Merci pour les réponses à la 2,3, et 5, ça nous a bien aidé à comprendre.

Pour la question 1, nous avons toujours de la difficulté à comprendre cette ligne de code. Si on se fie à la documentation de GNU prolog, on a que :

“var(Term) succeeds if Term is currently uninstantiated (which therefore has not been bound to anything, except possibly another uninstantiated variable). “,  
“atom(Term) succeeds if Term is currently instantiated to an atom.”  
et  
“ground(Term) succeeds if Term is a ground term.”.

(source : <http://www.gprolog.org/manual/gprolog.html#var%2F1>)

Donc, selon ces définitions, j’en comprends que (si on peut se permettre de les représenter comme ça), si on dit que “var“ est un ensemble et “atom“ est un ensemble, “var“ et “atom“ sont des ensembles disjoints, c-à-d : une “var“ ne peut jamais être un “atom“ et un “atom“ ne peut jamais être une “var“. Est-ce que je me trompe dans mon raisonnement?

Ensuite, en essayant de comprendre la différence entre un “atom“ et une “var“, nous sommes tombés sur ceci :

“Let’s see how they behave.

?- atom(a).

yes

?- atom(7).

no

?- atom(loves(vincent,mia)).

no

These three examples behave exactly as we would expect. But what happens, when we call atom/1 with a variable as argument?

?- atom(X).

no

This makes sense, since an uninstantiated variable is not an atom. However if we instantiate X with an atom first and then ask atom(X) , Prolog answers yes.

?- X = a, atom(X).

X = a

yes

But it is important that the instantiation is done before the test:

?- atom(X), X = a.

no” (source : <https://bit.ly/3fzIuPu>)

On voit ici qu’une variable ne peut pas être un “atom“, ce qui me rend encore plus confus.

Cependant, une variable peut être un atom si cette variable a été unifiée avec un “atom“.

Donc, pour que la ligne “type(var(X)) :- atom(X).“ soit valide, il faudrait qu’entre le moment où on vérifie que X est une variable et que X est un atom, X soit unifié avec un atom?

Donc nous avons effectué quelques tests nous-même :

“

| ?- var(X), X = a, atom(X).

X = a

yes

| ?- var(X), X = a, atom(X), var(X).

no“

Donc à partir du moment où on a unifié la variable “X“ avec un “atom“, “X“ n’est plus une variable.

Donc une variable ne peut pas être un “atom“.

En prenant un peu de recul nous avons réalisé ceci :

On sait aussi que les arguments d’une clause sont traités comme des données littéralement, donc que fait/signifie exactement “type(var(X)) :- atom(X).“, plus particulièrement “type(var(X))“?

Est-ce voulu que var(X) et atom(X) soient mutuellement exclusifs et on s’attend justement à ce que cette clause ne soit jamais validée? Ça me semble peu probable mais j’ai du mal à comprendre …

Qu’est-ce que “var(X) est un type“ signifie, surtout considérant le fait que X est un atom et que var(X) est ici un test, qui est un type?