Ambiguity, unknowingness, incompleteness, incorrectness

in analysis of ancient Sumerian texts

Wojciech Jaworski
Warsaw University

Warsaw, December 17 2005

Example of tablet

```
&P123831 = OIP 121, 101
@tablet
@obverse
1. 1(disz) sila4 ur-mes ensi2
2. 1(disz)# sila4 da-da dumu lugal
3. 1(disz)# sila4 id-da-a
@reverse
1. u4 2(u) 3(asz@t)-kam
$ 1 line blank
3. mu-DU
4. ab-ba-sa6-ga i3-dab5
5. iti sze-KIN-ku5
6. mu en {d}inanna ba-hun
@left
1. 3(disz)
```

Main obstacles

Ambiguity: Sequence of signs can be interpreted in a few different ways that provide different meanings.

Unknowingness: The computer must be learned Sumerian language.

Incompleteness: Parts of the tablets are broken, so the texts are incomplete.

Incorrectness: Some the texts was written or transliterated with errors.

The idea of algorithm

- We performed syntax analysis using non context grammar without recursion.
- For each text's subsequence we find all its possible interpretations and add them to text as new terms.
- Then we use the enriched text for finding next level interpretations.

Data representation

- We need representation that can describe ambiguous, partially interpreted text.
- We use the directed acyclic graph whose each edge is labeled by term.
- While applying the rule we find path in graph and add to it new edge from beginning to end of path labelled with term constructed according to rule and terms on the path.

Sample rule set

<disz> ::= 1(disz) | 2(disz) | ... | 9(disz)

<u> ::= 1(u) | 2(u) | ... | 5(u)

<gesz2> ::= 1(gesz2) | 2(gesz2) | ... | 9(gesz2)

[u] ::= <gesz2> | <gesz2> <u>

[number] ::= [u] | [u] <disz> | [u] la2 <disz>

Learning

- Up to now we discovered how to analyze syntax of ambiguous text using non context grammar.
- The question is: How to generate the grammar rules?
- The process of generating rules is the process of learning (acquiring knowledge).
- Our tests showed that it is impossible to discover grammatical rules of language without using its semantic.
- Although it is possible that if computer understood one language it would learn the other one without human tuition.
- In a present state of art grammar must be input manually.

Teaching the Computer Reader

- It is theoretically possible to teach the Computer Reader inputting rules for each table, yet it is time consuming and provides rules that are not properly generalized.
- Sets of similar rules should be generated automatically without need of inputting its content.
- We derived special interface that allows to create rules in reasonable time.

Damaged tablets

- The simplest method of working with damaged tablets is to treat broken fragments as unknown ones. It is quite efficient method since the non context rules are local.
- More elaborated tactics by the way the information about damages is provided.
- They are provided in following ways:
  - One sign is unreadable
  - Part of verse is unreadable
  - n verses are unreadable
  - Big part of tablet (many verses) are unreadable

Damaged tablets

One sign unreadable We interpret the damaged sign into every possible sign.

n **verses are unreadable** We interpret each of damaged verses into every typical parsed verse. Here the hierarchical approach is necessary.

Part of verse is unreadable We interpreter it into every typical parsed verse. Then we try to connect terms in verse with its hypothetical interpretation and reject the interpretations that doesn’t suite.

Semantics

- Semantics must be written in some formal language.
- Such a language must have possibly big power of expression, yet it must be algoritmizable.
- As the consequence the best candidate for language of semantics is programming language.
- Functional programming languages are similar to pseudocode of *denotational semantics* and λ-*calculus*.
- Since whole system is written in *caml* the semantics also will be written in *caml*.

Example: Semantic of Year Names

$$[\![\texttt{en}]\!] = \{AS04, AS05, AS08, AS09\}$$

$$[\![\texttt{\{d\}nanna}]\!] = \{AS04, AS09\}$$

$$[\![\texttt{ba-hun}]\!] = \{AS04, AS05, AS08, AS09\}$$

$$[\![\textit{unknown term}]\!] = \{AS01, AS02, \ldots, AS09\}$$

$$[\![x_1\ x_2 \ldots x_n]\!] = \bigcap_{i=1}^{n} x_i$$

For example:

$$[\![\texttt{mu en \{d\}nanna ba-hun}]\!] =$$

$$= [\![\texttt{mu}]\!] \cap [\![\texttt{en}]\!] \cap [\![\texttt{\{d\}nanna}]\!] \cap [\![\texttt{ba-hun}]\!] =$$

$$= \{AS04, AS09\}$$

Composing semantics of terms into semantics of text

We found two levels of semantics:

- Text's semantics as a sequence of semantics of short fragments such as Animal Descriptions, Year Names etc. Useful for gaining information from partially unknown and/or damaged texts.
- Semantics of the complete event described in text. It places the text into the model of Sumerian reality.

Attributes

Simplifying a little bit we have following information located in texts:

- Animal Descriptions,
- Description of the person who gave animals,
- Description of person who took animals,
- Dates.

We have also implicit information whether on tablet is written every transaction made by given person during given time interval.

‘Understand’ means ‘situate in the model of world’

- Each day is a time point
- For every day we have graph whose vertexes are labeled by Persons and edges are labeled by Animals passed on.
- Each text describes part of the model.
- The model is further complicated by the fact the in transaction can take part overseer, it may by made via middleman or the last receiver may be mentioned.

Conclusions

The presented system

- is a novel approach in the **Text Mining** field.
- It **Manage Knowledge** about natural language.
- The **Knowledge** is **discovered** from the **data**
- during **interaction** with an **expert**
- by means of interface that **visualize data**.
- The system deals with the problem of **incomplete knowledge**
- as well as with **incompleteness**, **incorrectness** and **ambiguity** of data.
- The system is also a proposition for method of constructing formal, constructive semantics for natural language.