

CLASSIFICATION DES DINOSAURES

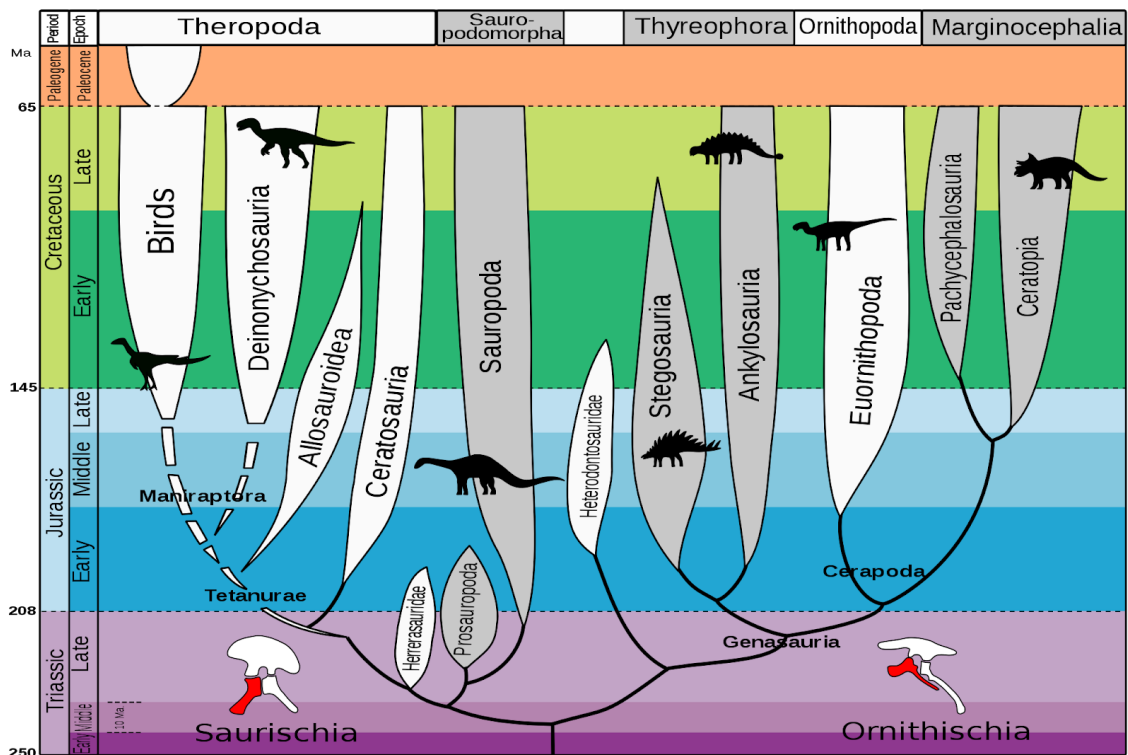


Figure 0: Image représentant la classification des dinosaures [1].

Auteurs

Alex Lebrun - alex.lebrun.1@ulaval.ca

Walter Bonetti - walter.bonetti.1@ulaval.ca

Isabelle Eysseric - isabelle.eysseric.1@ulaval.ca

1. Description du sujet d'expertise	3
1.1 Schéma conceptuel	4
1.2 Problème à résoudre	9
1.3 Base de connaissance	10
2. Éléments du système développé	11
2.1 Règles expertes	11
2.2 Fichiers nécessaires à l'exécution	12
2.3 Guides	12
3. Validation des résultats	15
3.1 Cas-tests	15
3.2 Résultats obtenues	20
3.3 Discussion sur les résultats	21
4. Bilan de l'expérimentation	22
4.1 Avantage d'utiliser un SBC	22
4.2 Difficulté pour déterminer l'expertise	23
4.3 Avantages et contraintes de la coquille	25
4.4 Avantages et limites du système	26
4.5 Amélioration à apporter au système	26
5. Bibliographie	27
6. Annexes	27

1. Description du sujet d'expertise

Dans le cadre de ce TP nous allons prendre un sujet bien connu, les dinosaures. Ils sont souvent très appréciés des enfants, mais leur classification demeure un casse-tête même pour les adultes.

Dans cette section, nous analyserons et détaillerons le problème en l'accompagnant de son schéma conceptuel et de sa base de connaissances.

