

**Conhecimento sub-simbólico: Redes Neuronais Artificiais**

Bruno Pereira, 69303

João Mano, 69854

Patrícia Rocha, 69636

**Resumo**

Este relatório tem como objetivo explicar e documentar todo o processo de desenvolvimento do terceiro trabalho prático da unidade curricular de Sistemas de Representação de Conhecimento e Raciocínio no âmbito de conhecimento sub-simbólico, nomeadamente, o uso de Redes Neuronais Artificiais, RNAs.

As decisões e soluções implementadas serão aqui explicitadas e detalhadas por forma a apresentar todo o trabalho envolvido neste desafio.

Para a realização desta tarefa foram aplicados todos os conhecimentos, teóricos e práticos, adquiridos durantes as aulas desta unidade curricular.

Tabela de Conteúdos

**Extensão à Programação em Lógica e Conhecimento Imperfeito**

Introdução**6**

Objetivos**7**

Preliminares**7**

Descrição do Trabalho e Análise de Resultados**8**

Base de Conhecimento**8**

Implementação dos Predicados**9**

Predicado demo**9**

Evolução do Predicado demo**10**

Predicado evolEvol**11**

Predicados de Contexto**12**

Conhecimento Imperfeito**12**

Conhecimento Impreciso**12**

Conhecimento Incerto**13**

Conhecimento Interdito**13**

Invariantes **14**

Análise de Resultados**15**

Conclusões e Sugestões**25**

Tabela de Figuras

**­­Figura 1 –** Questão sobre a cor do automóvel a0002**15**

**Figura 2 –** Questão sobre a cor do automóvel a0002**15**

**Figura 3 –** Questão sobre a cor do automóvel a0002**15**

**Figura 4 –** Questão sobre qual o proprietário antigo do automóvel a0002**16**

**Figura 5 –** Questão sobre qual o novo proprietário do automóvel a0004 **16**

**Figura 6 –** Tentativa de inserção do registo do novo proprietário **16**

**Figura 7 –** Tentativa de inserção de matrícula existente **17**

**Figura 8 –** Tentativa de inserção de automóvel com código existente**17**

**Figura 9 –** demoE com verdadeiro e falso**17**

**Figura 10 –** demoE com verdadeiro e desconhecido**18**

**Figura 11 –** demoE com falso e desconhecido**18**

**Figura 12 –** demoE com verdadeiro e verdadeiro**18**

**Figura 13 –** demoOu com verdadeiro e falso**18**

**Figura 14 –** demoOu com verdadeiro e desconhecido**19**

**Figura 15 –** demoOu com falso e desconhecido **19**

**Figura 16 –** demoOu com falso e falso **19**

**Figura 17 –** Inserção de conhecimento verdadeiro repetido**20**

**Figura 18 –** Inserção de conhecimento positivo com negativo na base**20**

**Figura 19 –** Inserção de conhecimento positivo com desconhecido incerto na base **20**

**Figura 20 –** Inserção de conhecimento positivo com desconhecido impreciso na base**20**

**Figura 21 –**

Inserção de conhecimento negativo com positivo na base**21**

**Figura 22 –** Inserção de conhecimento negativo com negativo na base**21**

**Figura 23 –** Inserção de conhecimento negativo com desconhecido incerto na base**21**

**Figura 24 –**

Inserção de conhecimento negativo com desconhecido impreciso na base**21**

**Figura 25 –** Inserção de conhecimento incerto com positivo na base **22**

**Figura 26 –** Inserção de conhecimento incerto com negativo na base **22**

**Figura 27 –** Inserção de conhecimento incerto com conhecimento incerto na base**22**

**Figura 28 –** Inserção de conhecimento incerto com impreciso na base **23**

**Figura 29 –** Inserção de conhecimento impreciso com positivo na base **23**

**Figura 30 –** Inserção de conhecimento impreciso com negativo na base **23**

**Figura 31 –** Inserção de conhecimento impreciso com incerto na base **23**

**Figura 32 –** Inserção de conhecimento impreciso com impreciso na base**24**

**Conhecimento Sub-simbólico: Redes Neuronais Artificiais**

O objetivo deste relatório prende-se na documentação e explicitação do processo de resolução do terceiro exercício prático da unidade curricular se Sistemas de Representação de Conhecimento e Raciocínio.

A realização deste exercício te como objetivo incitar ao uso de sistemas sub-simbólicos face à representação de conhecimento e desenvolvimento de mecanismos de raciocínio.

Ao longo do documento expõem-se as decisões e conceitos definidos relevantes para a resolução do problema.

1. **Introdução**

Redes Neuronais Artificiais são técnicas computacionais, inspiradas no sistema nervoso, que adquirem conhecimento através de processamento de casos base e aprendizagem sobre estes. Estas técnicas permitem solucionar problemas de inteligência artificial através da construção de um processo baseado em circuitos imitadores do processamento e comportamento do sistema neuronal humano, adquirindo conhecimento através de experiências e erros.

As redes neuronais correspondem a grafos, com algumas restrições, onde a informação é distribuída para os nodos pelos caminhos.

O nível de cansaço aquando o uso de um computador pode ser medido pela análise de parâmetros como o movimento do rato ou *clicks*, assim sendo, é possível identificar através de uma recolha de dados o nível de fadiga de um utilizador. Neste contexto, através do uso de uma rede neuronal, teoricamente, será possível treinar a rede (com um erro de previsão associado) para que esta indique qual o nível de cansaço de um indivíduo face a um conjunto de dados sobre a utilização do computador.

Assim sendo, neste trabalho, serão usadas redes neuronais por forma a definir qual o nível de fadiga de um indivíduo.

1. **Objetivos**

Como já foi mencionado, o objetivo deste trabalho é utilizar redes neuronais para calcular níveis de fadiga de um utilizador, exercitando o conhecimento sobre este tema já abordado na unidade curricular.

O trabalho será realizado na linguagem de programação R, onde será criada e treinada a rede neuronal para utilização da mesma.

É então requerido que, face a um conjunto de casos base, se treine uma rede para que os seus valores de decisão retornados sigam, o mais próximo possível, os valores disponibilizados. É ainda pedido que, também utilizando os dados disponibilizados, a escala de valores retornados apenas indique se existe ou não fadiga e, por último, criar uma escala de valores de fadiga que melhor se aproxime do pretendido.

1. **Preliminares**

Quebrando a dependência da representação de conhecimento através do uso de símbolos, foi criada a representação de conhecimento sub-simbólicos. Neste trabalho apenas se abordaram um “ramo” deste amplo tema, Redes neuronais artificiais (RNAs). Para permitir uma melhor compreensão do tema de seguida explicar-se-ão conceitos essenciais à compreensão do abordado ao longo do relatório.

Redes neuronais artificiais são estruturas de resolução de problemas que quebram a dependência da utilização de símbolos. Baseia-se na conexão entre unidades de processamento e a sua nomenclatura é herdada da biologia.

Desta forma uma rede neuronal é constituída por:

* **Neurónio:** unidades de processamento;
* **Dentrite:** associadas aos neurónios, recebem a informação que depois é processada;
* **Axónio:** também associados aos neurónios, são responsáveis pela passagem de informação.

Um neurónio pode possuir várias dentrites mas apenas um axónio. À passagem de informação dá-se o nome de transferência/sinapse, esta apenas ocorre caso o estado de excitação dos neurónios seja suficiente. Este estado é regulado pela informação que chega ao neurónio.

A rede neuronal recebe então n parâmetros de um caso como input e, faz esta informação percorrer a sua rede até que é retornado um ou mais valor/valores de output. A aprendizagem da rede é definida pela regra de transferência que a rede neuronal implementa, isto será outro parâmetro que decidirá o funcionamento da rede.

