

Are computers capable of thought?

Discover it

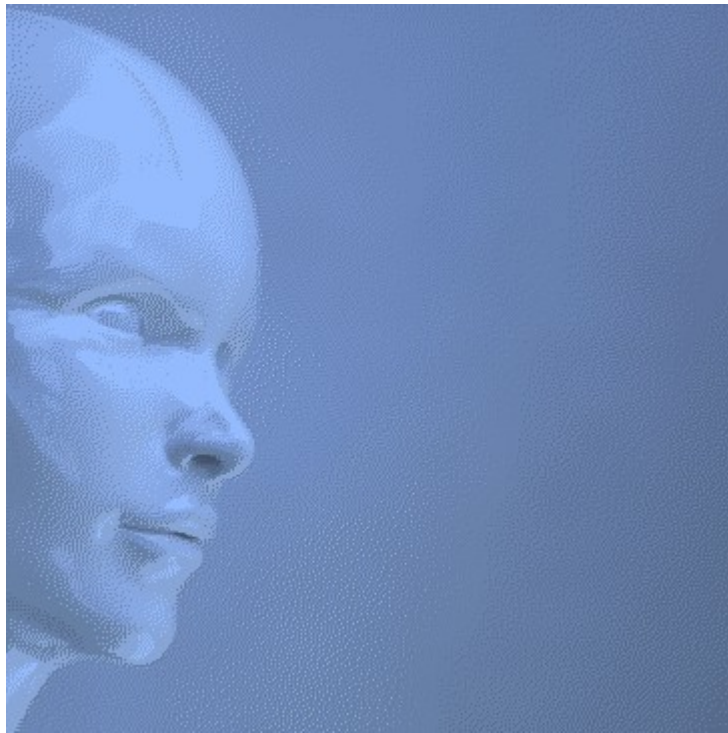


Table of Contents

1. Introduction.....	4
1.1. The Thinknowlogy Concept.....	4
2. The Algebra and Logic of Natural Language.....	5
2.1. A Definition of Generalization / Specification.....	5
2.1.1. A Relational Specification.....	5
2.1.2. Collections.....	6
2.1.2.1. A Collection of Specifications.....	6
2.1.2.2. A Collection of Relations.....	6
2.1.2.3. A Collection of Generalizations.....	6
2.1.3. Assignments.....	7
2.1.3.1. A Specification Assignment.....	7
2.1.3.2. A Generalization Assignment.....	7
2.1.3.3. Assignments and grammatical Past Tense.....	8
2.2. Selections.....	9
2.3. Drawing Conclusions Independently.....	10
2.3.1. Specification Substitution Conclusions.....	10
2.3.1.1. Compound Specification Substitution Conclusion.....	11
2.3.2. Possessive Reversible Conclusions.....	12
2.4. Making Assumptions.....	13
2.4.1. Generalized Assumptions.....	13
2.4.2. Possessive Conditional Specification Assumptions.....	14
2.5. Asking Independent Questions.....	15
2.5.1. Questions to ask to obtain more specific information.....	15
2.6. Finding Causal Relations.....	15
3. Building the Knowledge-Structure.....	16
3.1. The Creation of a Generalization Specification.....	16
3.1.1. The Creation of New Words.....	17
3.1.2. The Creation of a Specification Connection.....	18
3.1.2.1. The Creation of a Relation Connection.....	18
3.1.3. The Creation of Collections.....	19
3.1.4. The Creation of an Assignment.....	19
3.1.4.1. The Creation of a Specification Assignment.....	20
3.1.4.2. The Creation of a Generalization Assignment.....	20
3.2. The Creation of a Selection.....	22
3.2.1. The Execution of Selections.....	22
4. Dialogues.....	23
4.1. The Grammar.....	23
4.1.1. Reading a Sentence.....	23
4.1.1.1. Acceptation of Recognized Sentence Structures.....	24
4.1.2. A Complete Sentence Answer.....	24
4.1.2.1. Asking Questions Independently.....	24
5. Applications.....	25
5.1. Programming in natural language.....	25

5.2. Reviewing and Correcting Texts and Specifications.....25
5.3. The Automation of Summaries.....26
5.4. Translation Programs that Understand the Context.....26
5.5. A Search Engine that Answers Questions.....26
5.6. A Digital Teacher or Independent Studies.....27

1. Introduction

- Is it possible to attain more from computer software than just improved computations, the storage of your files and the automation of business processes?
- Will we ever be able to make internet searches by asking questions of our search engines instead of using single or a combination of single words?
- Will computers ever be able to deal with natural language?
- Computers are able to process data. But will computers ever process information? In other words: Will we ever be able to automate information?