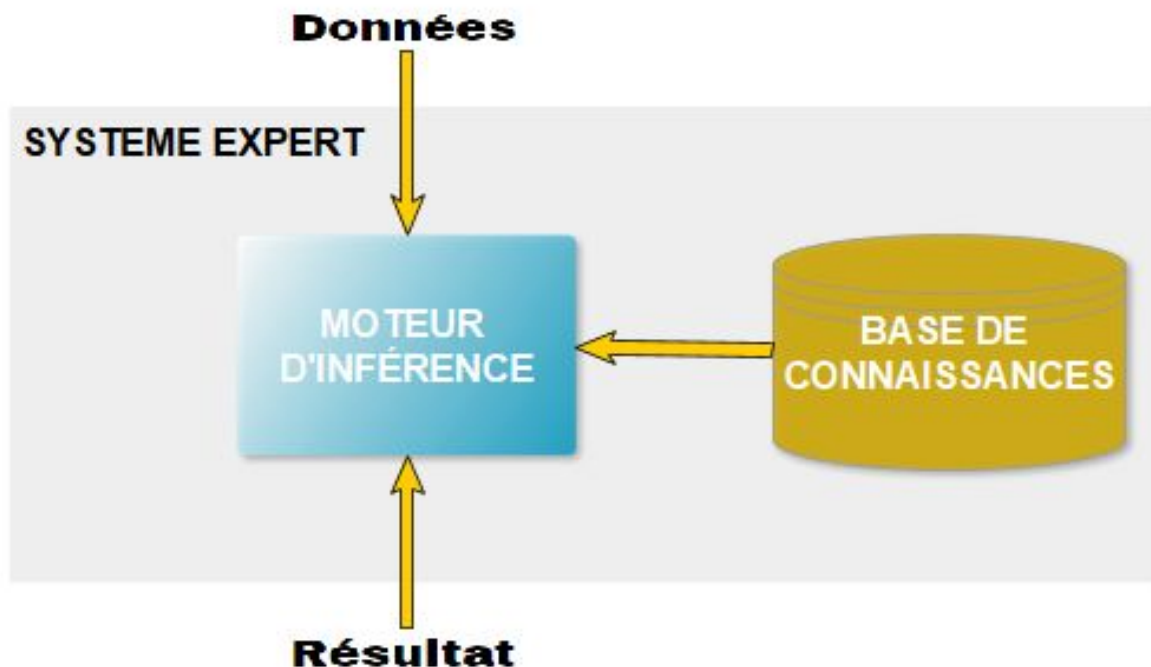


Figure 1: Représentation des composantes de base d'un système expert. Le moteur d'inférence fait fonctionner le système expert et la base de connaissances lui donne les connaissances nécessaires. L'utilisateur pose la question (données) et le système donne la réponse (résultat d'expertise) [2].

1.1 Schéma conceptuel

Ce diagramme est présenté uniquement pour donner une idée générale du réseau de sémantique. Une version en parties séparées (plus facile à lire) suit aux pages suivantes.

