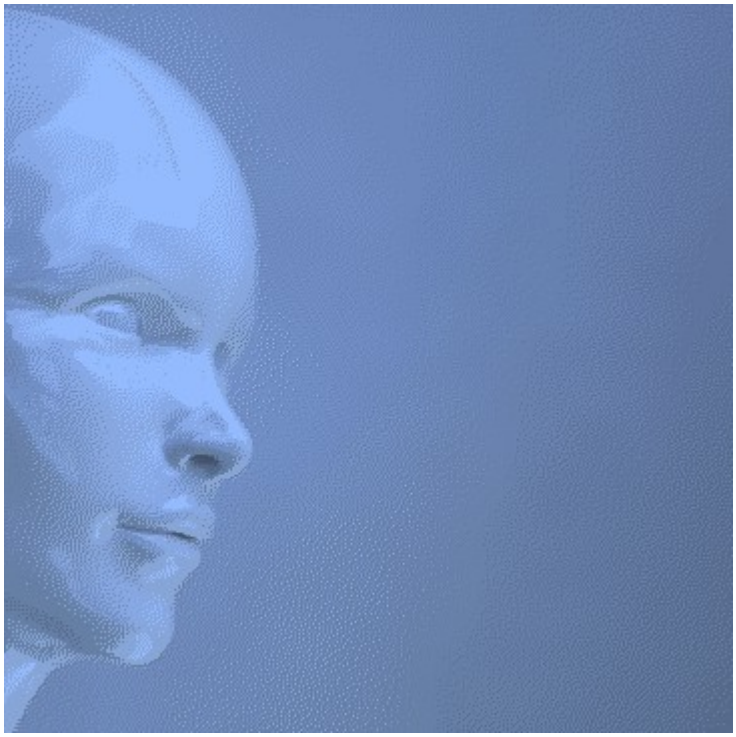
Computadores podem pensar?

Descubra você mesmo



Índice

1. Introdução

- É possível fazer mais com um computador do que somente algumas contas complicadas, armazenamento de palavras e a automação de processos em empresas?

- Será que algum dia poderemos fazer perguntas para as máquinas de busca, ao invés de procurar por palavras ou combinações delas, pelas páginas da web?

- Será que algum dia computadores compreenderão a língua humana?

- Computadores conseguem processar dados. Mas será que algum dia eles conseguirão processar informações? Melhor dizendo: será que algum dia conseguiremos automatizar a informação?

Já foram feitas muitas tentativas para conseguir fazer um computador 'pensar', mas se não tiver um conceito por detrás desta tentativa, estas tentativas são inúteis. Somente com um conceito ponderado existe uma chance de sucesso.

1.1. O conceito Thinknowlogy

O conceito Thinknowlogy conhece os seguintes princípios:

1. Linguagens de programação são baseadas em álgebra e lógica.
2. Todo humano desde o nascimento tem certa capacidade de raciocínio lógico e de álgebra, mesmo que em algumas pessoas esta capacidade seja mais desenvolvida do que em outras. Isto leva, entre outras coisas, à manifestação em linguagem natural.
3. Combinando álgebra e lógica em linguagem natural com a álgebra e lógica da linguagem de programação, é possível programar em linguagem natural.
4. O objetivo final é conseguir automatizar informação.

Primeiro pesquisaremos como álgebra e lógica em linguagem natural podem ser combinadas com a álgebra e lógica das linguagens de programação, para que possamos programar em linguagem natural.

Depois pesquisaremos como podemos automatizar informações.

2. A álgebra e lógica em linguagem natural

Abaixo seguem os vários aspectos da álgebra e lógica na linguagem natural.

2.1. Uma definição generalizada/específica

Os humanos organizam seus pensamentos agrupando-os, colocando-os em quadradinhos, generalizando-os. Generalizar significa na verdade: Separar o principal do secundário. O principal chamamos de **generalização** e o secundário chamamos de **especificação**.

O processo de diferenciação de generalizações e especificações está oculto até mesmo nos níveis mais baixos da linguagem. Pegue como exemplo a frase: *"João é um homem."*.

"João" é o assunto e *"homem"* é a característica de *"João"*. Melhor dizendo: *"João"* é a generalização da especificação *"homem"*, porque *"homem"* conta algo específico sobre o assunto *"João"*.