O cálculo do valor de ativação dos neurónios é influenciado pela informação que chega aos mesmos, pelos dados de *input* e pelo valor de ativação anterior (armazenado em memória).

Apesar do uso das redes neuronais ser bastante vantajoso é de notar que todos os valores obtidos são apenas aproximações e que existe uma dependência na existência de “pré-conhecimento”, ou seja, são necessários casos de treino com informação já real.

Fiquei aqui….

1. **Descrição do Trabalho e Análise de Resultados**

Numa primeira parte da descrição será apresentado o contexto escolhido e quais os predicados desenvolvidos ao longo deste trabalho, de seguida apresentam-se as representações de conhecimento imperfeito e respetivas justificações ou origens. Por último, são apresentados exemplos práticos sendo em primeiro lugar apresentado o conhecimento inserido para se poderem validar os exemplos de forma mais certa.

* 1. **Base de Conhecimento**

Uma vez que era pedido para incorporar um universo do comércio automóvel, foi desenvolvido este trabalho direcionado para um *stand* de automóveis usados, o “AutoUsados”, assim sendo foram construídas as seguintes extensões de predicados:

* Extensão do predicado automovel que representa a existência de um dado automóvel:

automovel : CodigoAutomovel, Construtor, Marca, Modelo, AnoFabrico, Combustivel, Potencia, Lugares -> {V,F,D}

* Extensão do predicado matricula que associa uma matrícula a um automóvel:

matricula : Matricula, Automovel, AnoDeRegisto -> {V,F,D}

* Extensão do predicado cor que associa uma cor a um automóvel:

cor: Cor, Automovel -> {V,F,D}

* Extensão do predicado conservacao que dita qual o estado de conservação de um automóvel:

conservacao: Estado, Automovel -> {V,F,D}

* Extensão do predicado preco que associa a um automóvel o seu preço:

preco: Preco, Automovel -> {V,F,D}

* Extensão do predicado registoCompra que regista a compra de um automóvel por parte do *stand*:

registoCompra: Preco, Automovel, ProprietárioAntigo -> {V,F,D}

* Extensão do predicado registoVenda que regista a venda de um automóvel do *stand*:

registoVenda: Preco, Automovel, ProprietárioNovo -> {V,F,D}

Para além destes predicados relativos ao tema do trabalho é necessária a definição dos seguintes:

* Predicado evolucao
* Predicado evolEvol
* Predicado insercao
* Predicado teste
* nao
* pertence
* solucoes
* comprimento
* demo
* demoE
* demoOu

Destes apenas demo é introduzido na extensão à programação em lógica, todos os outros já tinham sido abordados nas aulas desta unidade curricular no tema de programação em lógica por isso não serão abordados à exceção do predicado demo.

O predicado demo representa a extensão à programação em lógica, definido como um predicado em lógica permite inserir o valor de verdade desconhecido e abandonar os pressupostos PMF e PDF.

Para além deste predicado e a sua evolução, demoE e demoOu, foi implementada ainda a evolução do predicado evolução, evolEvol.

* 1. **Implementação dos Predicados**
     1. **Predicado demo**

De maneira a utilizar a extensão à programação em lógica é necessário construir o predicado demo que, sendo construído à custa da programação em lógica permite o uso da extensão à mesma.

*demo( Questao,verdadeiro ) :-*

*Questao.*

*demo( Questao, falso ) :-*

*-Questao.*

*demo( Questao,desconhecido ) :-*

*nao( Questao ),*

*nao( -Questao ).*

O predicado demo recebe uma questão e forma uma resposta, assim sendo a resposta será verdadeiro se a questão se verificar (1ª definição), a resposta só será falso se existir conhecimento que tal é falso (2ª definição – negação forte) e, caso não exista conhecimento sobre esta questão, ou seja, não exista conhecimento positivo nem negativo acerca da mesma (negação fraca ou por falha na prova – predicado não), o valor retornado será desconhecido.

* + 1. **Evolução do Predicado demo**

Ainda existe um problema em relação a este predicado. O predicado demo “simples” apenas aceita uma expressão e o valor que retorna é sobre esta…

Assim sendo é necessário evoluir este predicado para conseguir retornar um valor para um conjunto de questões. Foram então criados os predicados demoE e demoOu que, retornam o valor de verdade de, respetivamente, a conjunção dos valores de verdade das questões com o operador lógico ‘e’ e a conjunção dos valores de verdade das questões com o operador lógico ‘ou’.

O predicado demoE tem então a seguinte implementação:

*demoE([],verdadeiro).*

*demoE([E|RL],verdadeiro):-*

*demo(E,verdadeiro), demoE(RL,verdadeiro).*

*demoE([E|RL],falso):-*

*demo(E,falso), nao(demoE(RL,desconhecido)).*

*demoE([E|RL],falso):-*

*demo(E,verdadeiro), demoE(RL,falso).*

*demoE([E|RL],desconhecido):-*

*demo(E,desconhecido).*

*demoE([E|RL],desconhecido):-*

*nao(demo(E,desconhecido)), demoE(RL,desconhecido).*

Por sua vez o predicado demoOu segue a dada especificação:

*demoOu([],falso).*

*demoOu([E|RL],verdadeiro):-*

*demo(E,verdadeiro).*

*demoOu([E|RL],verdadeiro):-*

*nao(demo(E,verdadeiro)), demoOu(RL,verdadeiro).*

*demoOu([E|RL],falso):-*

*demo(E,falso), demoOu(RL,falso).*

*demoOu([E|RL],desconhecido):-*

*demo(E,desconhecido), nao(demoOu(RL,verdadeiro)).*

*demoOu([E|RL],desconhecido):-*

*nao(demo(E,desconhecido)), demoOu(RL,desconhecido).*

Os valores de verdade para ambos os casos foram decididos da seguinte forma:

|  |  |  |  |
| --- | --- | --- | --- |
| demoE | Verdadeiro | Falso | Desconhecido |
| Verdadeiro | Verdadeiro | Falso | Desconhecido |
| Falso | Falso | Falso | Desconhecido |
| Desconhecido | Desconhecido | Desconhecido | Desconhecido |

Tabela - Valores de verdade para o demoE

|  |  |  |  |
| --- | --- | --- | --- |
| demoOu | Verdadeiro | Falso | Desconhecido |
| Verdadeiro | Verdadeiro | Verdadeiro | Verdadeiro |
| Falso | Verdadeiro | Falso | Desconhecido |
| Desconhecido | Verdadeiro | Desconhecido | Desconhecido |

Tabela - Valores de verdade para o demoOu

Foi assim decidido porque em termos da operação ‘e’ mal exista um desconhecido não sabemos qual será o resultado logo o desconhecido deverá ser o elemento absorvente desta operação, no caso do ‘ou’ o verdadeiro é o elemento absorvente.

Para os restantes casos, no demoE, foram aplicadas as regras normais da lógica e, para o demoOu, decidiu-se que desconhecido e falso deveria dar desconhecido uma vez que o valor de verdade é, de facto, desconhecido.

* + 1. **Predicado evolEvol**

Para este trabalho prático foi ainda evoluído o predicado de evolução mas, apenas foi definido para casos particulares, nomeadamente para o predicado cor e alguns casos para o predicado matricula. Não foi definido para os restantes predicados pois achou-se que a definição para um predicado já demonstraria a capacidade de evoluir o predicado evolução.

Assim, foram então definidos todos os casos possíveis seguindo a tabela de verdade:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| evolEvol | Verdadeiro | Falso | D. (Incerto) | D. (Impreciso) |
| Verdadeiro | Não | Insere e retira o Falso | Insere e retira o Incerto | Insere e retira todos os Imprecisos |
| Falso | Insere e retira o Verdadeiro | Não | Insere e retira Incerto | Insere e retira o Impreciso particular |
| D. (Incerto) | Não | Não | Não | Não |
| D. (Impreciso) | Não | Não | Insere e retira Incerto | Insere |

Tabela - Tabela de ação do predicado evolucao

Note-se que não se suporta evolução de conhecimento interdito e que todas as inserções estão sujeitas aos invariantes, assumindo que respeitam os invariantes então o insere da tabela é verificado.