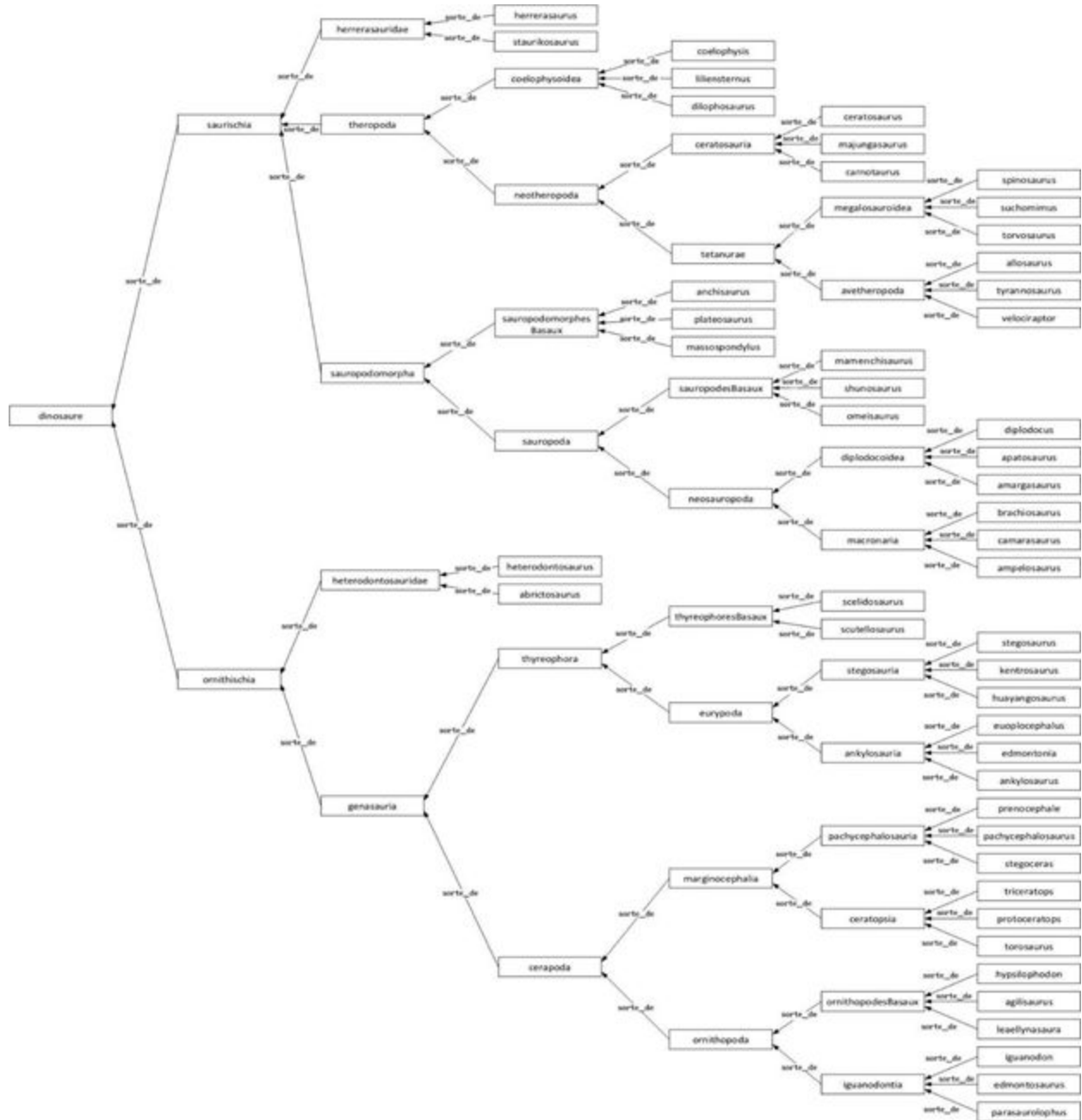


Figure 2: Diagramme visualisant toutes les catégories de dinosaures.

Comme le schéma sémantique est grand et difficile à mettre dans une page, le voici séparé en plusieurs morceaux.

La partie ci-bas représente les catégories principales de dinosaures. Les catégories *genasauria*, *sauropodomorpha* et *theropoda* sont détaillées dans les pages suivantes.

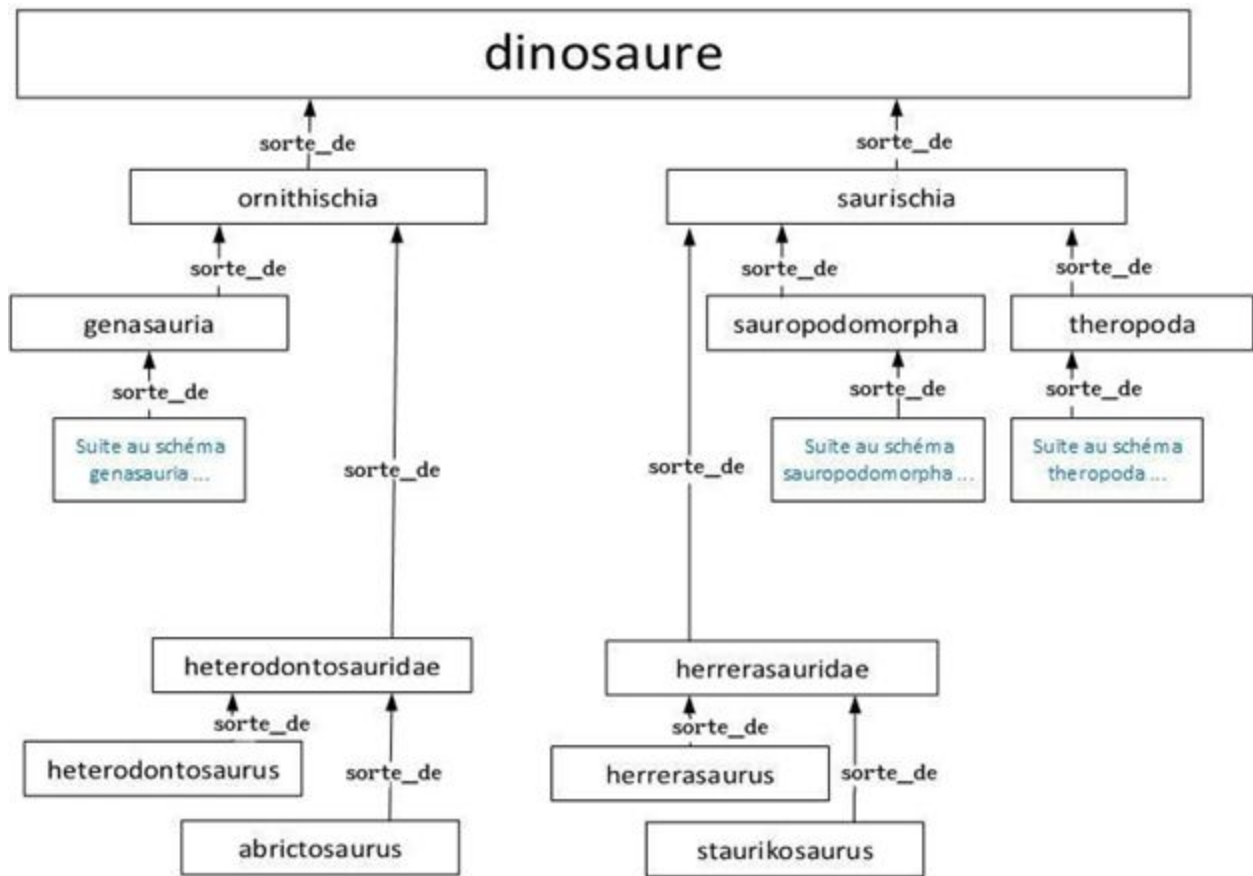


Figure 3: Schéma sémantique des principales catégories de dinosaures. Lorsque vous arrivez à une case avec du texte en bleu, poursuivez au diagramme indiquer pour poursuivre dans cette branche.