Many attempts have been made to have computers 'think', but without a concept as basis for these attempts, these remain useless. With a well thought out concept as base, you have a chance to achieve the goal.

1.1. *The Thinknowlogy Concept*

The Thinknowlogy Concept is based upon the following rules:

- 1) Programming languages are based upon algebra and logic.
- 2) Every person has an inborn feel for algebra and logic, although it is more developed in one person than another. This is communicated for the most part through the use of natural language.
- 3) In combining algebra and logic in natural language with algebra and logic of programming language, it is possible to program using natural language.
- 4) The final goal is the automation of information.

In the first place it is researched how the algebra and logic of natural language can be combined with the algebra and logic of programming, so that programming using natural language is possible.

Later the automation of information is researched.

2. The Algebra and Logic of Natural Language

Several aspects of the algebra and logic of Natural Language are described here below.

2.1. A Definition of Generalization / Specification

A person organizes his or her thoughts by grouping things together; this we call generalization. The meaning of generalization is actually: Separating the main point from supporting facts. The main points are called the **generalizations** and the supporting facts are the **specifications**.

The process of separating the generalizations from the specifications begins with the most basic use of language. For example, use the sentence: “*John is a person.*”.

“*John*” is the subject and “*person*” is the adjective, which tells more about who “*John*” is. In other words, “*John*” is the generalization of the specification “*person*”, because the word “*person*” describes something about “*John*”.

More specifications of John:

- “*John is a father.*”
- “*John is a baker.*”

In the above mentioned sentences the article “*a*” has been used: “*John is a person.*”, “*John is a father.*” and “*John is a baker.*”. In the Thinknowlogy Concept we call this structure a **generalization / specification definition**.

2.1.1. A Relational Specification

Take the example: “*John is the friend of Marc.*”.

Generalization “*John*” has a “*friend*” relation(ship) with “*Marc*”. In this example, “*friend*” is called a **specification** and “*Marc*” the **relation**. Therefore, this is called a **relational specification**.

2.1.2. Collections

How words form a collection is explained below.

2.1.2.1. A Collection of Specifications

In a generalization / specification definition, a generalization can have several specifications that are linked together, such as: “*A glass is full or empty.*”. Or more specific: “*A glass is full, half full or empty.*”.

We name such a group of specifications, a **collection of specifications** or a **specification collection**.

2.1.2.2. A Collection of Relations

Take the example: “*John is a friend of Marc and Paula.*”.

In this case the related persons, “*Marc*” and “*Paula*” are collected together to form what we call a **relational collection**.

2.1.2.3. A Collection of Generalizations

Take the following examples:

- “*Bush is the previous president of the United States of America.*”
- “*Obama is the current president of the United States of America.*”

In this case the specification and relation are the same but the generalizations are different. Because “*Bush*” and “*Obama*” have something in common, these generalizations can be combined.

This we call a **collection of generalizations** or a **generalization collection**.

2.1.3. Assignments

An **assignment** is a generalization / specification with an allocated value. A generalization / specification is generally static, whereas an assignment is dynamic because the situation can change.

An assignment provides the actual situation of a generalization / specification and is identifiable by a **definite article** in the sentence.

2.1.3.1. A Specification Assignment

Take the example: “*The glass is half full.*”.

This is a generalization / specification structure in which a definite article is used and the specification belongs to a specification collection, like in: “*A glass is full, half full or empty.*”. We call the first example a **specification assignment**.

An assignment defines the actual situation and can, therefore, be changed. For example if a glass of water is drunken empty than there is a new situation: “*The glass is empty.*”.

An example of an assignment of a relational specification is: “*John is the father of Pete.*”.