* + 1. **Predicados de Contexto**

Apesar da extensão à programação em lógica abandonar o PMF, para este universo considerou-se que este pressuposto deveria ser aplicado, ou seja, se não houver um registo de um carro então o carro não existe no universo do *stand*, assim como se não houver um registo de compra ou de venda não existe essa compra ou venda.

Têm-se então definições semelhantes para todos os predicados de contexto:

*-automovel( CA, C, MA, MO, AF, CB, P, L ) :-*

*nao( automovel( CA, C, MA, MO, AF, CB, P, L ) ),*

*nao( excecao( automovel( CA, C, MA, MO, AF, CB, P, L ) ) ).*

*-preco( P, A ) :-*

*nao( preco( P, A ) ),*

*nao( excecao( preco( P, A ) ) ).*

*-registoVenda( V, A, P ) :-*

*nao( registoVenda( V, A, P ) ),*

*nao( excecao( registoVenda( V, A, P ) ) ).*

* + 1. **Conhecimento Imperfeito**

Por forma a aplicar os três tipos de conhecimento imperfeito foram elaboradas histórias que originam este tipo de conhecimento. De seguida apresenta-se a história e como foi implementado este tipo de conhecimento.

* + - 1. **Conhecimento Impreciso**

Assumiu-se que dado ao desgaste de um carro ao longo do tempo a cor tinha sido degradada e por isso, apesar de se situar numa gama de tons de vermelho não se conseguia identificar ao certo qual a cor do carro.

Para implementar este conhecimento tem-se:

*cor('Carmesim', a0002).*

*cor('Coral Claro', a0002).*

*cor('Salmao', a0002).*

*cor('Vermelho Indiano', a0002).*

Assim sendo é esperado que para todas estas cores o valor de verdade seja verdadeiro e para todas as outras seja falso, ou seja a cor do carro poderá sempre ser uma destas cores – ver exemplo prático 4.2.5.I.

* + - 1. **Conhecimento Incerto**

Para a representação deste tipo de conhecimento admitiu-se que o *stand* teria adquirido um automóvel apreendido pela polícia e como tal o dono desse veículo era desconhecido. Para a representação deste caso tem-se:

*registoCompra(300,a0002,proprietariodesc).*

*excecao( registoCompra( V, C, P ) ):-*

*registoCompra( Vs, C, proprietariodesc ).*

Isto significa que o registoCompra deste carro em especial recebe como antigo proprietário uma *tag* (valor específico) que representa o não saber qual o proprietário, assim sendo o proprietariodesc é dado como uma exceção ao registoCompra.

Desta forma é esperado que quando se pergunta se o Amílcar ou a Joana é o proprietário antigo deste carro o valor recebido seja desconhecido, qualquer que seja o valor introduzido pois só dependerá do código do automóvel – ver exemplo prático 4.2.5.II.

* + - 1. **Conhecimento Interdito**

Por forma a representar conhecimento interdito foi criada a seguinte história: uma celebridade atualmente na falência precisa de comprar um carro com valor muito mais baixo do que normalmente compraria, uma vez que a mesma quer impedir que a sua situação seja conhecida pelo público em geral exige que o registo seja agora e sempre censurado.

Para a representação deste conhecimento foi definido:

*registoVenda(200,a0004,interdito).*

*excecao(registoVenda(V,A,P)):-*

*registoVenda(Vs,A,interdito).*

*nulo(interdito).*

*+registoVenda( V, A, P ) :: (solucoes( Ps,( registoVenda(V,a0004,Ps), nao( nulo(Ps) ) ), S ),*

*comprimento( S,N ), N == 0*

*).*

Com este tipo de implementação acontece a mesma coisa que no conhecimento incerto mas a *tag* é declarada como um valor nulo e criado um invariante de inserção de conhecimento que não permitirá inserir nunca o verdadeiro proprietário antigo (ou qualquer proprietário).

Assim sendo é esperado que qualquer que seja a pergunta sobre o proprietário deste registo de venda a resposta tenha valor desconhecido- ver exemplo prático 4.2.5.III.

* + 1. **Invariantes**

Os invariantes implementados procuram refletir os casos reais do universo, assim sendo, para além dos invariantes estruturais de não permissão de inserção de conhecimento repetido existem os seguintes invariantes:

1. Não permitir a associação de uma matrícula a mais que um carro:

*+matricula( M, A, AR ) :: (solucoes( ( As, ARs ),(matricula( M, As, ARs )),S ),*

*comprimento( S,N ),*

*N == 1*

*).*

1. Não permitir a associação de um códido de automóvel a mais que um automóvel:

*+automovel( CA, C, MA, MO, AF, CB, P, L ) :: (solucoes( ( Cs, MAs, MOs, AFs, CBs, Ps, Ls ),(automovel( CA, Cs, MAs, MOs, AFs, CBs, Ps, Ls )),S ),*

*comprimento( S,N ),*

*N == 1*

*).*

1. Não permitir que um código de automóvel seja associado a mais que um registo (de venda e de compra):

*+registoVenda( V, A, P ) :: (solucoes( ( Vs, A, Ps ), ( registoVenda( Vs, A, Ps ) ), S ),*

*comprimento( S,N ),*

*N==1*

*).*

*+registoCompra( V, A, P ) :: (solucoes( ( Vs, A, Ps ), ( registoCompra( Vs, A, Ps ) ), S ),*

*comprimento( S,N ),*

*N==1*

*).*

O bom funcionamento destes invariantes poderá ser verificado pelos exemplos práticos apresentados na próxima subsecção.

* + 1. **Análise de Resultados**

Nesta secção serão apresentados alguns exemplos práticos por forma a comprovar o bom funcionamento da base de conhecimento construída.

1. **Conhecimento Impreciso**

Para este exemplo como referido anteriormente é esperado que a resposta seja verdadeira quando se pergunta se a cor do automóvel a0002 é salmão ou carmesim e falso para cores que não estão na gama destes valores como por exemplo azul.

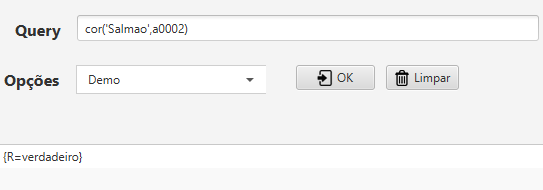


Figura - Questão sobre a cor do automóvel a0002

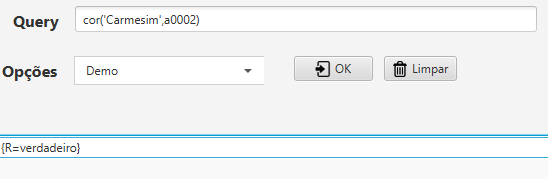


Figura - Questão sobre a cor do automóvel a0002

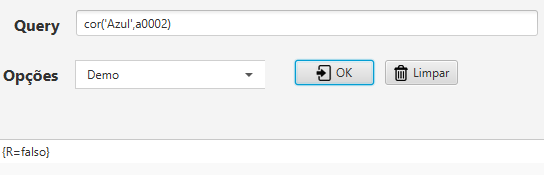


Figura - Questão sobre a cor do automóvel a0002

1. **Conhecimento Incerto**

Para este exemplo espera-se que qualquer pergunta sobre o proprietário do automóvel a0002 retorne o valor de verdade desconhecido.

