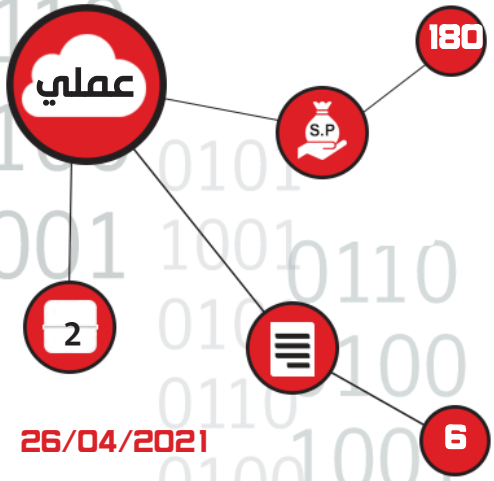


Intro to Prolog

محتوى مجاني غير مخصص للبيع التجاري

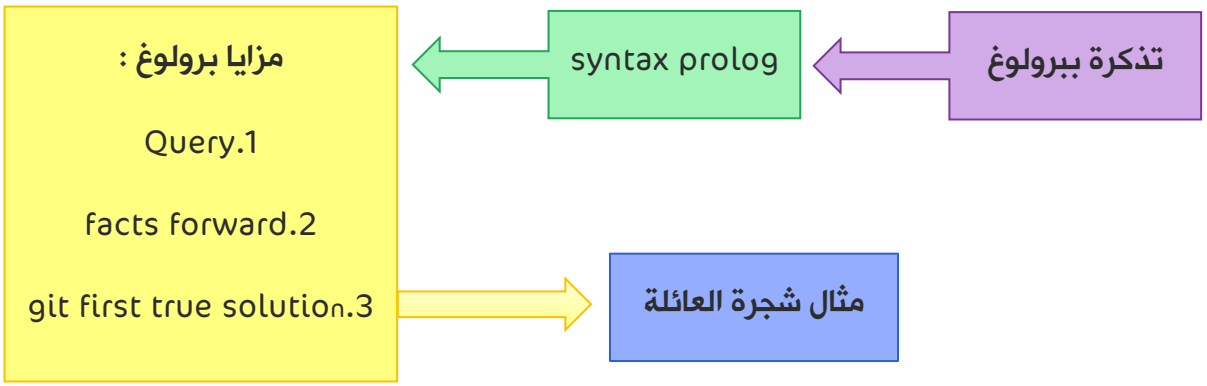
عملي مشترك

26/04/2021

RB Informatics;

مبادئ الذكاء الصنعي

محتويات المحاضرة



تذكرة:

- البرولوغ : هي لغة تصريحية تستخدم لتوصيف خبرة الخبير ضمن مجال معين و هي قابلة للتوصيف على شكل Rules وهذه الـ Rules تكون من الشكل if...else...then.
- و لكتابة أي برنامج منطقي ضمن برولوغ يجب أن يحتوي على:

1.Rules 2.Facts 3.Variable 4.Querys

Prolog syntax:

و لدينا مجموعة من القواعد التي يجب الالتزام بها أثناء العمل على البرنامج:

1. Constants (الثوابت):

يجب أن تبدأ بحرف صغير و يمكن أن يتبعه أي رقم أو حرف.

مثال: (ahmad,x5,xY,x).

2. Variables (المتحولات):

يجب أن تبدأ بحرف كبير و يمكن أن يتبعه أي رقم أو حرف.

مثال: (X,X5,Xo).

3. Facts (الحقائق):

يجب أن تبدأ بحرف صغير و يمكن أن يتبعه أي رقم أو حرف.

مثال: father(omar,ahmad)

4. Logical Operation (العمليات المنطقية):

a. And → ,

b. Or → ;

c. Not Equal → \=

5. Rules (القواعد):

father(Y,X) :- parent(Y,X) , male(Y)

6. يجب أن تنتهي الحقائق و القواعد بـ " . ".

ملاحظة:

قاعدة father(X,Y):
تعني يكون X أب لـ Y إذا
حقق parent(X,Y)
وكان X ذكراً أي
يحقق male(X) .

Example:

parent(ahmad,sami).
parent(salma,ahmad).
male(sami).
female(salma).

حقائق

father(X,Y) :- parent(X,Y) , male(X). قاعدة

القواعد التي تكون مكتوبة على شكل if...then....

تكتب ضمن برولوغ بالشكل then... : - if....

مثال:

father(X,Y) :- parent(X,Y) , male(Y).
mother(X,Y) :- parent(X,Y) , female(Y).