Mais especificações de João:

- *"João é um pai."*
- *"João é um padeiro."*

Nas frases anteriores usamos o tempo todo um artigo indefinido: *"João é um homem."*, *"João é um pai."* e *"João é um padeiro."*. Esta estrutura chamamos de o conceito Thinknowlogy, uma **generalização/especificação** ou uma **definição-generalizada/específica**.

2.1.1. Uma especificação relacional

Olhe a frase: *"João é um amigo do Marcos."*.

Generalização *"João"* tem um *"amigo"* relação com *"Marcos"*. *"Amigo"* aqui é chamado de **especificação** e *"Marcos"* é chamado de **relação**. Por isto esta frase são chamadas de **especificação relacional**.

2.1.2. Agrupamentos

A seguir explicaremos como diversas palavras são ‘agrupadas‘.

2.1.2.1. Um agrupamento de especificações

Em uma definição generalizada/específica, a generalização pode ter várias especificações que estão conectadas, como em: *“Um copo está cheio ou vazio.”*. Ou mais específico: *“Um copo está cheio, meio cheio ou vazio.”*.

Um grupo de especificações deste tipo chamamos de **agrupamento de especificações**.

2.1.2.2. Um agrupamento de relações

Olhe a frase: *“João é um amigo do Marcos e da Paula.”*.

Neste caso as relações “*Marcos*“ e “*Paula*“ são agrupadas, o que também chamamos de uma **agrupamento de relações**.

2.1.2.3. Um agrupamento de generalizações

Agora olhe as seguintes frases:

- *“Bush foi o último presidente dos Estados Unidos.”*
- *“Obama é o atual presidente dos Estados Unidos.”*

Nestes exemplos as especificações e relações são iguais, porém as generalizações são diferentes. Já que *“Bush”* e *“Obama”* estão conectados, temos que agrupar estas generalizações.

Chamamos este agrupamento de **agrupamento de generalizações**.

2.1.3. Atribuições

Uma **atribuição** é uma definição generalizada/específica com um valor atribuído. Uma generalização/especificação é em princípio estática, porém uma atribuição é dinâmica, já que a situação pode mudar.

Logo, uma atribuição nos dá uma situação atual de uma definição generalizada/específica e é reconhecida por um **artigo definido** na frase.

2.1.3.1. Uma atribuição de especificação

Exemplo: *"O copo está meio cheio."*.

Esta é uma definição generalizada/específica, onde um artigo definido é usado e onde a especificação pertence a um agrupamento de especificações, por exemplo: *"Um copo está cheio, meio cheio ou vazio."*. Chamamos isto de uma **atribuição de especificação**.

Por a atribuição mostrar a situação atual, esta situação pode mudar, porque então alguém bebe do copo, logo a situação muda: *"O copo está vazio."*.

Um exemplo de uma atribuição de uma especificação relacional: *"João é o pai de Pedro."*.

2.1.3.2. Uma atribuição de generalização

Tenha como exemplo as frases:

- *"Bush foi o último presidente dos Estados Unidos."*
- *"Obama é o atual presidente dos Estados Unidos."*

Estas também são Atribuições, porque também usam um artigo definido. Porém agora as generalizações são diferentes e formam um agrupamento de generalizações. Por isto estas atribuições são chamadas de **atribuições de generalização**.

2.1.3.3. Atribuições e o pretérito

Se uma atribuição muda, na linguística chamamos a antiga situação de: **pretérito**.

Para o copo, algum dia foi válido: *"O copo está meio cheio."*. Mas a água foi bebida e agora é válido:

- *"O copo estava meio cheio."*
- *" O copo (agora) está vazio."*

Algum dia também foi válido: *"Bush é o presidente dos Estados Unidos."*, mas a situação mudou para:

- *"Bush foi o último presidente dos Estados Unidos." ou "Bush era o presidente dos Estados Unidos."*
- *"Obama é o (atual) presidente dos Estados Unidos."*

2.2. Seleções

Linguagem natural provém também de seleções, como por exemplo na frase: *"Se o sinal de trânsito estiver amarelo ou vermelho, você deve parar."*. Então dentro da condição *"O sinal de trânsito está amarelo ou vermelho"* vale a ação *"você deve parar"*.