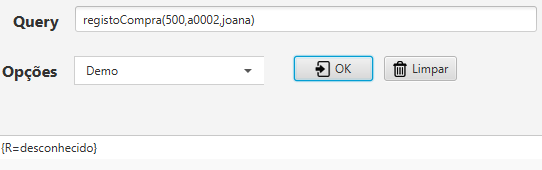


Figura - Questão sobre qual o proprietário antigo do automóvel a0002

1. **Conhecimento Interdito**

É esperado que para este exemplo prático qualquer que seja a questão sobre o novo proprietário do automóvel a0004 retorne o valor desconhecido e, aquando a tentativa de inserção desse conhecimento, dado que o mesmo é interdito, essa tentativa falhe.

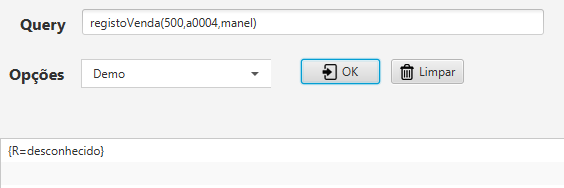


Figura - Questão sobre qual o novo proprietário do automóvel a0004

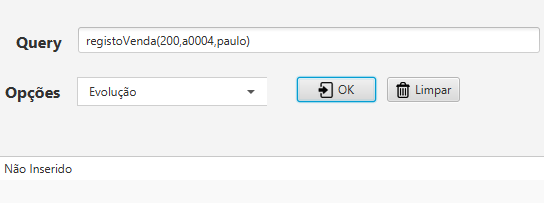


Figura - Tentativa de inserção do registo do novo proprietário

1. **Invariante: Não permitir mais que um automóvel com a mesma matrícula**

Neste teste, após a tentativa de inserção de um registo de associação de uma matrícula existente a um automóvel, espera-se que a sua inserção falhe.

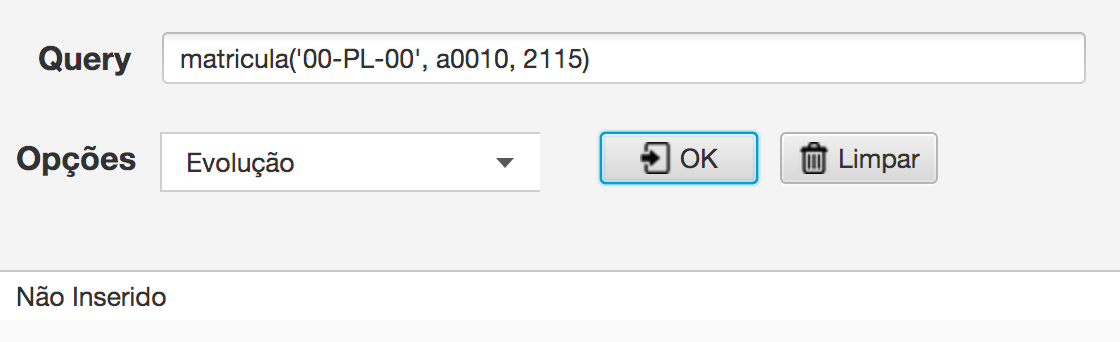
****

Figura - Tentativa de inserção de matrícula existente

1. **Invariante: Não permitir mais que um automóvel com o mesmo código de automóvel**

Assim como o anterior, esta tentativa de inserção de um automóvel com um código já existente deverá ser impossibilitada.

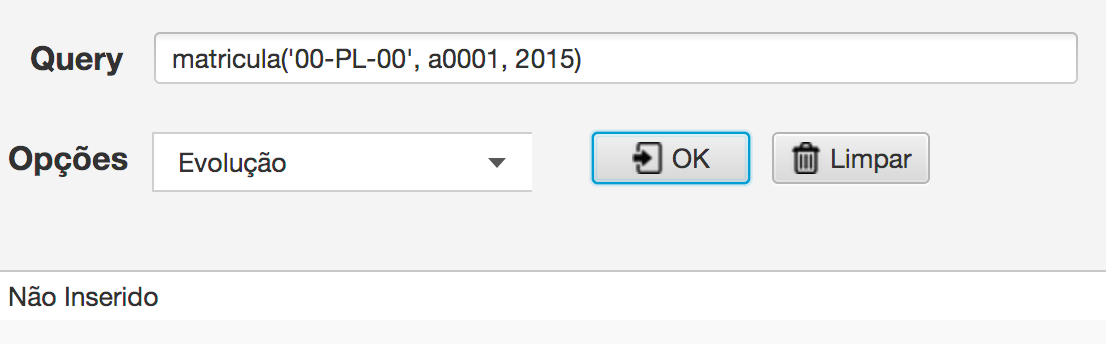
****

Figura - Tentativa de inserção de automóvel com código existente

1. **Evolução do predicado demo**
   * **demoE**
     + Questões com valor verdadeiro e falso – É esperado que o valor de retorno seja falso.

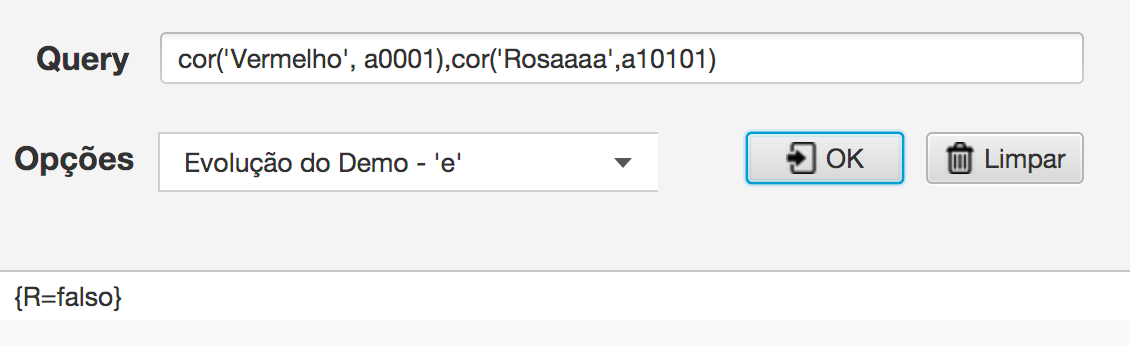


Figura - demoE com verdadeiro e falso

* + - Questões com valor verdadeiro e desconhecido – É esperado que o valor de retorno seja desconhecido.

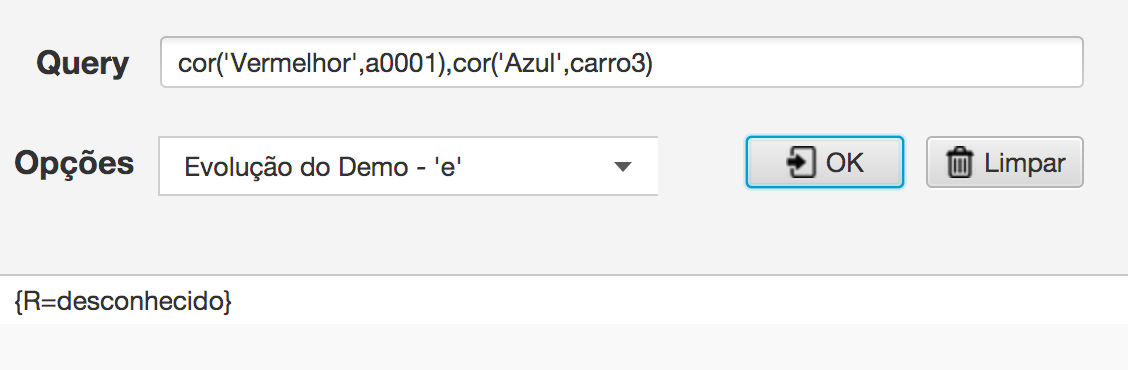


Figura - demoE com verdadeiro e desconhecido

* + - Questões com valor falso e desconhecido – É esperado que o valor de retorno seja desconhecido.