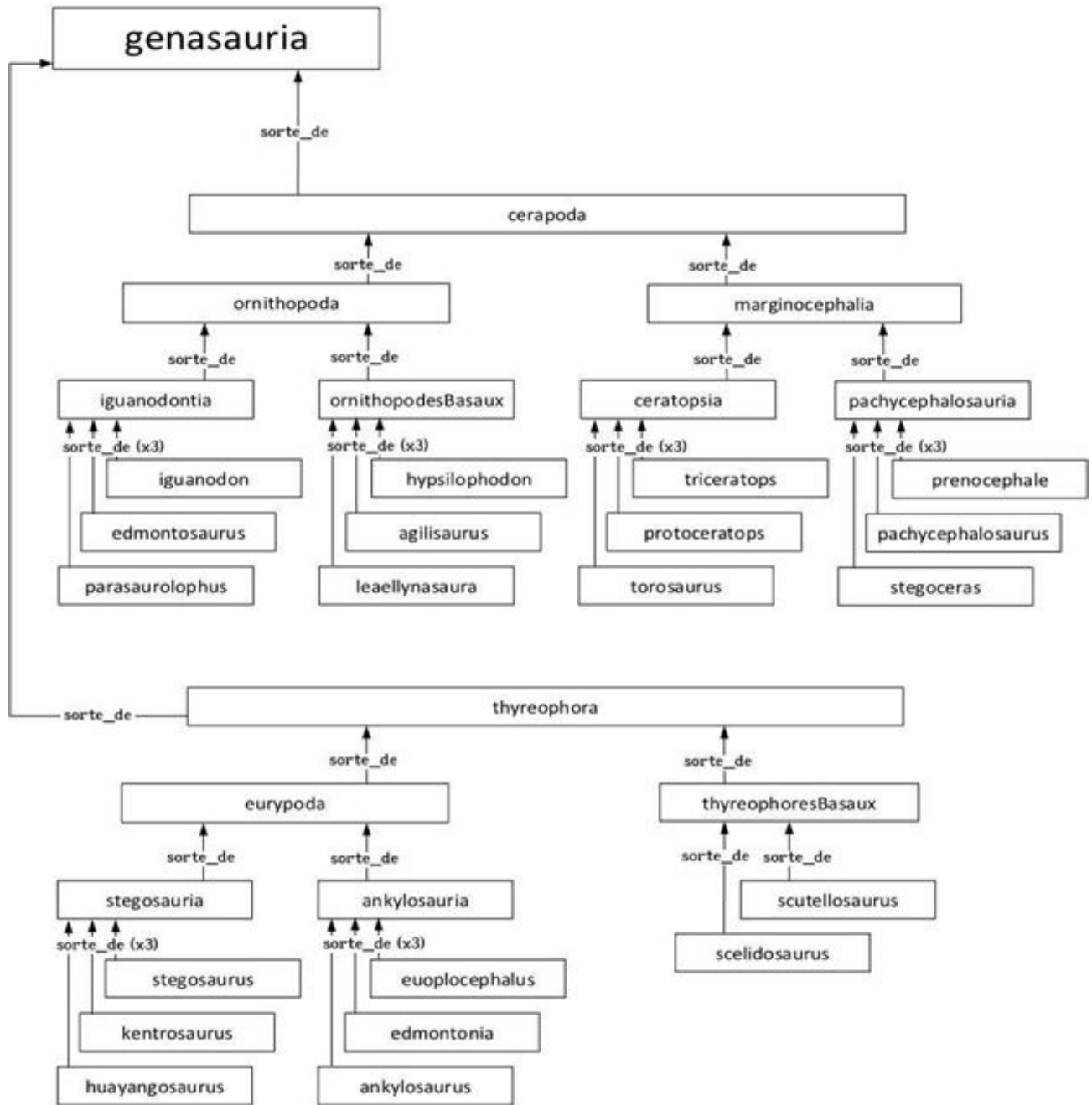


Figure 4: Schéma sémantique de *genasauria*, sous catégorie de dinosaures. Un dinosaure bien connu de cette catégorie est l'*ankylosaurus* ou bien encore le *triceraptops*.