2.1.3.2. A Generalization Assignment

Take the following examples:

- “*Bush is the previous president of the United States of America.*”
- “*Obama is the current president of the United States of America.*”

These are also assignments because they use a definite article. The generalizations are in this case different and form a generalization collection, For that reason these assignments are called **generalization assignments**.

2.1.3.3. Assignments and grammatical Past Tense

When the assignment has changed, in linguistics we name the former situation: **Past Tense**.

Sometime in the past the situation of the glass was: “*The glass **is** half full*”, but the glass has now been drunken empty and now we say:

- “*The glass **was** half full.*”
- and/or “*The glass **is** (**now**) empty.*”

Sometime in the past the situation was: “*Bush **is** the president of the United States of America.*”; but the situation has been changed since the elections to:

- “*Bush **is** the **previous** president of the United States of America.*” or “*Bush **was** the president of the United States of America.*”
- “*Obama **is** the (**current**) president of the United States of America.*”

2.2. Selections

Natural language also contains selections, such as in the sentence, “*If the traffic light is yellow or red, you must stop.*”. Under the condition “*the traffic light is yellow or red*” the action “*you must stop*” applies here.

Often there are also alternatives: Otherwise, if the traffic light is green, “*You must ride ahead.*” applies here. Combined: “*If the traffic light is yellow or red you must stop, otherwise you must ride ahead.*”.

A selection is made up of two or three parts:

- A condition (“*If the traffic light is yellow or red*”),
- An action (“*You must stop.*”),
- and possibly an alternative action (“*You must ride ahead.*”).

The following generalization / specification structures apply here:

- The condition: “*A traffic light is yellow or red.*”;
- The action: “*You must stop or ride ahead.*”.

Conditions as well as the actions are assignments:

- The assignments of a condition are used to decide whether the defined condition is correct. The condition of the example is only true when one of the following assignments applies: “*The traffic light is yellow.*” or “*The traffic light is red.*”;
- The assignment of the action or alternative action depends on that result, thus, “*You must drive ahead.*” or “*You must stop.*”.

2.3. Drawing Conclusions Independently

We are not used to software that can draw its own conclusions. Below is an explanation of how this can be made possible.

2.3.1. Specification Substitution Conclusions

Consider the following sentences:

- “*John is a father.*”
- “*A father is a man.*”

First, a comment: the last sentence we call a **definition sentence** because the generalization is formed through a indefinite article and a noun. In addition the specification is formed by one or more indefinite articles and nouns.

From these sentences we can conclude that: “*John is a man.*”.

This conclusion can be automated by applying the following rules:

- 1) Take the sentence in which the singular specification is formed by way of a indefinite article and a noun;
- 2) With that specification, search the definition sentences in which the specifications (identified above) are the same as the generalization of the definition sentence;
- 3) In the case of the first sentence, the specification is applicable to every associated definition.

So, in the case of the above mentioned sentence:

- 1) Take the sentence about “*John*”. The (singular) specification of that sentence is: “*a father*”;
- 2) For that specification the applicable definition is: “*A father is a man.*”;
- 3) The software can now independently draw the conclusion that: “*John is a man.*”, by replacing the specification of the sentence about “*John*” with the specification of the definition sentence.

We call this a **specification substitution conclusion** because the specification of a proper noun sentence can be replaced by the specification of a definition sentence.

2.3.1.1. Compound Specification Substitution Conclusion

Take the sentences:

- “*A parent is a father or (a) mother.*”
- “*A father is a man.*”
- “*A mother is a woman.*”

According to the (singular) specification substitution conclusion the singular specification of a sentence may be replaced with the specification of a definition sentence. Also in the case of a sentence with a compound specification such as “*A parent is a father or (a) mother.*”, the specifications may be replaced as long as there is a definition for both.

By replacing both the specifications “*father*” as well as “*mother*” by using the terms “*man*” and “*woman*”, it could be concluded (from that specific sentence) that “*A parent is a man or (a) woman.*”.

Such a conclusion we call a **compound specification substitution conclusion** because the compound specification is replaced by another definition.

Another form of a compound specification-substitution conclusion is formed when a singular specification is replaced by a compound specification of a definition sentence, such as in these sentences:

- “*Pete is a child (of John).*”
- “*A child is a son or (a) daughter.*”

Here the conclusion is: “*Pete is a son or (a) daughter (of John).*”.