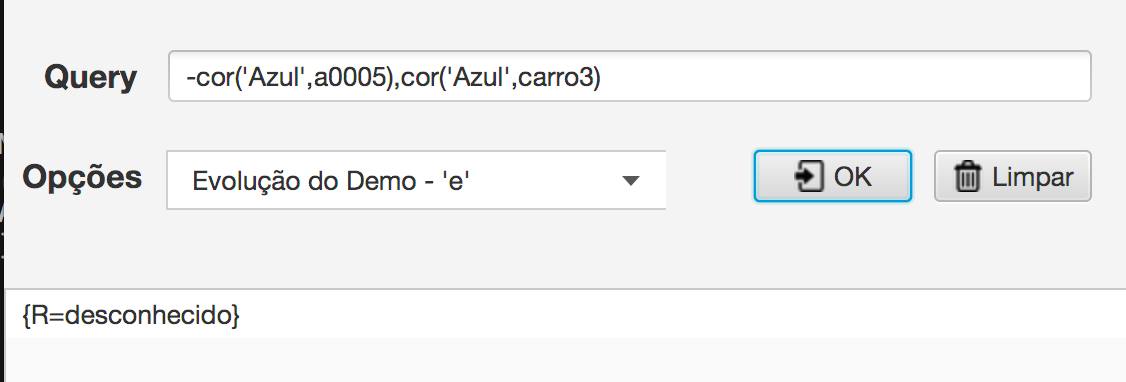


Figura - demoE com falso e desconhecido

* + - Questões com valor verdadeiro e verdadeiro – É esperado que o valor de retorno seja verdadeiro.

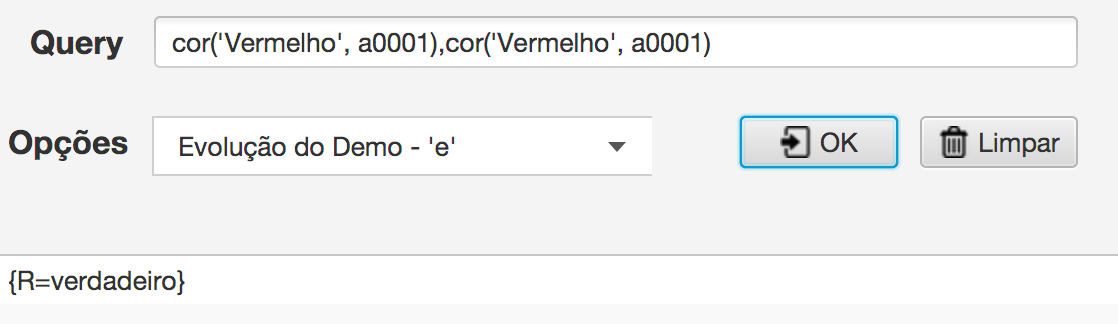


Figura - demoE com verdadeiro e verdadeiro

* + **demoOu**
    - Questões com valor verdadeiro e falso – É esperado que o valor de retorno seja verdadeiro.

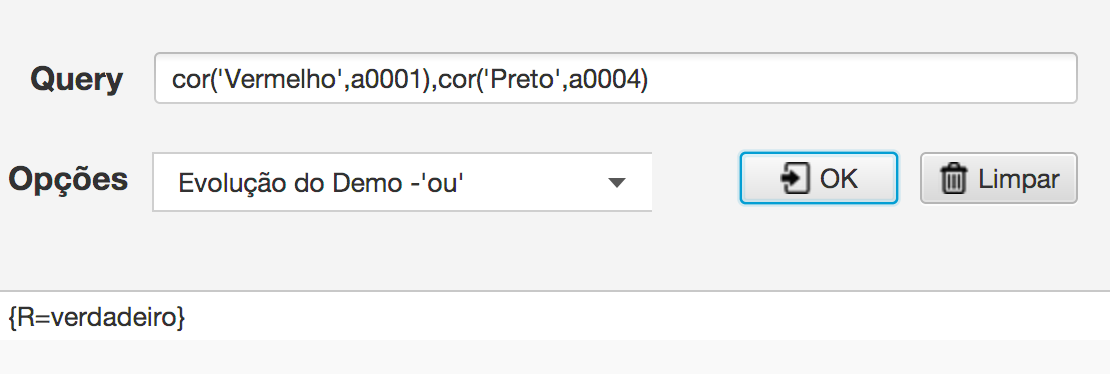


Figura - demoOu com verdadeiro e falso

* + - Questões com valor verdadeiro e desconhecido – É esperado que o valor de retorno seja verdadeiro.

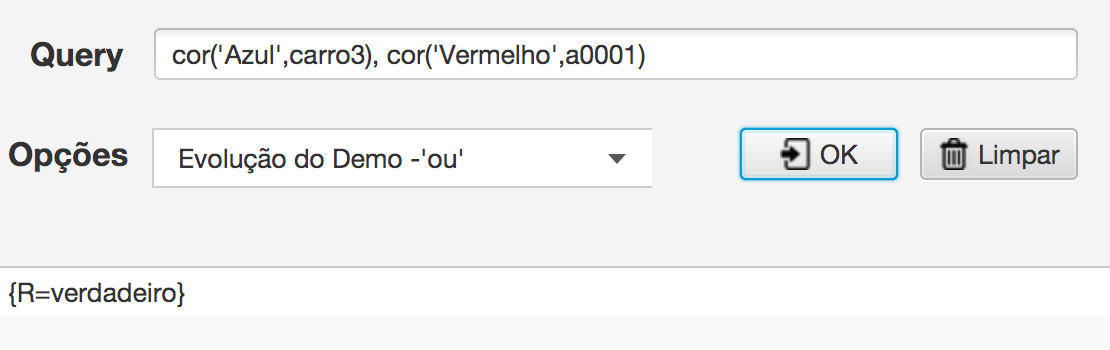


Figura - demoOu com verdadeiro e desconhecido

* + - Questões com valor falso e desconhecido – É esperado que o valor de retorno seja desconhecido.

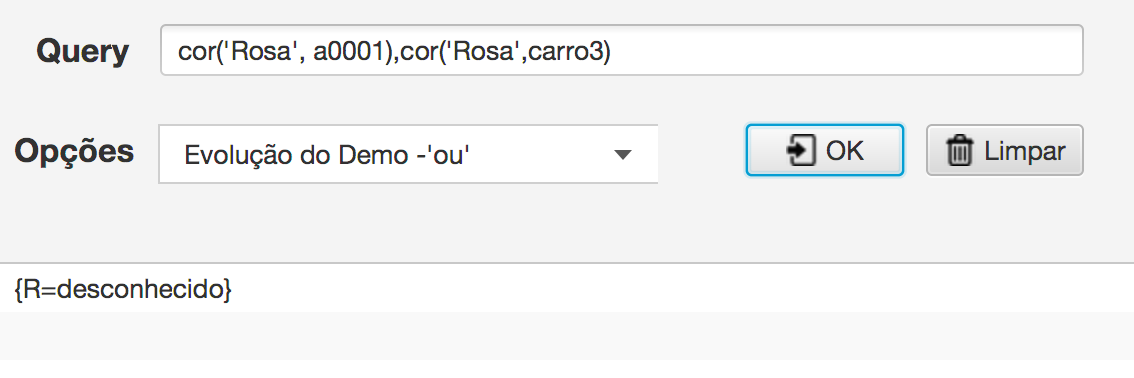


Figura – demoOu com falso e desconhecido

* + - Questões com valor falso e falso – É esperado que o valor de retorno seja falso.