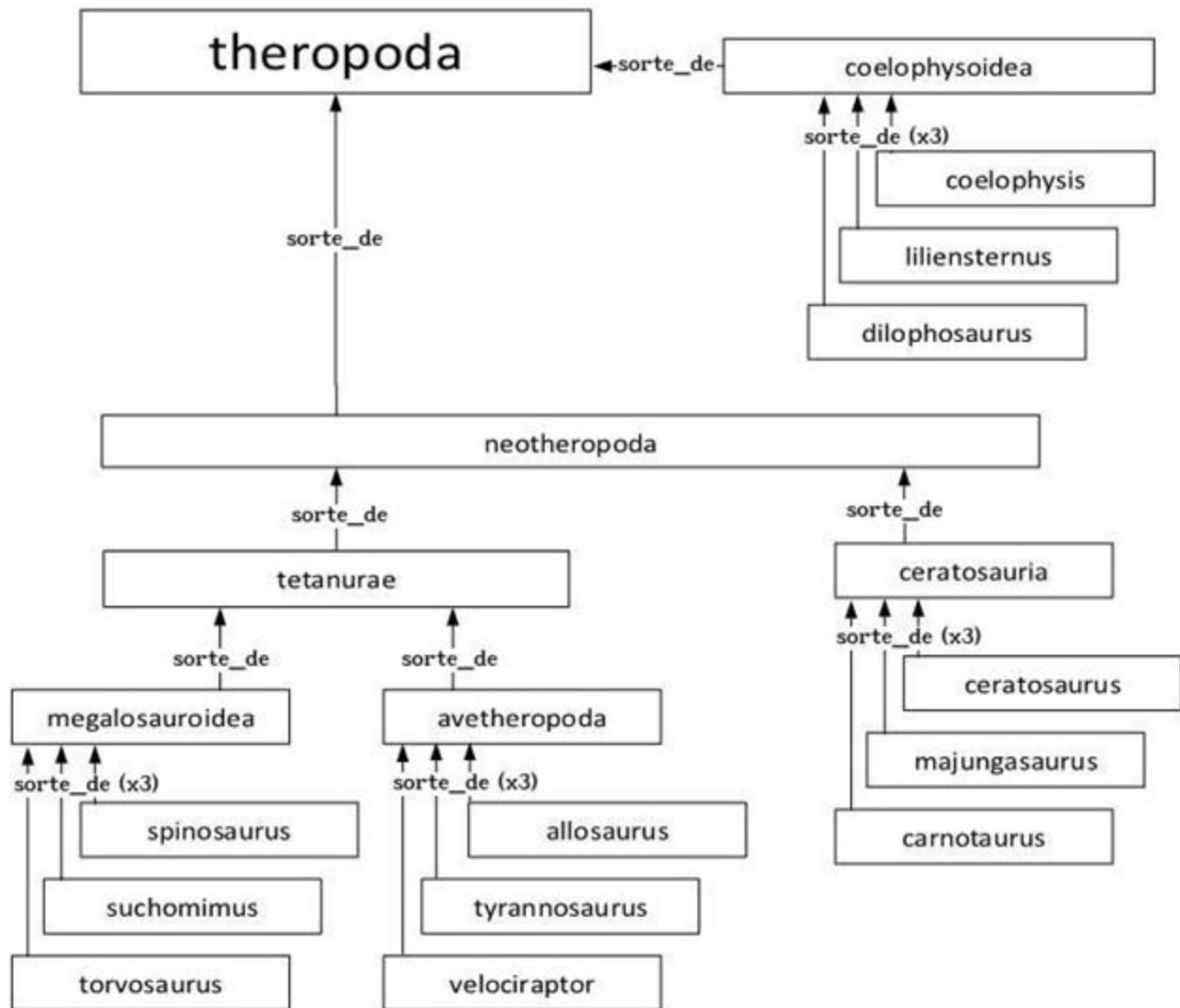


Figure 5: Schéma sémantique de theropoda, sous catégorie de dinosaures. Un dinosaure bien connu de cette catégorie est le tyrannosaurus ou bien encore le velociraptor.