2.3.2. Possessive Reversible Conclusions

Take the sentence: “*John is the father of Pete.*”

From this sentence we conclude that: “*Pete has a father (named John).*”.

First of all, a conjugation of the verb “*to have*” is named (here below) a **possessive verb**, because it indicates the ownership of the generalization.

The rules which apply to the **Possessive Reversible Conclusion** are:

- 1) In the sentence, the generalization as well as the relation must be a proper noun. In making the conclusion these must be exchanged;
- 2) The sentence must include a prior term of the verb “*to be (am / is / are)*”, which in the conclusion is to be replaced by a prior term of the possessive verb “*to have*”;
- 3) The sentence must have a singular specification with a noun. If, in the specification, a definite article is used, then it must be replaced by an indefinite article in the conclusion.

Now to apply the rules to the above mentioned sentence:

- 1) The generalization “*John*” and the relation “*Pete*” are proper nouns and are exchanged;
- 2) The verb “*is*” is replaced by the possessive verb “*has*”;
- 3) The singular specification “*the father*” contains a noun. The definite article “*the*” is replaced by the indefinite article “*a*”.

The result is: “*Pete has a father (named John).*”.

2.4. Making Assumptions

Aside from conclusions, we can also make assumption on the basis of the given information, although the validity is not secured. Below is a description of how assumptions can be made automated.

2.4.1. Generalized Assumptions

Take this sentence first: “*A parent is a father or (a) mother.*”.

By using the conjunction “*or*” the first sentence can be dissected in two singular definitions:

- “*A father is a parent.*”
- “*A mother is a parent.*”

Now take the following sentences:

- “*A father is a parent.*”
- “*John is a father.*” or “*John is the father of Pete.*”

It is now clear that the specification word of the sentence about “*John*” is the same as the generalization word of the definition sentence. Therefore, we may make a specification substitution conclusion.

So from the original sentence, “*A parent is a father or (a) mother.*”, and the sentence, “*John is the father of Pete.*”, we can conclude that “*John is a parent (of Pete).*”.

This 'conclusion' we call a **generalized assumption**, because the noun specification of a sentence may be replaced by the generalization of an exclusive specification collection of an (original) definition sentence.

A generalization often has exceptions, therefore it is an assumption and not a conclusion.

2.4.2. Possessive Conditional Specification Assumptions

Take the sentences:

- “A family *has parents and children.*”
- “John is a *parent of Pete.*”

Two comments about the first sentence:

- 1) This sentence has a prior term of the possessive verb “*to have*”;
- 2) The conjunction “*and*” indicates that **both** specifications are necessary to validate the definition (condition).

Two comments about the second sentence:

- 1) We can read this relational specification as “*John*” has a “*parent(al)*” relation with “*Pete*”;
- 2) However, the reverse relation(ship) from “*Pete*” to “*John*” is unknown, in which case we may not make any conclusions about “*Pete*” although we would like to claim that:
 - “*Pete is a child of John.*”
 - “*John has a child (named Pete).*”

However, we would still like to do something with this information as it can provide a wealth of information, if the above mentioned assumptions about “*Pete*” are true. If these assumptions prove to be false, there is in any case clarity created in the situation and the information can be more specified, which we strive to realize.

We may make **possessive conditional specification assumptions**:

- 1) When there is a prior term of the possessive verb “*to have*” used;
- 2) When there is a **conditional** specification collection;
- 3) By using the word, “*children*”, a relationship with “*parents*” is at the moment the only possible relation that can be reversed;
- 4) If later on other relationships appear possible, the assumptions in question will need to be tested;
- 5) If we take all conclusions, that can be deducted from an assumption, and name them assumptions until the assumption is confirmed or not, then all primary assumptions must be revisited;
- 6) If an assumption is confirmed, then it receives the status of a conclusion or proof;
- 7) If, on the basis of a compound specification substitution, a conclusion can be made, then we do not call this an assumption but a question.

2.5. Asking Independent Questions

If there is information missed in the system, it is possible to have the system automatically ask the user questions. Conflicting information can also be identified and through the asking of questions can be corrected.