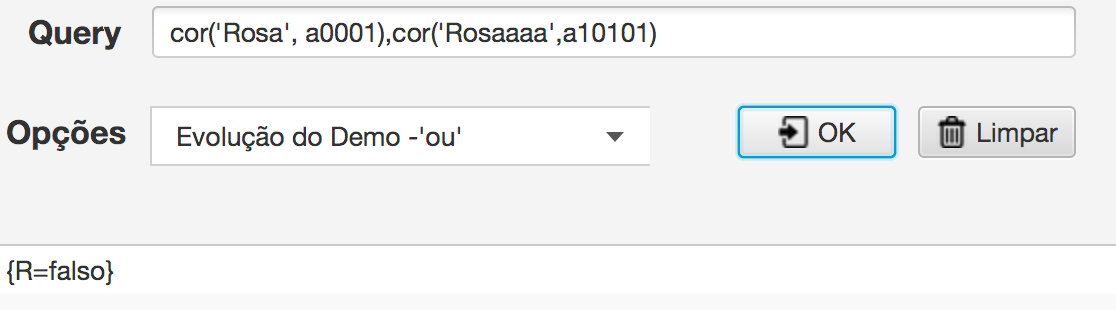


Figura - demoOu com falso e falso

1. **Evolução do predicado evolucao**
   * Introdução de conhecimento positivo e na base de conhecimento tem-se:
     + Conhecimento positivo (repetido) – É esperado que não seja inserido.

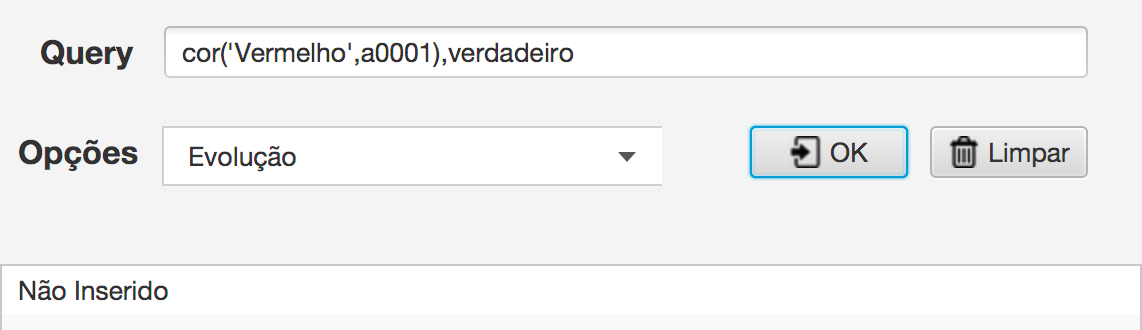
****

Figura - Inserção de conhecimento verdadeiro repetido

* + - Conhecimento negativo – É esperado que seja inserido e o negativo retirado.

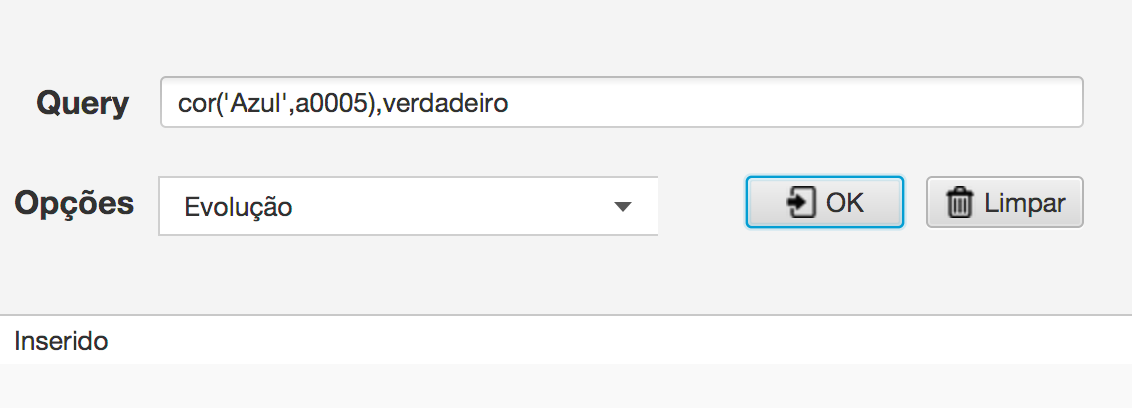
****

Figura - Inserção de conhecimento positivo com negativo na base

* + - Conhecimento desconhecido incerto – É esperado que seja inserido e o desconhecido sobre aquele seja removido.

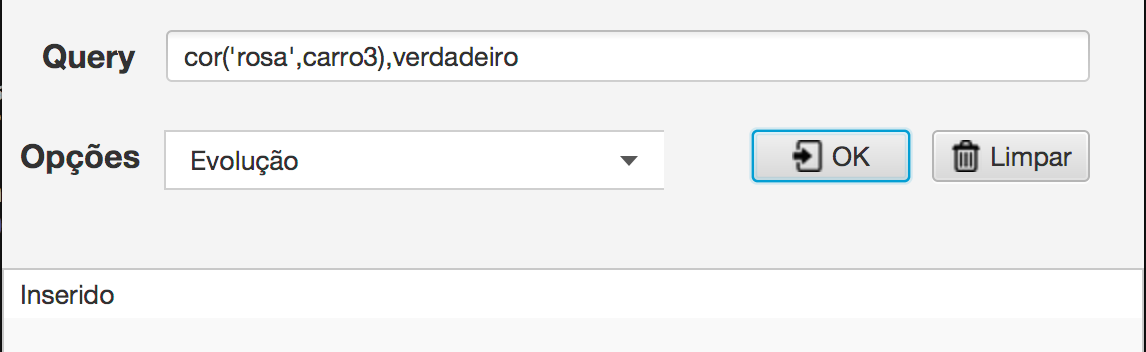
****

Figura - Inserção de conhecimento positivo com desconhecido incerto na base

* + - Conhecimento desconhecido impreciso – É esperado que o conhecimento seja atualizado para positivo.

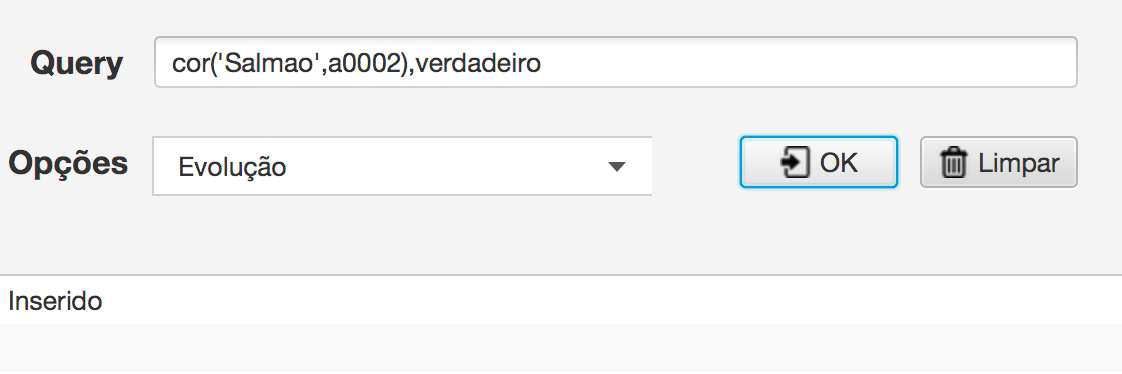
****

Figura - Inserção de conhecimento positivo com desconhecido impreciso na base

* + Introdução de conhecimento negativo e na base de conhecimento tem-se:
    - Conhecimento positivo – É esperado que o conhecimento seja atualizado.



Figura - Inserção de conhecimento negativo com positivo na base

* + - Conhecimento negativo (repetido) – É esperado que não seja inserido.