Muitas vezes temos alternativas. Senão, se o sinal estiver verde, vale: *"Você deve continuar."*. Então, juntando fica: *"Se o sinal de trânsito estiver amarelo ou vermelho, você deve parar, senão você deve continuar."*.

Uma seleção então é feita de três partes:

- Uma condição (*"Se o sinal de trânsito estiver amarelo ou vermelho"*),
- uma ação (*"você deve parar."*)
- e eventualmente uma ação alternativa (*"você deve continuar."*).

Aqui valem as estruturas de generalização/especificação:

- A condição: *"Um sinal de trânsito está amarelo ou vermelho."*;
- A ação: *"Você deve parar ou continuar."*.

Tanto condições e ações são Atribuições:

- A atribuição de uma condição é usada para determinar se a dada condição é verdade ou não. A condição do exemplo é verdadeira se uma das duas as atribuições for verdadeira: *"O sinal de trânsito está amarelo."* ou *"O sinal de trânsito está vermelho."*;
- Dependendo deste resultado será executada a atribuição da ação ou da ação alternativa, logo: *"Você deve parar."* ou *"Você deve continuar."*.

2.3. O tirar conclusões de forma autônoma

Não estamos acustumados à ideia de um software conseguir tirar conclusões próprias. Abaixo explicaremos como isto é realizável.

2.3.1. Conclusões especificação-substituição

Observe as frases:

- *"João é um pai."*
- *"Um pai é um homem."*

Primeiro uma observação: A última frase nós chamamos de uma **frase-definição**, porque a generalização é formada por um artigo indefinido e um substantivo e também a especificação é formada por um ou mais artigos indefinidos e substantivos.

Dessas frases tiramos a conclusão de que: *"João é um homem."*.

Esta conclusão pode ser automatizada, aplicando as seguintes regras:

1) Pegue uma frase onde a especificação é formada por um artigo indefinido e um substantivo;
2) Procure com esta especificação pôr frases-definição onde a especificação citada acima é igual à generalização da frase-definição;
3) Para a primeira frase citada valem também as especificações de cada uma das definições.

Então, no caso da frase acima:

1) Pegue a frase sobre *"João"*. A especificação (no singular) desta frase é: *"um pai"*;
2) Para esta especificação vale a definição: *"Um pai é um homem."*;
3) O software agora pode tirar a conclusão: *"João é um homem."* de forma autônoma, substituindo a especificação da frase sobre *"João"* pela especificação da frase-definição.

Nós chamamos isto de uma **conclusão especificação-substituição**, porque a especificação de uma frase-substantivo pode ser substituída por uma especificação de uma frase-definição.

2.3.1.1. Conclusões especificação-substituição compiladas

Observe as frases:

- *"Um cônjuge é um marido ou uma esposa."*
- *"Um marido é um homem."*
- *"Uma esposa é uma mulher."*

De acordo com a conclusão especificação-substituição (no singular) podemos substituir uma única especificação por uma especificação de uma frase-definição. Mas também em uma frase com uma especificação compilada, como em *"Um cônjuge é um marido ou uma esposa."*, podemos substituir as especificações se para ambas existir uma definição.

Substituindo respectivamente ambas as especificações, *"marido"* e *"esposa"*, por *"homem"* e *"mulher"*, chegamos a conclusão: *"Um cônjuge é um homem ou uma mulher."*.

Uma conclusão destas chamamos de uma **conclusão especificação-substituição compilada**, porque uma especificação compilada é substituída por uma outra definição.

Uma outra forma de uma conclusão especificação-substituição compilada surge quando substituimos uma especificação única por uma especificação compilada de uma frase-definição, como nas frases:

- *"Pedro é uma criança (de João)."*
- *"Um criança é um filho ou uma filha."*

Aqui a conclusão seria: *"Pedro é um filho ou uma filha (de João)."*.

2.3.2. Conclusão-reflectiva possessiva

Observe a frase: *"João é o pai de Pedro."*.

Deste frase podemos tirar a conclusão: *"Pedro tem um pai (chamado João)."*.

Primeiramente: O verbo *"ter"* chamamos de **verbo possessivo,** pois ele demonstra que a generalização possui algo.