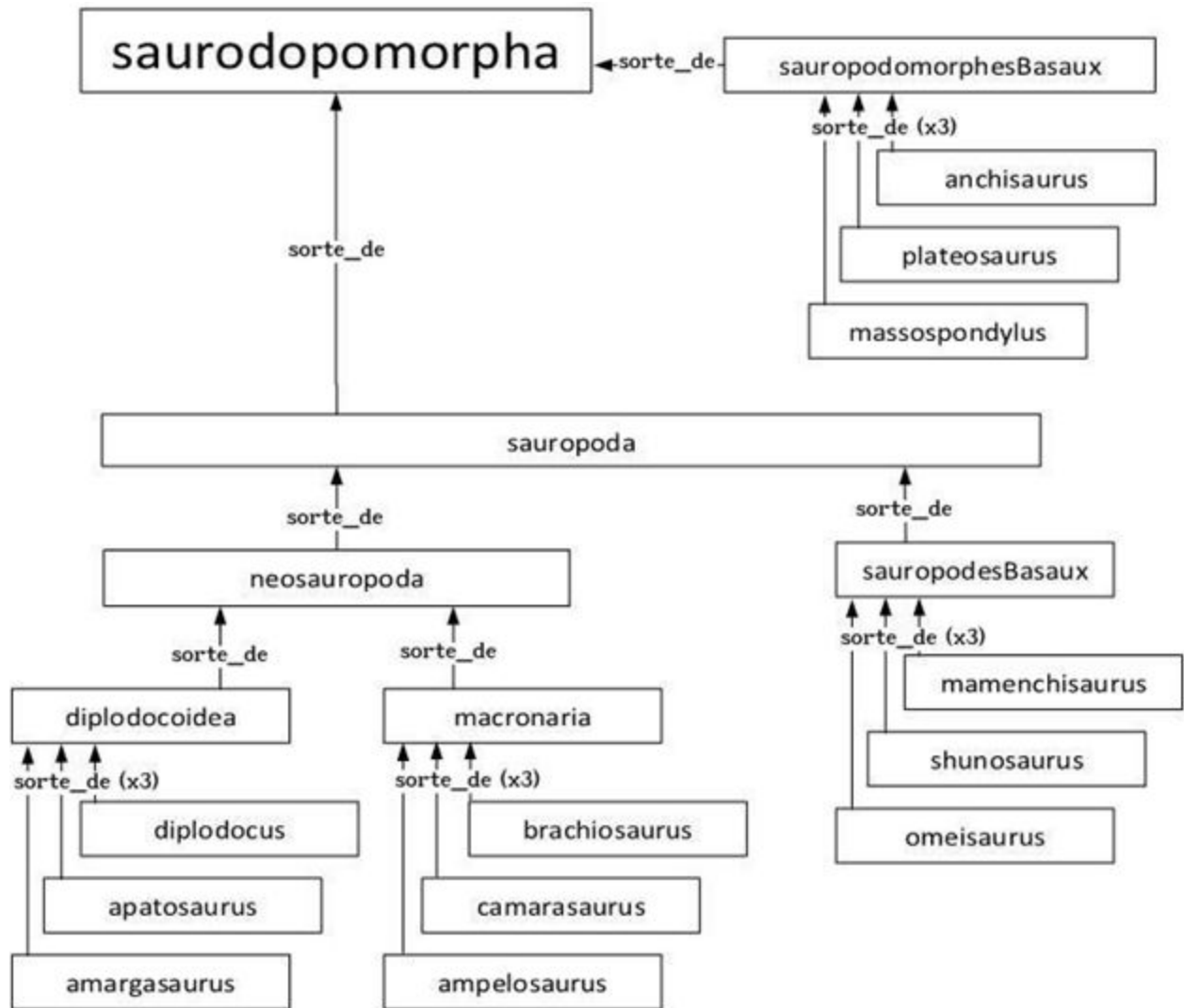


Figure 6: Schéma sémantique de "sauropodomorpha", une sous catégorie de dinosaures. Un dinosaure bien connu de cette catégorie est le "diplodocus" ou encore le "brachiosaure".

1.2 Problème à résoudre

Nous allons concevoir une banque de connaissance qui pourra vérifier une connaissance : ce dinosaure appartient à la classe X et nous retourner "true" si cette affirmation est vraie.

Nous allons fournir également une fonction qui donnera un exemple de dinosaure appartenant à la classe donnée en paramètre.

Nous utiliserons une méthode récursive pour chercher dans la base de connaissance comme dans l'exemple si bas.

```
classeW(dinosaureX) .
```

```
R1 = si classeW(dinonosaureX) alors classeX(dinonosaureX) .
```

```
R2 = si classeX(dinonosaureX) alors classeY(dinonosaureX) .
```

```
R3 = si classeY(dinonosaureX) alors classeZ(dinonosaureX) .
```

```
...
```

Pour finalement associer le `dinosaureX` à la classe voulue.

Puisque notre système à base de connaissances est de type général, nous utiliserons les règles comme stratégie de résolution de problème. De plus, notre problème étant d'ordre 0, c'est-à-dire fondé sur des calculs propositionnels avec des faits à valeurs booléennes, nous utiliserons le chaînage avant et arrière comme méthode de résolution de problèmes.

1.3 Base de connaissance

%Faits saurischia

fait(herrerasauridae(herrerasaurus)).
 fait(herrerasauridae(staurikosaurus)).
 fait(coelophysoidea(coelophysus)).
 fait(coelophysoidea(liliensternus)).
 fait(coelophysoidea(dilophosaurus)).
 fait(ceratosauria(ceratosaurus)).
 fait(ceratosauria(majungasaurus)).
 fait(ceratosauria(carnotaurus)).
 fait(megalosauroides(spinosaurus)).
 fait(megalosauroides(suchomimus)).
 fait(megalosauroides(torvosaurus)).
 fait(avetheropoda(allosaurus)).
 fait(avetheropoda(tyrannosaurus)).
 fait(avetheropoda(velociraptor)).
 fait(sauropodomorphesBasaux(anchisaurus)).
 fait(sauropodomorphesBasaux(plateosaurus)).
 fait(sauropodomorphesBasaux(massospondylus)).
 fait(sauropodesBasaux(mamenchisaurus)).
 fait(sauropodesBasaux(omeisaurus)).
 fait(sauropodesBasaux(shunosaurus)).
 fait(diplodocoidea(apatosaurus)).
 fait(diplodocoidea(amargasaurus)).
 fait(macronaria(brachiosaurus)).
 fait(macronaria(camarasaurus)).
 fait(macronaria(ampelosaurus)).

%Faits ornithischia

fait(heterodontosauridae(heterodontosaurus)).
 fait(heterodontosauridae(abricotosaurus)).
 fait(thyreophoresBasaux(scelidosaurus)).
 fait(thyreophoresBasaux(scutellosaurus)).
 fait(stegosauria(stegosaurus)).
 fait(stegosauria(kentrosaurus)).
 fait(stegosauria(huayangosaurus)).
 fait(ankylosauria(euoplocephalus)).
 fait(ankylosauria(edmontonia)).
 fait(ankylosauria(ankylosaurus)).
 fait(pachycephalosauria(prenocephale)).
 fait(pachycephalosauria(pachycephalosaurus)).
 fait(pachycephalosauria(stegoceras)).
 fait(ceratopsia(triceratops)).
 fait(ceratopsia(protoceratops)).
 fait(ceratopsia(torosaurus)).
 fait(ornithopodesBasaux(hypsilophodon)).
 fait(ornithopodesBasaux(hypsilophodon)).
 fait(ornithopodesBasaux(agilisaurus)).
 fait(ornithopodesBasaux(leaellynasaura)).
 fait(iguanodontia(iguanodon)).
 fait(iguanodontia(edmontosaurus)).
 fait(iguanodontia(parasaurolophus)).

