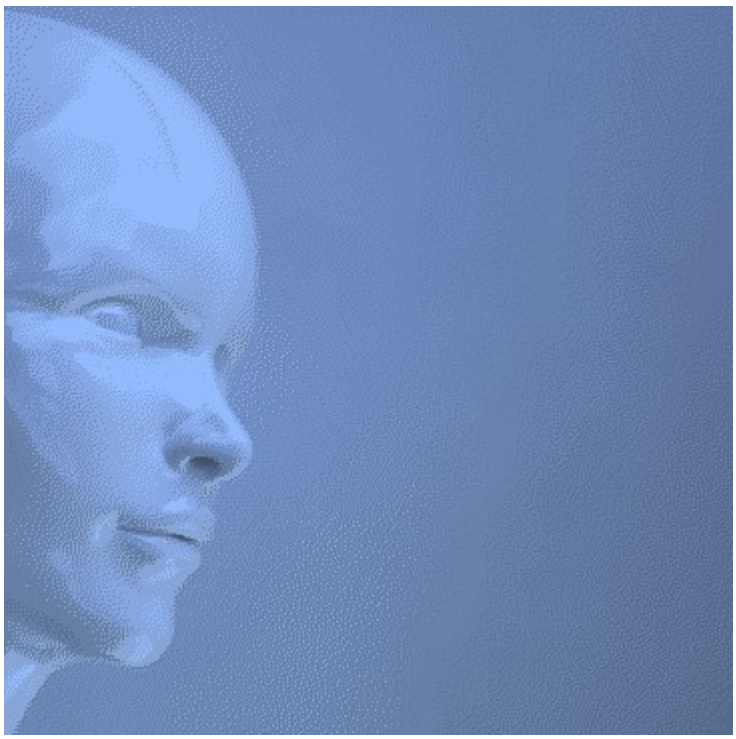
¿Pueden pensar los ordenadores?

¡Descubrir por ti mismo!



Esta teoría no ha sido actualizado para 2010.

¿Quieres ayudar a traducir?

Póngase en contacto conmigo a través de http://mafait.org.

Índice

1. Introducción

- ¿Puede se hacer más con programas informáticos que solamente hacer cuentas especiales, almacenar archivos y automatizar los procesos de negocio?
- ¿Podremos hacer preguntas á búsquedas en vez de buscar palabras y combinaciones de palabras en páginas web?
- ¿Conseguirán comprender lenguaje natural una vez?
- Ordenadores pueden digerir datos. ¿Pero podrán los ordenadores digerir información en algún tiempo? En otras palabras: ¿Podremos automatizar la información?

Muchos han echo intentos a causa de hacer ‘pensar’ los ordenadores. Pero si no hay ningún concepto por detrás d’estos intentos, son intentos ilusorios.

1.1. El Thinknowlogy concepto

El Thinknowlogy concepto tiene los siguientes principios:

1) Lenguaje de programación es basado en álgebra y lógica.
2) Toda persona tiene una sensación innato para álgebra y lógica, mismo que esto es más culto en uno que en otro. Esto se nota en su lenguaje natural.
3) Por hacer una combinación entre lenguaje natural y álgebra con la lógica de lenguajes de programación, es posible programar en lenguaje natural.
4) El objetivo final es automatizar información.

Primero exploremos cómo se puede combinar el álgebra y la lógica con lenguaje natural para que podremos programar en lenguaje natural.

Después exploremos como automatizar la información.

2. El álgebra y la lógica en lenguaje natural

Por debajo unos aspectos del álgebra ya de la lógica en lenguaje natural.

2.1. Una definición-generalización/especificación

Cualquier persona organiza sus pensamientos por hacer engrupamientos y generalizar. El significado de generalizar es por supuesto: Distinguir la cuestión principal de las cosas secundarias. Llamemos las cuestiones principales de **generalizaciones** y las cosas secundarias de **especificaciones**.

El proceso; distinguir generalizaciones de especificaciones, se encuentra mismo en el nivel mas bajo del lenguaje. Como ejemplo la frase: *"Juan es un hombre."*.

"Juan" es el sujeto y *"hombre"* es una calidad de *"Juan"*. En otras palavras: *"Juan"* es la generalización de la especificación *"hombre"*, por que *"hombre"* quiere decir algo acerca del sujeto *"Juan"*.

Más especificaciones de Juan:

- *"Juan es un padre."*
- *"Juan es un panadero."*

En las frases anteriores se utiliza cada vez un artículo indefinido: *"Juan es un hombre."*, *"Juan es un padre."* y *"Juan es un panadero."*. Dentro del Thinknowlogy concepto, llamamos esta estructura una **definición-generalización/especificación**.