Rules :

Rule 1 : if Y parent X and Y male, then Y father X

Rule 2 : if Y parent X and Y female, then Y mother X

- البرولوغ هو tool او inference engine.

لكتابة برنامج ضمن بيئة swi-prolog :

File->new ثم تفتح نافذة جديدة نقوم بكتابة الحقائق والقواعد ثم File -> save ضمن هذه الواجهة

وضمن الواجهة الثانية نقوم بـ file->consult ومن ثم نقوم بكتابة الطلبات (Querys) ضمن الواجهة الأولى .

ملاحظة : عند إجراء أي تعديل ضمن الحقائق يجب علينا القيام بالخطوات السابقة نفسها .

مزايا برولوغ:

1. Query based:

حيث بعد كتابة الحقائق والقواعد ضمن البرنامج المنطقي يمكننا الاستعلام ضمنها .

يوجد ضمن برولوغ نوعان من الطلبات :

- استعلامية(استفهامية):

حيث يكون الخرج قيمة بولیانية true or false.

فالطلب يحتوي على قيم صريحة (ahmed,rama,...).

مثال:

```
male(ahmed).
male(fadi).
male(ramez).
```

```
?- male(ahmed).
true.

?- male(rama).
false.
```

- matching:

حيث الخرج هو القيمة التي تحقق الطلب أو false أي لم يجد جواب مطابق.

فالطلب يحتوي على متغيرات (X,Y,...).

مثال:

```
male(ahmed).
male(fadi).
male(ramez).
```

```
?- male(X).
X = ahmed ;
X = fadi ;
X = ramez.
```

2. Facts forward:

أي يقوم بالتحقق من الطلب ضمن الحقائق من البداية إلى النهاية ويكون الخرج حسب ترتيب ورود الحقائق المدخلة، حيث ضمن المثال السابق دوماً جواب الطلب الذي استعلمنا عليه سوف يكون ضمن هذا الترتيب طالما الحقائق مرتبة كما هي موجودة ضمن ملف الحقائق والقواعد.

3. Get first true solution:

يقوم بإعطاء أول حل صحيح من مجموعة الحلول الصحيحة.

مثال :

sec2.pl

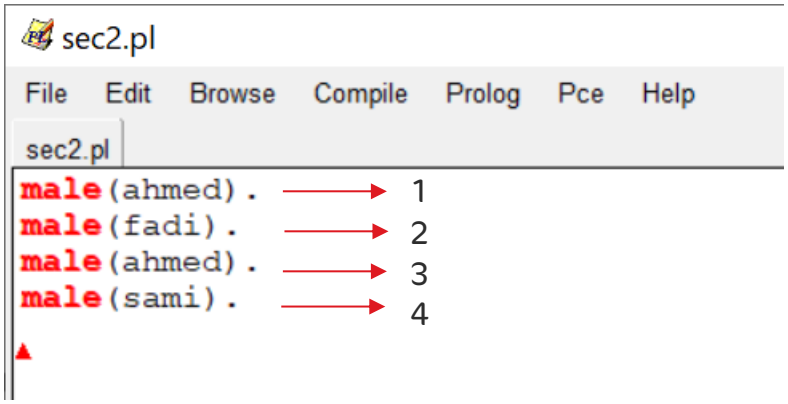
File Edit Browse Compile Prolog Pce Help

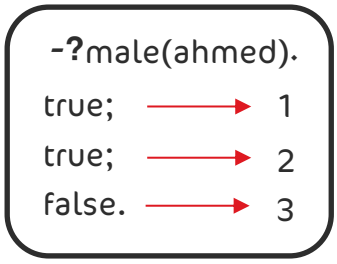
sec2.pl

```
male(ahmed).    → 1
male(fadi).     → 2
male(ahmed).    → 3
```

```
?- male(ahmed).
true ;          → 1
true.           → 2
```

☜ حيث أن الجواب الصحيح الأول هو نتيجة للحقيقة رقم 1 والجواب الصحيح الثاني هو اعتماداً على الحقيقة رقم3 و يقوم أيضاً باختصار مجموعة الحلول الخاطئة بعد آخر حل صحيح بحل خاطئ وحيد،
وتجاهل مجموعة الحلول الخاطئة بين حلول صحيحة.
حيث يقوم بإظهار حل خاطئ وحيد بعد آخر حل صحيح أن وجد حلول خاطئة بعد هذا الحل الصحيح.
مثال:



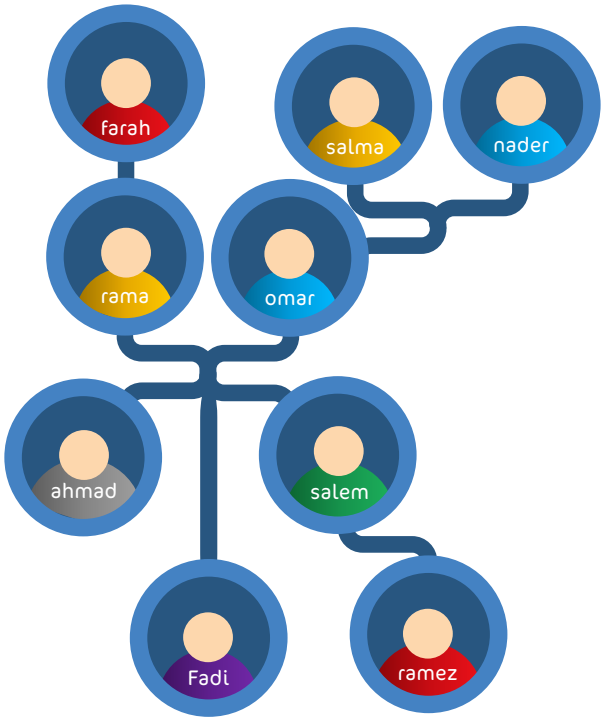


حيث أن الجواب الصحيح الأول هو نتيجة للحقيقة رقم 1 والجواب الصحيح الثاني هو اعتماداً على الحقيقة رقم 3 والجواب الخاطئ نتيجة للحقيقة رقم 4 حيث أن الجواب الخاطئ للحقيقة رقم 2 تم تجاهله.

ملاحظة : في بعض النسخ من swi-prolog يقوم بتجاهل إظهار الحل الخاطئ بعد آخر حل صحيح ضمن الطلبات الاستفهامية .

تمرين:

ليكن لدينا شجرة العائلة التالية :



والمطلوب:

كتابة الحقائق parent , female , male.

وكتابة القواعد ancestor , sibling , uncle , mother , father.

البرنامج:

parent facts

```
parent(rama , farah).
parent(omar , salma).
parent(omar , nader).
parent(ahmed , rama).
parent(fadi , rama).
parent(salem , rama).
parent(ahmed , omar).
parent(fadi , omar).
parent(salem , omar).
parent(ramez , salem).
```

female facts:

```
female(salma).
female(farah).
female(rama).
```

male facts:

```
male(nader).
male(omar).
male(ahmad).
male(fadi).
male(salem).
male(ramez).
```

Mother rules

```
mother(X,Y) :- female(Y) , parent(X,Y).
```

حيث Y أم لـ X عندما يكون Y أنثى و Y ولي لـ X.

father rules

```
father(X,Y) :- male(Y) , parent(X,Y).
```

حيث Y أب لـ X عندما يكون Y ذكر و Y ولي لـ X.

Sibling rules

```
sibling(X,Y) :- father(X,F) , father(Y,F) , mother(X,M) , mother(Y,M) , X\=Y.
```

يكون X شقيق/ة لـ Y إذا كان Z أب مشترك لهما و M أم مشترك لهما وكان X لا يساوي Y.

ancestor rules

```
ancestor(X,Y) :- parent(X,Z), ancestor(Z,Y).
ancestor(X,Y) :- parent(X,Y).
```

يكون Y سلف لـ X اذا كان Z أب X و Y سلف لـ Z

حيث شرط التوقف هنا أن يكون Y سلف لـ X إذا كان وليه

uncle rules

```
uncle(X,Y) :- father(X,Z) , sibling(Z,Y).
```

يكون Y عم/ة لـ X إذا كان Z أب لـ X و Z شقيق/ة Y.

القاعدة السابقة تعتمد على العودية وعلى قاعدة أخرى التي تمثل شرط التوقف

وهي تكافئ في اللغات عالية المستوى التابع التالي:

ancestor.c++

```
1   #include<iostream>
2
3   ancestor(X,Y){
4       if(parent(X,Y)) return X&Y;
5       return parent(X,Z)&ancestor(Z,Y);
6
7   }
```

ancestor(X,Y) :- parent(X,Z), ancestor(Z,Y).
ancestor(X,Y) :- parent(X,Y).

ملاحظة: عند كتابة قاعدة تعتمد على العودية يجب مراعاة ترتيب الاستدعاء العودي.

أي في القاعدة السابقة من الخطأ كتابة :

ancestor(X,Y) :- **ancestor(Z,Y)**, parent(X,Z).

حيث يؤدي هذا إلى مشكلة تشابه stack overflow .

انتهت المحاضرة ♡