2. Éléments du système développé

2.1 Règles expertes

R1 = si ornithischia(X) alors dinosaure(X).
R2 = si saurischia(X) alors dinosaure(X).
R3 = si iguanodontia(X) alors ornithopoda(X).
R4 = si ornithopodesBasaux(X) alors ornithopoda(X).
R5 = si ceratopsia(X) alors marginocephalia(X).
R6 = si pachycephalosauria(X) alors marginocephalia(X).
R7 = si ornithopoda(X) alors cerapoda(X).
R8 = si marginocephalia(X) alors cerapoda(X).
R9 = si ankylosauria(X) alors euryopoda(X).
R10 = si stegosauria(X) alors euryopoda(X).
R11 = si enryopoda(X) alors thyreophora(X).
R12 = si thyreophoresBasaux(X) alors thyreophora(X).
R13 = si cerapoda(X) alors genasauria(X).
R14 = si thyreophora(X) alors genasauria(X).
R15 = si genasauria(X) alors ornithischia(X).
R16 = si heterodontosauridae(X) alors ornithischia(X).
R17 = si macronaria(X) alors neosauropoda(X).
R18 = si diplodocoidea(X) alors neosauropoda(X).
R19 = si neosauropoda(X) alors sauropoda(X).
R20 = si sauropodesBasaux(X) alors sauropoda(X).
R21 = si sauropoda(X) alors sauropodomorpha(X).
R22 = si sauropodomorphesBasaux(X) alors sauropodomorpha(X).
R23 = si avetheropoda(X) alors tetanurae(X).
R24 = si megalosauroida(X) alors tetanurae(X).
R25 = si tetanurae(X) alors neotherapoda(X).
R26 = si ceratusauria(X) alors neotherapoda(X).
R27 = si neotherapoda(X) alors theropoda(X).
R28 = si coelophysoidea(X) alors theropoda(X).
R29 = si sauropodomorpha(X) alors saurischia(X).
R30 = si theropoda(X) alors saurischia(X).
R31 = si herrerasauridae(X) alors saurischia(X).

2.2 Fichiers nécessaires à l'exécution

- dinosaures.pl

2.3 Guides

Démarrez Prolog et aller dans File/Edit... et choisissez "dinosaures.pl".
Prenez l'option Compile/Compile buffer dans la nouvelle fenêtre.
Retourner à la console (fenêtre précédente).

Le programme est muni de deux fonctions pour l'utilisateur:

1. Pour trouver si un dinosaure appartient à une famille précise, nous utiliserons la fonction:

dinosaure_est(X,Y).

où **X** est la classe de dinosaure

Y est le nom du dinosaure

```
?- dinosaure_est(genasauria, ankylosaurus).  
true .  
  
?- dinosaure_est(heterodontosauridae, ankylosaurus).  
false .
```

Figure 7: exemple avec la question "dinosaure_est(). La première est vrai, par contre la deuxième est fausse et le programme vous affiche false quand le dinosaure n'appartient pas à la bonne famille.

2. Pour avoir le nom d'un dinosaure appartenant une famille demandée:

exemple_de(Famille(X)).

Ici nous recherchons un dinosaure **X** Appartenant à la famille **Famille**.

```
?- exemple_de(herrerasauridae(X)).
nouveau fait : ornithopoda(iguanodon)
nouveau fait : marginocephalia(triceratops)
nouveau fait : cerapoda(iguanodon)
nouveau fait : thyreophora(euoplocephalus)
nouveau fait : genasauria(iguanodon)
nouveau fait : ornithischia(iguanodon)
nouveau fait : neosauropoda(brachiosaurus)
nouveau fait : sauropoda(brachiosaurus)
nouveau fait : sauropodomorpha(brachiosaurus)
nouveau fait : tetanurae(allosaurus)
nouveau fait : neotherapoda(allosaurus)
nouveau fait : therapoda(allosaurus)
nouveau fait : saurischia(brachiosaurus)
Plus de nouveaux faits d'Ã©crits, la BC est saturÃ©e.
X = herrerasaurus ■
```

Figure 8: exemple avec la question "exemple_de().

Voici les choix possibles pour pouvoir utiliser la base de connaissance:

Dinosaures:

<i>herrerasaurus</i>	<i>staurikosaurus</i>	<i>coelophys</i>	<i>liliensternus</i>
<i>dilophosaurus</i>	<i>ceratosaurus</i>	<i>majungasaurus</i>	<i>carnotaurus</i>
<i>spinosaurus</i>	<i>suchomimus</i>	<i>torvosaurus</i>	<i>allosaurus</i>
<i>tyrannosaurus</i>	<i>velociraptor</i>	<i>anchisaurus</i>	<i>plateosaurus</i>
<i>massospondylus</i>	<i>mamenchisaurus</i>	<i>omeisaurus</i>	<i>shunosaurus</i>
<i>apatosaurus</i>	<i>amargasaurus</i>	<i>brachiosaurus</i>	<i>camarasaurus</i>
<i>ampelosaurus</i>	<i>iguanodon</i>	<i>abrichtosaurus</i>	<i>scelidosaurus</i>
<i>scutellosaurus</i>	<i>stegosaurus</i>	<i>kentrosaurus</i>	<i>huayangosaurus</i>
<i>euoplocephalus</i>	<i>edmontonia</i>	<i>ankylosaurus</i>	<i>prenocephale</i>
<i>stegoceras</i>	<i>triceratops</i>	<i>protoceratops</i>	<i>torosaurus</i>
<i>hypsilophodon</i>	<i>agilisaurus</i>	<i>leaellynasaura</i>	<i>heterodontosaurus</i>
<i>edmontosaurus</i>	<i>parasaurolophus</i>	<i>diplodocus</i>	<i>pachycephalosaurus</i>