2.1.1. Una especificación-relación

Oje la frase: *"Juan es een vriend van Mark."*.

Generalisatie *"Jan"* heeft een *"vriend"*-relatie met *"Mark"*. Hierbij is *"vriend"* de specificatie en *"Mark"* de relatie. Daarom wordt dit een **relatie-specificatie** genoemd.

2.1.2. Colecciones

Hieronder wordt besproken hoe diverse woorden 'verzameld' worden.

2.1.2.1. Una colección-especificación

Dentro de una definición-generalización/especificación la generalización puede tener múltiplas especificaciones que pertenecen juntas, como en el siguiente ejemplo: *"Una copa es llena o vacía."*. O más especificado: *"Una copa es llena, medio llena o vacía."*.

Un grupo de especificaciones como el de encima llamamos de **colección de especificaciones** o **colección-especificación**.

2.1.2.2. Una colección-relación

Oje la frase: *"Juan es vriend van Mark en Paula."*.

In dit geval worden de relaties *"Mark"* en *"Paula"* verzameld, wat we dan ook een **relatie-verzameling** noemen.

2.1.2.3. Una colección-generalización

Ahora las siguientes frases:

- *"Bush es el ex presidente de los Estados Unidos."*
- *"Obama es el actual presidente de los Estados Unidos."*

En esta ocasión la especificación es igual, pero las generalizaciones son desiguales. Por caso que *"Bush"* y *"Obama"* pertenecen juntos en esta, las generalizaciones tienen que ser coleccionados.

Llamemos esto de una **colección-generalización**.

2.1.3. Asignaciones

Een **asignación** is een generalización/especificación met een toegewezen waarde. Een generalisatie/specificatie is in principe statisch, maar een toewijzing is dynamisch, omdat de toestand kan wijzigen.

Een toewijzing geeft dus de actuele toestand van een generalización/especificación aan en is te herkennen aan door het **bepaalde lidwoord** in die zin.

2.1.3.1. Una asignación-especificación

Ejemplo: *"La copa es medio llena."*.

Esto es una estructura-generalización/especificación, waarbij un artículo definido wordt gebruikt en de specificatie tot een colección-especificación behoort, bijvoorbeeld: *"Una copa es llena, medio llena o vacía."*. Llamamos esto de una **asignación-especificación**.

Een asignación geeft de actuele toestand aan en kan dus ook wijzigen, want als het glas leeggedronken is, geldt een nieuwe toestand: *"La copa es vacía."*.

Een voorbeeld van een asignación van een especificación-relación: *"Juan es el padre de Pedro."*.

2.1.3.2. Asignación-generalizaciones

Oje las siguientes frases:

- *"Bush es el ex presidente de los Estados Unidos."*
- *"Obama es el actual presidente de los Estados Unidos."*

Estas frases también son Asignaciones, por que también se usa un artículo definido. Aunque las generalizaciones ahora son desiguales y forman una colección-generalización. Por eso llamamos estas asignaciones de **asignaciones-generalizaciones**.

2.1.3.3. Asignaciones en verleden tijd

Als een asignación wijzigt, noemen we de oude toestand in de taalkunde: **verleden tijd**.

Voor het glas gold ooit: *"La copa es medio llena."*. Maar het glas is leeggedronken en nu geldt er:

- *"La copa was medio llena."*
- *"La copa is (nu) vacía."*

Ooit gold ook: *"Bush es el presidente de los Estados Unidos."*, maar de toestand is gewijzigd in:

- *"Bush es el ex presidente de los Estados Unidos."* o *"Bush was el presidente de los Estados Unidos."*
- *"Obama es el (actual) presidente de los Estados Unidos."*

2.2. Una selección

En lenguaje natural prevemos también selecciones, como en la frase: *"Si la luz del semáforo es amarilla o roja, tienes que parar."*. Entonces, la condición *"La luz del semáforo es amarilla o roja"* cuenta la acción "tienes que parar".

Hay alternativas con frecuencia. Si la luz del semáforo es verde, cuenta en esta: "Tienes que andar.". Entonces, fusionado: *"Si la luz del semáforo es amarilla o roja, tienes que parar, en otra ocasión tienes que andar."*.

Entonces una selección consiste en dos o tres partes:

- Una condición (*"Si la luz del semáforo es amarilla o roja"*),
- Una acción (*"Tienes que parar."*)
- Y por acaso una acción alternativa (*"Tienes que andar."*).

En esta cuentan estructura-generalización/especificación:

- La condición: *"Una luz del semáforo es amarilla o roja."*;
- Las acciones: *"Tienes que andar o parar."*.