As regras para uma **conclusão-reflectiva possessiva** são:

1) Na frase, tanto a generalização quanto a relação, precisam ser um nome próprio. Na hora de tirar a conclusão os dois são invertidos;
2) A frase precisa ter uma conjugação do verbo *"ser"*, que na conclusão precisa ser substituída por uma conjugação do verbo possessivo *"ter"*;
3) A frase precisa ter uma especificação (no singular) que contenha um substantivo. Se na especificação for usado um artigo definido, na conclusão este será substituido por um artigo indefinido.

Agora, aplicado na frase acima:

1) A generalização *"João"* e a sua relação *"Pedro"* são nomes próprios e então serão invertidos;
2) O verbo *"é"* é substituido pelo verbo possessivo *"tem"*;
3) A especificação (no singular) *"o pai"* possui um substantivo. O artigo definido *"o"* é substituído pelo artigo indefinido *"um"*.

O resultado então seria: *"Pedro tem um pai (chamado João)."*.

2.4. Fazendo suposições

Além de tirar conclusões, também podemos fazer suposições com base nas informações disponíveis. Mesmo que fazer suposições ainda não esteja integrado no sistema, a seguir segue uma explicação de como suposições poderiam ser automatizadas.

2.4.1. Suposições generalizadas

Observe primeiro esta frase: *"Um cônjuge é um marido ou uma esposa."*.

Por causa da palavra de conexão *"ou"* podemos dividir a primeira frase em duas definições simples:

- *"Um marido é um cônjuge."*
- *"Uma esposa é um cônjuge."*

Observe agora as frases:

- *"Um marido é um cônjuge."*
- *"João é um marido."* ou *"João é o marido de Paula."*

Agora vemos que a especificação da frase sobre *"João"* é igual à generalização da frase-definição. Logo, aqui podemos tirar uma conclusão especificação-substituição.

Então, das frases *"Um cônjuge é um marido ou uma esposa."* e *"João é o marido de Paula."* podemos tirar a 'conclusão': *"João é um cônjuge (de Paula)."*.

Esta 'conclusão' chamamos de uma **suposição generalizada**, porque a especificação-substantivo de uma frase pode ser substituída por uma generalização de um agrupamento de especificações exclusivas de uma frase-definição.

Uma generalização tem muitas exceções, por isto é uma suposição e não uma conclusão.

2.4.2. Suposições de especificações condicionais possessivas

Observe as frases:

- *"Uma aula tem professores e alunos."*
- *"João é (um) professor de Paulo."*

Duas observações sobre a primeira frase:

1) Esta frase tem uma conjugação do verbo possessivo *"ter"*;
2) A conjunção *"e"* demonstra que **ambas** as especificações são necessárias para fazer a definição ser válida (condição).

E duas observações sobre a segunda frase:

1) Podemos ler esta especificação relacional como: *"João"* tem uma relação *"professor"* com *"Paulo"*;
2) Mas a relação no sentido contrário, de *"Paulo"* para *"João"*, é desconhecida, e por isto não podemos tirar conclusões sobre *"Paulo"*, mesmo que nosso desejo seja de afirmar que:
    - *"Paulo é um aluno de João."*
    - *"João tem um aluno (chamado Paulo)."*

Mesmo assim queremos fazer algo com esta informação, pois ela pode nos dar informações importantes se as suposições acima sobre *"Paulo"* estiverem corretas. E se descobrirmos que estas suposições são falsas, pelo menos teremos mais clareza sobre a situação e poderemos aumentar a precisão da informação, algo para que nos empenhamos.

Podemos fazer **suposições de especificações condicionais possessivas**:

1) Quando temos uma conjugação do verbo possessivo *"ter"*;
2) Quando temos um agrupamento de especificações **condicionais**;
3) Porque visto a partir de *"alunos"*, uma relação com *"professores"* é no momento a única relação possível no sentido contrário;
4) Se em um momento mais adiante outras relações se tornarem possíveis, as suposições feitas terão que ser testadas;
5) Outras conclusões que fizermos com base em suposições também devemos declarar como suposições até que a primeira suposição seja confirmada ou negada. Aí todas as suposições feitas com base na primeira suposição deverão ser reavalidas;
6) Se uma suposição é confirmada, ela ganha o status de afirmação ou conclusão;
7) Se com base em uma especificação-substituição compilada podemos tirar uma conclusão, não chamamos isto de uma suposição, mas de uma pergunta.