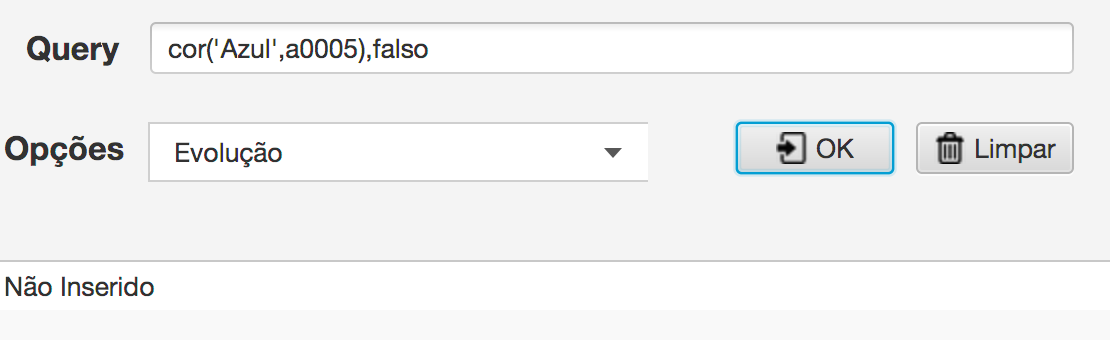
****

Figura - Inserção de conhecimento negativo com negativo na base

* + - Conhecimento desconhecido incerto – É esperado que se atualize a informação da base de conhecimento.

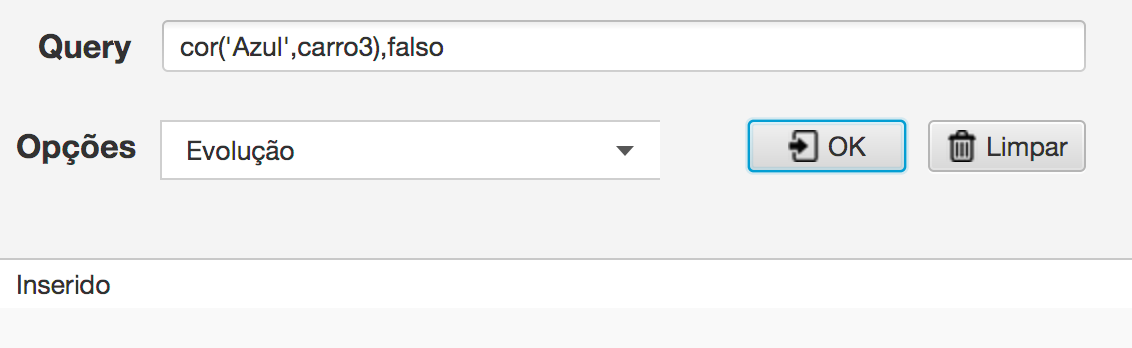
****

Figura - Inserção de conhecimento negativo com desconhecido incerto na base

* + - Conhecimento desconhecido impreciso – É esperado que se atualize o conhecimento.

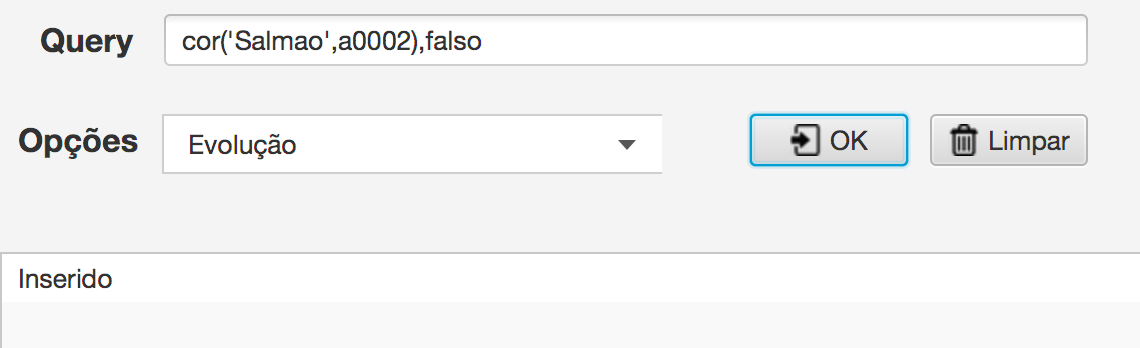


Figura - Inserção de conhecimento negativo com desconhecido impreciso na base

* + Introdução de conhecimento imperfeito do tipo incerto e na base de conhecimento tem-se:
    - Conhecimento positivo – É esperado que não seja inserido.

****

Figura - Inserção de conhecimento incerto com positivo na base

* + - Conhecimento negativo – É esperado que não seja inserido.

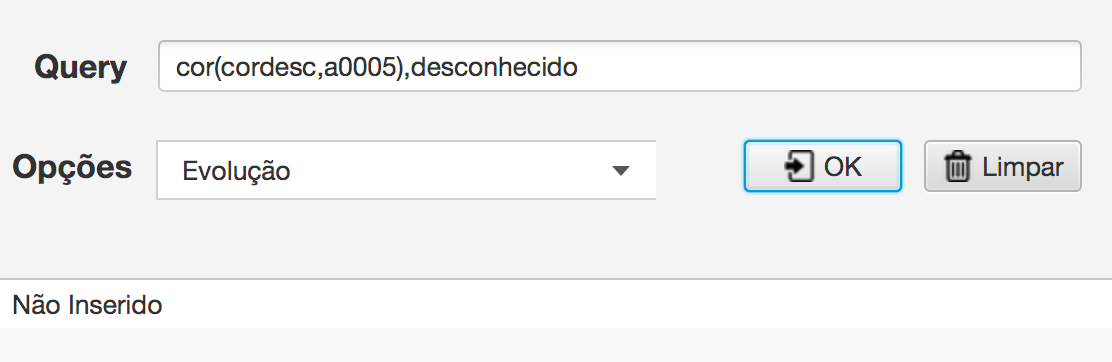
****

Figura - Inserção de conhecimento incerto com negativo na base

* + - Conhecimento desconhecido incerto (repetido) – É esperado que não se insira.

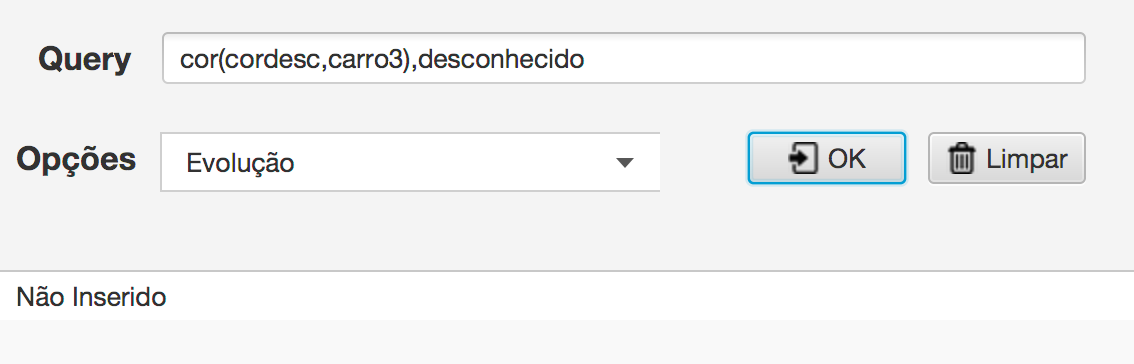
****

Figura - Inserção de conhecimento incerto com conhecimento incerto na base

* + - Conhecimento desconhecido impreciso – É esperado que não seja inserido na base de conhecimento.