Ambos condiciones y acciones son Asignaciones:

- Las asignaciones de una condición son usas para determinar si la condición es verdadera. La condición en el ejemplo es verdadera si cuenta una de las asignaciones: *"La luz del semáforo es amarilla."* o *"La luz del semáforo es roja."*;
- Dependiente mente del resultado se hace la asignación de la acción o de la alternativa acción, entonces: *"Tienes que parar."* o *"Tienes que andar."*.

Esta teoría no ha sido actualizado para 2010.

¿Quieres ayudar a traducir?

Póngase en contacto conmigo a través de http://mafait.org.

2.5. Encontrando vínculos causales

Cuando el systema hace falta de información, es posible programar el sistema para hacer preguntas a usuarios del sistema. Del mismo modo se puede demostrar información contradictoria y corregir las fallas por haciendo preguntas.

Hieronder wordt een methode uitgewerkt om het aanscherpen van informatie te stimuleren.

2.6. Haciendo preguntas independiente

Cuando el systema hace falta de información, es posible programar el sistema para hacer preguntas a usuarios del sistema. Del mismo modo se puede demostrar información contradictoria y corregir las fallas por haciendo preguntas. Rematemos más sobre este asunte en otra ora.

2.7. Semantic ambiguity

Two types of ambiguity can be distinguished: static ambiguity and dynamic (e.g. time-related) ambiguity.

An example of static ambiguity:
• "*Boston is a city in both the United States and the United Kingdom*".

An example of dynamic ambiguity:
• "*Bush is inaugurated as (the) president of the United States.*", because George H. W. Bush was inaugurated in 1989, and his son George W. Bush was inaugurated in 2001, and re-inaugurated in 2005.

2.7.1. Autonomous semantic disambiguation

When a sentence is entered and its context (semantics) is not clear, the system can either:
• use deduction to determine which context is meant by the user;
• make an assumption, when the meant context cannot be determined, but when it is quite obvious;
• or ask a question, when the system has no clue about the context.

3. Creando una estructura de conocimiento

Probablemente cualquiera persona con experiencia en programación ha descubrido ya algunos principios básicos de lenguajes de programación: Una assignment en la forma de una asignación, una if-then-else estructura en la forma de selección y quizá también una declaración de un variante en la forma de definición-generalización/especificación: Mismo que ya se puede dar la estructura a l'asignación, no se puede asignar valor a l'asignación.

Con estas similitudes entre lenguaje natural y lenguaje de programación se puede arreglar una estructura de conocimiento. Por esto se orige un vínculo entre el ya mencionado lenguaje natural y el lenguaje de programación aunque se hace posible en principio que se puede programar en lenguaje natural. Esto puede ser la base para hacer un ordenador que puede 'pensar'.

Ahora sigue la explicación individual de como se puede hacer una estructura de conocimiento. Hacemos esto con frases en lenguaje natural.

3.1. Creando una especificación-generalización

Un elemento básico de la estructura de conocimiento es la generalización/especificación. Diversos aspectos de esta estructura son describidnos por debajo. También es describid cómo se hace la estructura.

3.1.1. La creación de palavras nuevas

Oje la frase: *"Juan es un padre."*.

Antes de hacer la estructura de conocimiento, el sistema tiene que crear primero las palabras *"Juan"* y *"padre"*. Entonces, si estas palabras ya no existen en el sistema, o no son del tipo justo de gramática, tienen que ser creadas primero.

Juan
La palabra *"Juan"* empieza con una letra mayúscula y es la primera palabra de la frase. Entonces puede ser del tipo gramatical nombre propio o de un tipo de gramática desconocido, si la palabra originaria no empieza con una letra mayúscula.

En este caso son creado dos palabras:

- *"Juan"* del tipo gramatical nombre *propio*;
- *"Juan"* (sin letra mayúscula) del tipo gramatical desconocido.

Durante se carpintea la estructura de conocimiento se hace uso de una de ambas palabras. La palabra desconocida es eliminada para que no se contamine el sistema. Sobre este asunto proseguiremos posteriormente.

padre
Posterior a la palabra *"padre"* se hace uso de un artículo, entonces por aquí el tipo gramatical es obvio: es un substantivo.

En la frase *"La copa es llena, media llena o vacía."*, no está claro el tipo gramatical da las palabras *"llena"*, *"media llena"* y *"vacía"*. Aunque esperamos que son del mismo tipo de gramática.

Esta teoría no ha sido actualizado para 2010.

¿Quieres ayudar a traducir?

Póngase en contacto conmigo a través de http://mafait.org.