2.5. Fazendo perguntas independentes

Se estiver faltando alguma informação em um sistema, é possível fazer o sistema perguntar ao usuário automaticamente. Informação contraditória também pode ser exibida e por meio de perguntas ao usuário, corrigida.

Abaixo mostraremos um método para estimular a precisão das informações.

2.5.1. Perguntas para estimular a precisão das informações

Anteriormente vimos a conclusão: *"Pedro é um filho ou uma filha (de João)."*.

Mas uma conclusão assim é um pouco vaga, por causa da escolha não respondida, demonstrada a partir da conjunção *"ou"*. Para estimular a precisão desta informação podemos colocar uma conclusão destas também em forma de pergunta. Fazemos isto pela conjugação do verbo *"ser"* no início da frase e adicionando um ponto de interogação ao final da frase.

O resultado é: *"Pedro é um filho ou uma filha (de João)?"*.

2.6. Encontrar ligações causais

É possível procurar automaticamente por ligações causais em um texto, das quais o sistema pode tirar conclusões com base em informações já conhecidas. Falaremos mais sobre isto adiante.

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. Construindo uma estrutura de conhecimento

Alguém com experiência em programação com certeza já encontrou no último capítulo alguns dos princípios básicos das linguagens de programação: Uma assignment na forma de uma atribuição, uma estrutura if-then-else na forma de uma seleção e talvez também a declaração de uma variável na forma da definição de generalização/especificação: Você declara a estrutura da variável, porém ainda atribui a ela um valor.

Com estas similaridades entra a linguagem natural e a linguagem de programação podemos construir uma estrutura de conhecimento. Assim surge uma relação entre a linguagem natural e a linguagem de programação.Por conseqüência, em princípio, seria possível programar em linguagem natural, o que poderia virar a base do computador que 'pensa'.

Abaixo segue a explicação de como construir uma estrutura de conhecimento com base em frases em linguagem natural.

3.1. A criação de uma generalização/especificação

Um importante elemento básico em uma estrutura de conhecimento é a generalização/especificação. Abaixo segue a descrição de diversos aspectos desta estrutura e como construir com estes uma estrutura de conhecimento.

3.1.1. A criação de novas palavras

Olhe a frase: *"João é um pai."*.

O sistema, primeiro, precisa criar as palavras *"João"* e *"pai"* antes que a estrutura de conhecimento possa ser construída. Então, se palavras ainda não existem em um sistema, ou se elas não são do tipo gramatical correto, estas palavras devem ser criadas primeiro.

João
A palavra *"João"* começa com uma letra maiúscula e é a primeira palavra da frase. Ela pode ser então do tipo gramatical *nome próprio* ou pode ser de um tipo gramatical desconhecido, se a maiúscula estiver ali apenas justificada por ser a letra inicial da primeira palavra da frase.

Neste caso são criadas duas palavras:

- *"João"* do tipo gramatical *nome próprio*;
- *"joão"* (sem letra maiúscula) de um tipo gramatical desconhecido.

Durante a construção da estrutura de conhecimento somente uma das palavras é usada. Após isto a palavra não usada é excluída, para que o sistema não fique poluído. Adiante falaremos mais sobre isto.

pai
A palavra *"pai"* vem logo após um artigo, então aqui o tipo gramatical é evidente: um substantivo.

Na frase *"Um copo está cheio, meio cheio ou vazio."* não está evidente qual o tipo gramatical das palavras *"cheio"*, *"meio cheio"* e *"vazio"*. Porém podemos presumir que são todos do mesmo tipo gramatical.

3.1.2. A criação de uma conexão de especificação

Agora que o sistema conhece as palavras, podemos conectar estas palavras umas com as outras. Melhor dizendo: A palavra generalizada será conectada à palavra especificada.

Exemplo: *"João é um pai."* e *"João é um padeiro."*.

Na primeira frase é criada, a partir da palavra generalizada *"João"*, uma conexão de especificação para a palavra especificada *"pai"* e na segunda frase, de *"João"* para *"padeiro"*.

Na estrutura de conhecimento agora é conhecido que João é tanto um pai quanto um padeiro.

