block 1

Linguistic algebra, applying crucial non-keywords “is”, “has”, “called”, “every” and “part of”:

“{proper noun 1} is {indefinite article + singular noun} van {proper noun 2}”

equals to

“{proper noun 2} has {indefinite article + singular noun} called {proper noun 1}”

“Every {singular noun 1} has {indefinite article + singular noun 2}”

equals to

“{indefinite article + singular noun 2} is part of every {singular noun 1}”

Implementation:

```
> Entered: “Paul is a son of John.”
•
• Generated conclusion:
< “John has a son, called Paul.”
>
> Entered: “Anna has a daughter, called Laura.”
•
• Generated conclusion:
< “Laura is a daughter of Anna.”
>
> Entered: “Every car has an engine.”
•
• Generated conclusion:
< “An engine is part of every car.”
>
> Entered: “A sail is part of every sailboat.”
•
• Generated conclusion:
< “Every sailboat has a sail.”
```

Block 2

Linguistic algebra, applying crucial non-keyword “of”, “is”, “has” and “called”:

“{proper noun 1} is {indefinite article + singular noun} of {proper noun 2}”

equals to

“{proper noun 2} has {indefinite article + singular noun} called {proper noun 1}”

“{proper noun 1} has {indefinite article + singular noun 1} called {proper noun 2}”

and

“{proper noun 1} has {indefinite article + singular noun 1} called {proper noun 3}”

equals to

“{proper noun 1} has {number: 2} {plural form of singular noun 1} called {proper noun 2} and {proper noun 3}”

Implementation:

```
> Entered: “John is a parent of Paul.”
•
• Generated conclusion:
< “Paul has a parent, called John.”
>
> Entered: “Anna is a parent of Paul.”
•
• Generated conclusion:
< “Paul has 2 parent [plural of 'parent' is unknown], called John and Anna.”
>
> Entered: “Paul has 2 parents, called John and Anna.”
•
• Detected that the generated conclusion is confirmed:
< “Paul has 2 parent [plural of 'parent' is unknown], called John and Anna.”
•
• Detected: You have entered plural noun “parents”, which was unknown to me.
```

Block 3

Linguistic algebra, applying crucial non-keywords “of”, “was”, “has”, “had” and “no […] anymore”:

“{proper noun 1} was {indefinite or definite article + singular noun} of {proper noun 2}”

equals to

“{proper noun 2} has no {singular noun} anymore”

“{proper noun 1} was {indefinite or definite article + singular noun} of {proper noun 2}”

equals to

“{proper noun 2} had {indefinite article + singular noun} called {proper noun 1}”.

Implementation:

> Entered: “James was the father of Michael and Peter.”
•
• Generated conclusions:
< “Michael has no father anymore.”
< “Michael had a father, called James.”
< “Peter has no father anymore.”
< “Peter had a father, called James.”

Block 4

Linguistic algebra, applying crucial non-keywords “every”, “is”, “or”, “and” and “not”:

“Every {singular noun 1} is {indefinite article + singular noun 2} **or** {indefinite article + singular noun 3}”

conflicts with

“{proper noun 1} is {indefinite article + singular noun 2} **and** {indefinite article + singular noun 3}

“Every {singular noun 1} is {indefinite article + singular noun 2} or {indefinite article + singular noun 3}”

and

“{proper noun 1} is {indefinite article + singular noun 1}”

from which can be concluded

“{proper noun 1} is {indefinite article + singular noun 2} or {indefinite article + singular noun 3}”

“{proper noun 1} is {indefinite article + singular noun 2} or {indefinite article + singular noun 3}”

equals to

“Is {proper noun 1} {indefinite article + singular noun 2} or {indefinite article + singular noun 3}?”

“Is {proper noun 1} {indefinite article + singular noun 1} or {indefinite article + singular noun 2}?”

and

“{proper noun 1} is not {indefinite article + singular noun 1}”

from which can be concluded

“{proper noun 1} is {indefinite article + singular noun 2}”

“Is {proper noun 1} {indefinite article + singular noun 1} or {indefinite article + singular noun 2}?”

and

“{proper noun 1} is not {indefinite article + singular noun 2}”

from which can be concluded

“{proper noun 1} is {indefinite article + singular noun 1}”

Implementation:

```
> Entered: “Every person is a man or a woman.”
> Entered: “Addison is a man and a woman.”
!
! Detected conflict: This sentence is not accepted, because it is in conflict with itself.
!
> Entered: “Addison is a person.”
•
• Generated question:
< “Is Addison a man or a woman?”
>
> Entered: “Addison is not a woman.”
•
• Detected that the generated question has been answered:
< “Is Addison a man or a woman?”
•
• Generated conclusion:
< “Addison is a man.”
```

Block 5

Linguistic algebra, applying crucial non-keywords “the”, “is”, “has”, “called”, “previous” and “new”:

“{proper noun 1} is {**definite** article + singular noun} of {proper noun 2}”

equals to

“{proper noun 2} has {indefinite article + singular noun} called {proper noun 1}”

“{proper noun 1} is {**definite** article + singular noun} of {proper noun 2}”

and

“{proper noun 3} is {**definite** article + singular noun} of {proper noun 2}”

equals to

“{proper noun 2} has {indefinite article} new {singular noun} called {proper noun 3}”

and

“{proper noun 2} has {indefinite article} previous {singular noun} called {proper noun 1}”

Implementation:

> Entered: “Bill Clinton is the president of the United States.”
•
• Generated conclusion:
< “The United States has a president, called Bill Clinton.”
>
> Entered: “Barack Obama is the president of the United States.”
•
• Generated conclusions:
< “The United States has a new president, called Barack Obama.”
< “The United States has a previous president, called Bill Clinton.”