Here below is a description of how to stimulate the obtaining of more specific information.

2.5.1. Questions to ask to obtain more specific information

Earlier, we saw the conclusion: *“Pete is a son or (a) daughter (of John).”*.

However, such a conclusion is a bit vague because of its unspecified choice indicated by the conjunction *“or”*. To stimulate the obtaining of more specific information, we can present the conclusion as a question. To do that we must place a prior tense of the verb *“to be (am / is / are)”* in the beginning of the sentence and end it with a question mark.

The result is: *“Is Pete a son or (a) daughter (of John)?”*.

2.6. Finding Causal Relations

It is possible to automatically find causal relations in a given text, through which the system can draw conclusions based upon the known information. This will be defined later.

3. Building the Knowledge-Structure

A person with some programming experience would probably have recognized some basic principles of programming language in the theory explained here above: An assignment, an **if-then-else** structure in the form of a selection and possibly also a **declaration of a variable** in the form of a generalization / specification definition. A declaration gives structure to a variable, but the variable isn't assigned to a value yet.

With these similarities between natural language and programming language a knowledge structure can be built. Through this - a connection between natural language and programming language is made and it should therefore be possible to program in natural language. That serves as a base or foundation of a "thinking" computer.

Here below is an explanation of each part of how the knowledge structure using natural language can be built.

3.1. The Creation of a Generalization Specification

An important primary (founding) element is the generalization / specification. Various aspects of this structure and their importance in building the knowledge structure are defined here below.

3.1.1. The Creation of New Words

Take the sentence: “*John is a father.*”.

The system must first recognize the words, “*John*” and “*father*” before the structure can be built up. Therefore, if the words do not exist in the system, or if they are not of the correct grammatical type, these words must first be created.

John

The word “*John*” begins with a capital letter and is the first word in the sentence. This could be a proper noun or another grammatical type as the original word does not need to begin with a capital letter.

In this case, two words are created:

- “*John*” as a *proper noun*;
- “*john*” (without capital) as an unknown grammatical type.

In the building up of the knowledge structure only one of the two words is used. After which, the unused word is deleted so that the system does not become cluttered. Later we will discuss this further.

father

The word “*father*” is preceded by an article, so the grammatical type is clear: a noun.

In the sentence, “*A glass is full, half full or empty.*” it is not clear what grammatical type the words, “*full*”, “*half full*” and “*empty*” are. It is assumed, however, that they are all three of the same grammatical type.

3.1.2. The Creation of a Specification Connection

Now that the system knows the words, the words can be connected to each other. In other words, the generalization word is connected with the specification word.

For example: “*John is a father.*” and “*John is a baker.*”.

In the first sentence, from the generalization word, “*John*” a **specification connection** is made to the specification word, “*father*” and in the second sentence from the word, “*John*” to the word “*baker*”.

In the knowledge structure it is known that John is a father as well as a baker.

3.1.2.1. The Creation of a Relation Connection

In order to add the relational specification, “*John is the friend of Marc.*” to the knowledge structure the generalization word “*John*” must be connected to specification “*friend*” through a specification connection containing a reference to relation “*Marc*”.

Thus there is one specification connection used containing a second reference. This special specification-connection is called a **relation specification connection**, or a **relation connection** for short.

The relation connection must be read as: “*John*” has a “*friend*” relation with “*Marc*”. The same goes for “*Bush*” and “*Obama*” who have a “*president*” relation with the “*United States of America*”.

In the sentence, “*John is the friend of Marc and Paula.*”, two relation connections are created stemming from “*John*”. Both include the specification “*friend*”, but one has a relation reference to “*Marc*” and one to “*Paula*”.

3.1.3. The Creation of Collections

Take the sentence: “*A glass is full, half full or empty.*”.

The specification **words** of generalization “*glass*” are ‘collected’. This is done by combining them through a collection-combination:

- “*full*” is connected with an ascending collection-connection to the word “*half full*”;
- “*half full*” is connected with an ascending collection-connection to the word “*empty*”;
- “*empty*” is connected with a descending collection-connection to the word “*half full*”;
- “*half full*” is connected with a descending collection-connection to the word “*full*”.

The ascent or descent of the collection-connection indicates the order in which it is added.