3.1.2.1. A criação de uma conexão de relação

Para adicionar a especificação relacional *"João é um amigo de Marcos."* à estrutura de conhecimento, a palavra generalização *"João"* é conectada por meio de uma conexão de especificação à especificação *"amigo"* e à relação *"Marcos"*.

É usada então uma conexão de especificação com uma segunda atribuição dentro dela. Esta conexão de especificação especial é chamada também de uma **conexão de especificação e relação** ou simplesmente **conexão de relação**.

Esta conexão de relação deve ser lida da seguinte forma: *"João"* tem uma relação *"amigo"* com *"Marcos"*. O mesmo vale para *"Bush"* e *"Obama"* que tem uma relação *"presidente"* com os *"Estados Unidos"*.

Na frase *"João é um amigo de Marcos e da Paula."* duas conexões de relação são criadas a partir de *"João"*: Ambas com a especificação *"amigo"*, mas uma com uma relação para *"Marcos"* e uma com *"Paula."*.

3.1.3. A criação de um agrupamento

Olhe a frase: *"Um copo está cheio, meio cheio ou vazio."*.

As **palavras-especificações** da generalização *"copo"* são 'agrupadas' conectando-se mutuamente com conexões de agrupamento:

- *"cheio"* está conectado por uma conexão de agrupamento crescente a *"meio cheio"*;
- *"meio cheio"* está conectado por uma conexão de agrupamento crescente a *"vazio"*;
- *"vazio"* está conectado por uma conexão de agrupamento decrescente a *"meio cheio"*;
- *"meio cheio"* está conectado por uma conexão de agrupamento decrescente a *"cheio"*.

O 'crescente' ou 'decrescente' da conexão de agrupamento demonstra ordem na lista.

O agrupamento de generalização e relação acontece da mesma maneira:

O agrupamento de palavras-generalização

- *"Bush foi o último presidente dos Estados Unidos."*
- *"Obama é o atual presidente dos Estados Unidos."*

O agrupamento de palavras-relação

- *"João é o pai de Pedro e Maria."*

3.1.4. A criação de uma atribuição

Uma atribuição é uma estrutura de generalização/especificação com um valor atribuído. Por isto primeiramente é criado uma estrutura de generalização/especificação, se este ainda não existir, e só depois uma estrutura de atribuição.

3.1.4.1. A criação de uma atribuição de especificação

Uma atribuição de especificação com um agrupamento de especificações só pode exibir uma das situações ao mesmo tempo quando a frase tem a conjução *"ou"*.

Olhe a frase *"Um copo está cheio, meio cheio ou vazio."* e a situação atual (atribuição): *"O como está meio cheio."*.

Se agora alguém beber o restante do líquido no copo, valerá a situação: *"O copo está vazio."*. Neste caso a conexão determinada de *"copo"* para *"meio cheio"* é desativada, fazendo com que esta atribuição seja tratada com pretérito. Depois disto criamos uma conexão determinada de *"copo"* para *"vazio"*, para determinar que a situação atual do copo é *"vazio"*.

Várias atribuições por generalização
Uma atribuição de especificação pode ter várias Atribuições. Assim, ambas as atribuições podem valer ao mesmo tempo: *"O copo está vazio."* e "*O copo está quebrado.*".

Uma atribuição com varias relações
Ao criar uma atribuição de especificação com mais de uma relação, é criada uma atribuição de especificação separada para cada relação.

Dois observações:

- Na frase *"João é o pai de Pedro Pedro e Maria."* a conjunção *"e"* demonstra que ambas as relações estão conectadas e logo valem ao mesmo tempo;
- Mesmo depois você falado: *"João é o padeiro de Paul."*, as Atribuições acima continuam ativas, já que *"pai"* e *"padeiro"* não formam um agrupamento.

3.1.4.2. A criação de atribuições de generalização

As atribuições de generalização também somente conseguem apresentar uma atribuição ao mesmo tempo, já que elas estão conectados por um agrupamento de generalizações:

Primeiro vale: *"Bush é o presidente dos Estados Unidos."* e depois da eleição vale *"Obama é o presidente dos Estados Unidos."*, aí a atribuição de *"Bush"* para *"presidente dos Estados Unidos"* é desativada, fazendo como que esta atribuição seja tratada como pretérito, e é criada uma atribuição de *"Obama"* para *"presidente dos Estados Unidos"*.