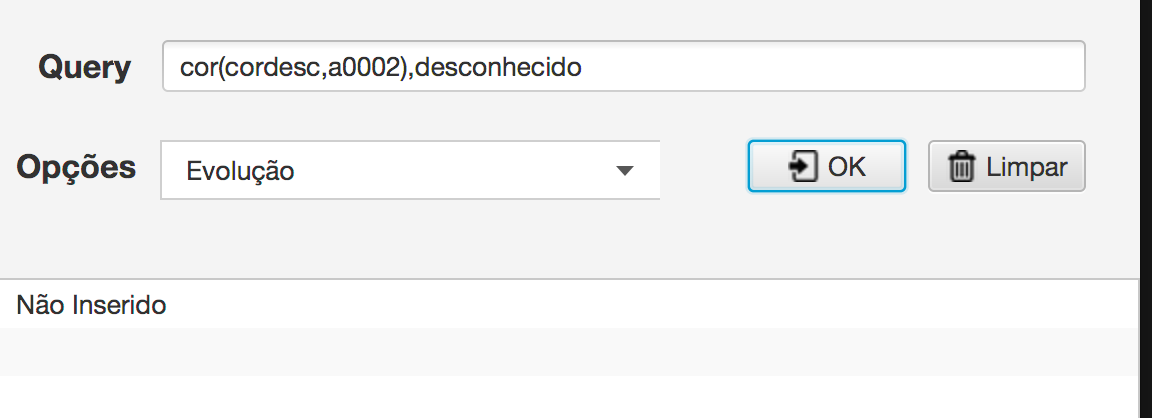


Figura - Inserção de conhecimento incerto com impreciso na base

* + Introdução de conhecimento imperfeito do tipo impreciso e na base de conhecimento tem-se:
    - Conhecimento positivo – É esperado que não seja inserido.

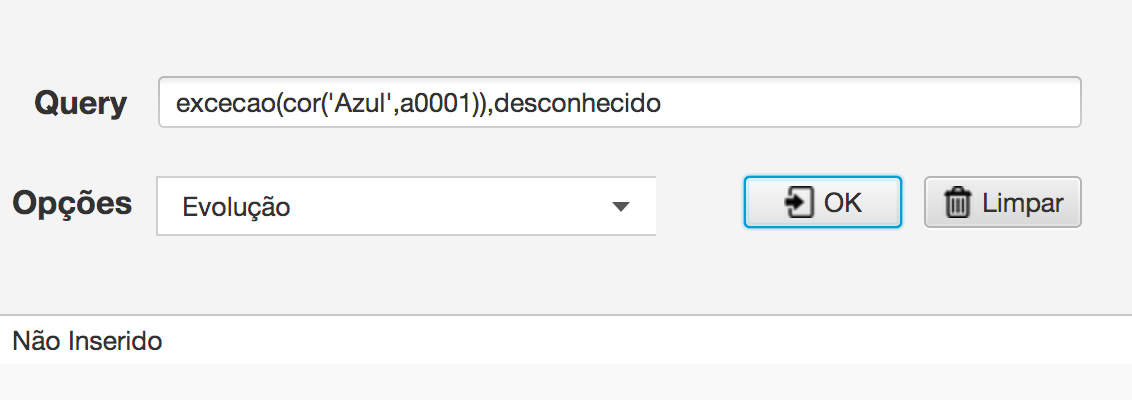
****

Figura - Inserção de conhecimento impreciso com positivo na base

* + - Conhecimento negativo – É esperado que não seja inserido.

****

Figura - Inserção de conhecimento impreciso com negativo na base

* + - Conhecimento desconhecido incerto – É esperado que o conhecimento seja inserido, note-se que apenas é inserido um facto de cada vez!

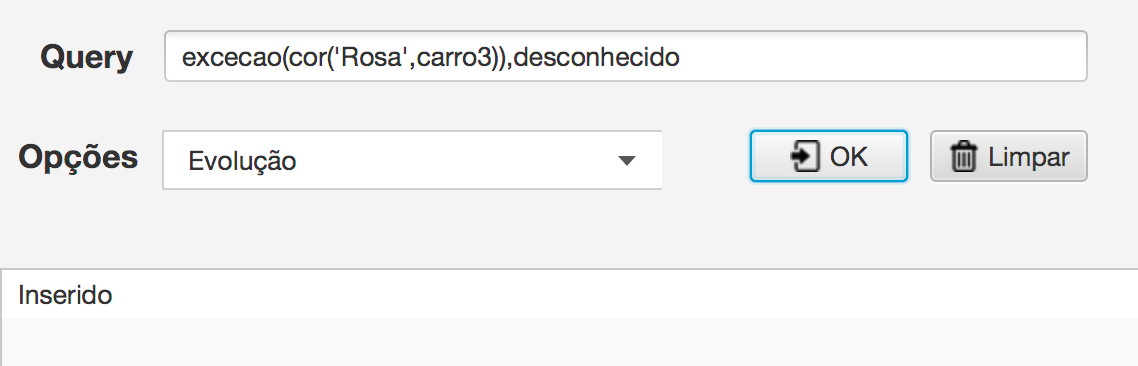
****

Figura - Inserção de conhecimento impreciso com incerto na base

* + - Conhecimento desconhecido impreciso – É esperado que seja inserido, adicionando à gama de imprecisão já existente o valor submetido.

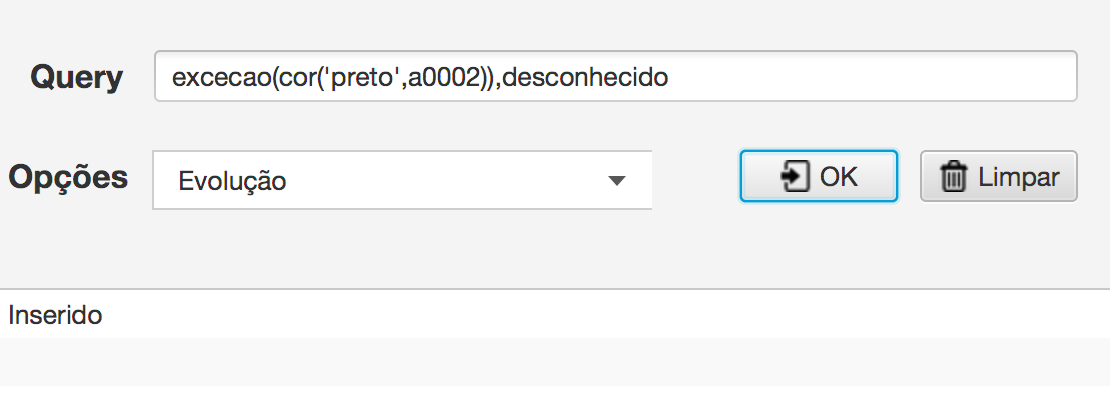
****

Figura - Inserção de conhecimento impreciso com impreciso na base

1. **Conclusões e Sugestões**

As maiores dificuldades prenderam-se nas decisões em relação ao contexto, ou seja, apesar de ter sido decidido aplicar o PMF foi bastante discutida a sua aplicação. Acabou-se por concordar que, no universo do *stand,* só existiria dado conhecimento se houvesse registo desse conhecimento.

Acima de tudo a maior dificuldade foi na evolução dos predicados demo e evolucao que foram um pouco trabalhosas e custosas de conseguir, não foi de nosso entendimento que o tivéssemos de aplicar e por consequente, quando assim o percebemos, tornou-se ainda mais desafiante a sua implementação.

A aplicação dos três tipos de conhecimento imperfeito não levantou quaisquer problemas e, todos os resultados dos testes e exemplos práticos foram os esperados.

Após identificadas as condições de inserção de registo, a construção dos invariantes foi também bastante simples e rápida.

Com a finalização deste trabalho prático conseguimos perceber o nível de conhecimento do nosso grupo neste tema em particular, praticar definições e construção de predicados recorrendo à extensão da programação em lógica e entender de forma mais completa o porquê da sua existência e necessidade.