The collecting of generalizations and relations happens in the same manner:

The collecting of generalisation words

- “*Bush is the previous president of the United States of America.*”
- “*Obama is the current president of the United States of America.*”

The collecting of relation words

- “*John is the father of Pete and Mary.*”

3.1.4. The Creation of an Assignment

An assignment is a generalization / specification structure with an assignment value. Therefore a generalization specification is first created, if it does not exist already, and then the assignment structure follows.

3.1.4.1. The Creation of a Specification Assignment

A specification assignment with a specification collection can only allocate one of the situations at a time when the conjunction “or” is used in the sentence.

Take the sentence “*A glass is full, half full or empty.*” and the situation (assignment): “*The glass is half full.*”.

If the glass is then emptied, the situation changes to, “*The glass is empty.*”. The assignment connection from “*glass*” to “*half full*” is then deactivated, through which this assignment is interpreted as being in the grammatical Past Tense form. After which an assignment connection from the word “*glass*” to “*empty*” is created in order to record that the situation of the glass is “*empty*” now.

Multiple assignments per Generalization

A specification assignment can have several assignments. For example, two assignments can be true at the same time, “*The glass is empty.*” and “*The glass is blue.*”.

An assignment with multiple relations

In the case of a specification assignment with more than one relation, a specification assignment is created for each relation.

Two comments:

- In the sentence, “*John is the father of Pete and Mary.*”, the conjunction “and” indicates that both relations are joined together and can both be true at the same time;
- Even if you later say later, “*John is the baker of Paul.*”, the above mentioned assignments remain active, as “*father*” and “*baker*” do not form a collection.

3.1.4.2. The Creation of a Generalization Assignment

Also generalization assignments can represent one situation at a time, because they are joined through a generalization collection:

First the assignment, “*Bush is the president of the United States of America.*”, was true and after the elections, “*Obama is the president of the United States of America.*”, is true. In that case the assignment of “*Bush*” to “*the president of the United States of America*” is deactivated, through which it is interpreted as in the grammatical Past Tense form, and consequently an assignment of “*Obama*” to “*the president of the United States of America*” is created.

3.2. The Creation of a Selection

Just as the generalization / specification the selection is also an important founding element of the knowledge-structure.

In order to build up the knowledge-structure of a selection, all generalization / specification structures must first be created from which the selection is built. Next the condition in the [condition list](#) the action in the [action list](#) and the alternative action in an [alternative action list](#) are all saved.

Take the sentence: *“If the traffic light is yellow or red, you must stop, otherwise you must drive ahead.”*.

First the generalization / specifications, *“The traffic light is yellow or red.”*, *“You must stop.”* and *“You must drive ahead.”* are created. Next, the sentence *“The traffic light is yellow or red.”* must be added to the list of conditions and be assigned to the related action and the alternative action. *“You must stop.”* would be added to the action list and *“You must drive ahead.”* would be added to the alternative action list.

3.2.1. The Execution of Selections

After the system has read and processed a sentence, all conditions in the knowledge structure are reviewed by the system and the corresponding action or alternative action is processed (executed). This process is repeated until there are no more changes in the system. The system is then in a state of rest and ready to read and process the next sentence.

In this way it seems as though the system is ‘thinking’, but it is actually only using programming that is written in natural language.

4. Dialogues

Here above we have translated sentences using natural language into a knowledge-structure. However, in order to hold a conversation with the user, the system must be able to answer in understandable sentences. To do this, the created knowledge structure must be translated again into natural language.

4.1. *The Grammar*

In order to both read and answer questions in a particular language, the system must be able to recognize the language. The grammar of each language must therefore first be defined.

To ensure the system remains flexible, the grammar is saved as a text file. One or more of the grammar files are read during the startup of the system.

It is also possible to converse in several languages and it should ultimately also be possible to have the system automatically translate. More information of this will follow.

4.1.1. Reading a Sentence

When reading a sentence, the grammar is used to:

- control whether or not a sentence fulfills the defined grammar;
- make a distinction between the different structures, such as generalization / specifications, assignment and selections;
- define specific words initially (more about that later).