3.2. A criação de uma seleção

Assim como a generalização/especificação, a seleção também é um elemento básico importante da estrutura de conhecimento.

Para construir uma estrutura de conhecimento de uma seleção, primeiro criamos todas as estruturas de generalização/especificação por quais a seleção é formada. Em seguida guardamos a condição em uma lista de condições, a ação em uma lista de ações e a ação alternativa em uma lista de ações alternativas.

Veja a frase: *"Se o sinal de trânsito estiver amarelo ou vermelho, você deve parar, senão você deve continuar."*.

Primeiro são criadas as generalizações/especificações *"O sinal de trânsito está amarelo ou vermelho."*, *"Você deve parar."* e *"Você deve continuar."*. Depois *"O sinal de trânsito está amarelo ou vermelho."* é colocada na lista de condições com uma referência à ação e a ação alternativa. *"Você deve parar."* é colocado na lista de ações e *"Você deve continuar."* é colocada na lista de ações alternativas.

3.2.1. Executando seleções

Após o sistema ler e processar uma frase, ele analisa todas as condições da estrutura de conhecimento e efetua, caso necessário, uma ação ou ação alternativa pertencente a condição. Este processo é repetido até que não aconteça mais nenhuma alteração no sistema. Então o sistema estará "em descanso" e estará pronto para ler e processar a próxima frase.

Assim parece um pouco que o sistema 'pensa', mas na verdade a única coisa que acontece é que um programa escrito em linguagem natural é executado.

4. Diálogos

Anteriormente convertemos frases em linguagem natural para uma estrutura de conhecimento. Porém, para manter um diálogo com o usuário, o sistema deve ser capaz de responder em frases legíveis também. Para isto a estrutura de conhecimento deve ser reconvertida para linguagem natural.

4.1. A gramática

Tanto para a leitura, quanto para a resposta em uma determinada língua, é necessário que o sistema conheça a gramática desta língua. Então, para qualquer língua desejada, primeiramente sua gramática deve ser definida.

Para manter um sistema flexível, esta gramática guardada em um documento de texto. Na hora de iniciar o sistema, ele mesmo importa um ou mais destes documentos de gramática.

Também é possível manter diálogos em várias línguas e no final seria possível também fazer o sistema traduzir automaticamente, porém só falaremos sobre mais para frente.

4.1.1. A leitura de uma frase

A gramática é usada na hora da leitura de uma frase para:

- controlar se a frase satisfaz a gramática definida;
- para poder diferenciar as várias estruturas, como a generalizada/específica, a atribuição e a seleção;
- para pré-determinar algumas palavras (Falaremos mais sobre isto adiante).

4.1.1.1. Somente aceitação de estruturas de frases conhecidas

Anteriormente vimos que para transformar frases em uma estrutura de conhecimento, precisa-se de um conceito para cada estrutura de frase (generalizada/específica, atribuição e seleção) analisada. Também vimos que estas estruturas de frases são determinadas em documentos de gramática.

Agora raciocinando ao contrário: O sistema **somente** pode aceitar estruturas de frases que tenham sido determinadas nos arquivos de gramática, porque somente para estes ele possui o conceito certo para colocar as informações na estrutura de conhecimento. Isto é uma limitação do sistema, porque na verdade o que queremos é que qualquer estrutura de frase possa ser lida e a informação convertida para a estrutura de conhecimento.

O objetivo final deste projeto, por isto, é criar conceitos para o máximo possível de estruturas de frases, que retirem informação desta e convertam-na para a estrutura de conhecimento.

Então ainda vai gastar muito tempo e trabalho para conseguir chegar ao objetivo, mas aí será possível fazer informação ser processada automaticamente e então o computador que 'pensa' estará próximo.

4.1.2. Respondendo com uma frase

A gramática é usada pelo sistema para formar com as palavras uma frase gramaticamente correta, e só então poder ser apresentada ao usuário.

4.1.2.1. Fazendo perguntas automaticamente

O sistema pode encontrar inconsistências e contradições na sua estrutura de conhecimento, mas falaremos mais sobre isto adiante.