4.1.1.1. Acceptation of Recognized Sentence Structures

In the earlier explanations, we saw that in order to change sentences into a knowledge-structure, a new concept, per sentence structure (generalization / specifications, assignments and selections), is necessary to translate the information from the sentences into the knowledge-structure. We have also seen that these sentence structures are saved in a grammar file.

Thus, the system can only accept sentence structures that are saved in a grammar file, because the system only has the concept for the specific information of that type of sentence, which is needed to translate it into the knowledge-structure. This is definitely a limitation of the system as we would prefer that it could read all possible sentence structures and be able to translate them into the knowledge-structure.

The ultimate goal of this project is to develop as many sentence structure concepts as possible, so they will translate the specific sentence structures into the knowledge-structure.

It will obviously be a difficult and lengthy process to fulfill this goal, however the results will ensure the possibility of automatic processing and the reality of a 'thinking' computer comes nearer.

4.1.2. A Complete Sentence Answer

The grammar is used to place the words, which the system wants to present to the user, in a grammatically correct sentence.

4.1.2.1. Asking Questions Independently

The system can identify gaps and contradictions within the knowledge structure, but more of this later.

Aside from this, the system can independently ask the user questions about identified gaps or contradictions within the saved information in order to be able to provide more information, corrections or explanations. This appears to be using intelligence and suggests that computers are capable of thought.

5. Applications

If the explained theory is adequately developed, then it is possible to have computers process and “understand” information in text form such as text on the internet, business documents, school books and technical designs of software developers.

Below is a list of possibilities for which this system can be used in the future if the grammar of the system can adequately process the sentence structures.

5.1. Programming in natural language

Through the selection concept, it is possible to have the system automatically perform tasks which are given in natural language. In this way, you can program in natural language. Therefore it would be possible to enter in the rules of a game, after which you could play the game against the computer, instead of having to program the game into the computer.

Software development

A more important example is the possibility of entering the Technical Design, of a new software development, to a certain extent without having to do any programming. In its most primitive form it could be used as a sort of prototype system, but in this way the automation of software development should ultimately also be possible.

This is actually the automation of automation. Food for thought: Is it cannibalism or inevitable progress?

5.2. Reviewing and Correcting Texts and Specifications

The system is able to find gaps and contradictions, even ones that man may not readily think of. It is possible for the system to identify and solve these by asking specific questions. Eventually the systems should be able to review and correct text and specifications.

5.3. *The Automation of Summaries*

The making of a summary is actually nothing more than separating the main point from the supporting facts, in which some supporting facts are sequentially omitted, until the text is of desired size.

Having a summary of a document be automatically made is especially helpful for managers going into meetings and will be using documents of a large size. In this way, it is possible for them to easily make a summary in the preferred document size, while ensuring that the audience is adequately informed about the contents of the document.

5.4. *Translation Programs that Understand the Context*

Most translation programs translate literally (word for word), but do not recognize sayings or quotes. Automatic translation programs are often of poor quality as many words have multiple meanings, and the translation programs do not understand the context in which the text is placed.

This system should be able in the future to “understand” the context and therefore can bring the automatic translation components to a higher level.

5.5. *A Search Engine that Answers Questions*

At present, we search the internet by carefully choosing key words, hoping that the use of those words will provide the appropriate web pages to answer our questions.

Using this manner, it is relatively easy to find the meaning of difficult words or terms, but the process becomes more difficult if we have questions for which specific expertise is needed to be able to answer them and you are unsure of the correct terminology or appropriate context.

And, in which language can you best find the information? In many cases you may try in English. But what if you don't know the correct translation or the information is only available in Chinese? How then do you get your questions answered if you do not speak the Chinese language?

Only in the case of a search engine that “understands” the information in the correct context is it possible to ask questions.

5.6. A Digital Teacher or Independent Studies

At the beginning of the school year, the teacher decides which chapters and topics will be covered after reviewing the textbooks and knowing the allotted hours. During each lesson the teacher gives a summary of the information which will be covered. Because the teacher understands the material thoroughly, the teacher is able to answer additional questions about it.

If it is possible both to make automatic Summaries (of the covered material) and to have Search Engines answer questions, then the teaching of a theoretical subject could be automated or in any case a sufficient digital teacher could be offered as help with independent studies.