Além disto, o sistema também pode fazer perguntas ao usuário automaticamente sobre as inconsistências e contradições, para assim completar, corrigir ou clarear as informações armazenadas. Isto parece um comportamento inteligente e auxiliará na idéia de que computadores podem pensar.

5. Aplicações

Se a teoria anteriormente discutida fosse suficientemente aperfeiçoada, computadores poderiam processar e "entender" informação textual, como por exemplo textos da internet, documentos de empresas, livros de escola e projetos de desenvolvedores de software.

Abaixo seguem algumas futuras possibilidades para quais este sistema poderia ser usado se a gramática do sistema suportasse suficientes estruturas de frases.

5.1. Programar em linguagem natural

Por meio do conceito de seleção é possível fazer um sistema executar automaticamente tarefas simples dadas em linguagem natural. Assim você poderia programar em linguagem natural e também, por exemplo, deixar um computador processar as regras de um jogo e depois jogar este jogo contra ele mesmo sem antes precisar programá-lo.

Desenvolvimento de Software
Um exemplo mais sério seria importar um esboço técnico de um desenvolvimento de um software e processar este documento, não precisando assim fazer o processo de programar. No mínimo este sistema poderia ser um protótipo para o software real, mas em princípio poderíamos automatizar também todo o processo de desenvolvimento de software.

Então na verdade seria a automação da automação. Para se pensar: *Seria isto canibalismo ou desenvolvimento inevitável?*

5.2. Avaliar e corrigir de textos e especificações

O sistema pode encontrar inconsistências e contradições em textos, também aquelas inconsistências e contradições em quais um humano não pensaria com muita facilidade. O sistema pode citá-los e eventualmente também corrigi-los, fazendo perguntas específicas para o usuário. Com o passar do tempo, seria assim possível deixar o sistema avaliar e corrigir textos e especificações.

5.3. A criação automatizada de resumos

A criação de resumos nada mais é do que separar os assuntos principais dos secundários, e depois retirar os assuntos secundários até o seu texto ficar do tamanho desejado.

Criar resumos automáticos de documentos seria útil principalmente para, por exemplo, gerentes que participam de reuniões com ducumentos imensos. Eles poderiam de forma simples, criar um resumo de tamanho desejado, e assim, estar sempre bem informado dos conteúdo do documento.

5.4. Programa de tradução que compreenda o contexto

A maioria dos programas de tradução traduz um texto palavra por palavra e não compreendem expressões. Por palavras várias vezes terem mais de um significado e os programas de tradução não entenderem o contexto da palavra, traduções automáticas geralmente são de qualidade ruim.

Este sistema pode, em princípio, 'entender' o contexto, assim será possível elevar a tradução automatizada a um nível bem superior e eficaz.

5.5. Um sistema de busca que possa responder perguntas

No momento ainda pesquisamos pela internet usando palavras chaves inteligentes, com quais esperamos encontrar páginas de web relevantes e assim respostas para as nossas perguntas.

É relativamente fácil encontrar o significado de termos e de palavras difíceis, mas fica mais complicado quando se quer uma resposta para uma pergunta mais específica, da qual você não conhece bem os termos ou com a qual seja difícil encontrar o contexto correto.

E em qual língua você tenta encontrar a informação? Provavelmente você tentará usar palavras em Inglês. Mas como você faz se não souber a tradução correta das palavras ou se a informação que está procurando esteja disponível, por exemplo, somente em Chinês? Como você consegue uma resposta para sua pergunta neste caso, se você não domina a língua?

Somente se os sistemas de busca conseguirem 'entender' a informação dada e conseguirem vê-la no contexto correto, será possível fazer perguntas para o sistema de busca.

5.6. Professor digital ou auto-didata

Um professor determina no início do ano escolar, por meio de livros de estudos e a quantidade de horas de aula disponíveis, quais capítulos e assuntos serão tratados naquele ano. Em toda aula o professor dá um resumo sobre os assuntos tratados aos alunos. Além disto o professor também pode responder perguntas sobre a matéria, porque ele/ela conhece a matéria em todos os detalhes.

Se for possível criar resumos automáticos (dos assuntos tratados) e fazer sistemas de busca responderem a perguntas, teria que ser possível também, automatizar professores de um assunto teórico, ou pelo menos, poder oferecer um bom professor digital como uma ajuda para pessoas auto-didatas.