Familles (classées par hiérarchie):

dinosaure

saurischia

herrerasauridae

therapoda

coelophysoidea

neotherapoda

ceratosauria

tetanurae

megalosauroida

avetheropoda

sauropodomorpha

sauropodomorphes Basaux

sauropoda

sauropodes Basaux

neosauropoda

diplodocoidea

macronaria

ornithischia

heterodontosauridae

genasauria

thyreophora

thyreophores Basaux

enrypoda

stegosauria

ankylosauria

cerapoda

marginocephalia

pachycephalosauria

ceratopsia

ornithopoda

ornithopodes Basaux

iguanodontia

3. Validation des résultats

3.1 Cas-tests

%Tests du réseau de connexions (chaque lien au moins une fois) ch_arriere

%Test pour <i>herrerosauridae->saurischia->dinsaure</i> dinsaure_est(dinsaure, herrerasaurus).	true.
%Test pour <i>coelophysoidea->therapoda->saurischia</i> dinsaure_est(saurischia, coelophysis).	true.
%Test pour <i>ceratosauria->neotherapoda->therapoda</i> dinsaure_est(theropoda, ceratosaurus).	true.
%Test pour <i>megalosauroides->tetanurae->neotherapoda</i> dinsaure_est(neotheropoda, spinosaurus).	true.
%Test pour <i>sauropodomorphesBasaux->sauropodomorpha->saurischia</i> dinsaure_est(saurischia, anchisaurus).	true.
%Test pour <i>sauropodesBasaux->sauropoda->sauropodomorpha</i> dinsaure_est(sauropodomorpha, mamenchisaurus).	true.
%Test pour <i>diplodocoidea->neosauropoda->sauropoda</i> dinsaure_est(sauropoda, diplodocus).	true.
%Test pour <i>macronaria->neosauropoda</i> dinsaure_est(neosauropoda, brachiosaurus).	true.
%Test pour <i>heterodontosauridae->ornithischia->dinsaure</i> dinsaure_est(dinsaure, heterodontosaurus).	true.
%Test pour <i>thyreophoresBasaux->thyreophora->genasauria->ornithischia</i> dinsaure_est(ornithischia, scelidosaurus).	true.
%Test pour <i>stegosauria->europa->thyreophora</i> dinsaure_est(thyreophora, stegosaurus).	true.
%Test pour <i>ankylosauria->europa</i> dinsaure_est(europa, ankylosaurus).	true.
%Test pour <i>pachycephalosauria->marginocephalia->cerapoda->genasauria</i> dinsaure_est(genasauria, pachycephalosaurus).	true.
%Test pour <i>ceratopsia->marginocephalia</i> dinsaure_est(marginocephalia, triceratops).	true.
%Test pour <i>ornithopodesBasaux->ornithopoda->cerapoda</i> dinsaure_est(cerapoda, hypsilophodon).	true.
%Test pour <i>iguanodontia->ornithopoda</i> dinsaure_est(ornithopoda, iguanodon).	true.

```
?- dinosaure_est(dinosaure, herrerasaurus).  
true .  
?- dinosaure_est(saurischia, coelophysus).  
true .  
?- dinosaure_est(theropoda, ceratosaurus).  
true .  
?- dinosaure_est(neotheropoda, spinosaurus).  
true .  
?- dinosaure_est(saurischia, anchisaurus).  
true .  
?- dinosaure_est(sauropodomorpha, mamenchisaurus).  
true .  
?- dinosaure_est(sauropoda, diplodocus).  
true .  
?- dinosaure_est(neosauropoda, brachiosaurus).  
true .  
?- dinosaure_est(dinosaure, heterodontosaurus).  
true .  
?- dinosaure_est(ornithischia, scelidosaurus).  
true .  
?- dinosaure_est(thyreophora, stegosaurus).  
true .  
?- dinosaure_est(eurypoda, ankylosaurus).  
true .  
?- dinosaure_est(genasauria, pachycephalosaurus).  
true .  
?- dinosaure_est(marginocephalia, triceratops).  
true .  
?- dinosaure_est(cerapoda, hypsilophodon).  
true .  
?- dinosaure_est(ornithopoda, iguanodon).  
true ■
```

Figure 9: Sortie du cas-test avec chaque lien au moins une fois en chaînage arrière de la question "dinosaure_est()".

%Tests du réseau de connexions (*connections non-permises*) *ch_arriere*

%Test pour 2 branches du même noeud

dinosaure_est(ornithopodesBasaux, iguanodon).

false.

%Test pour une branche différente d'un niveau de plus

dinosaure_est(ornithopoda, triceratops).

false.

%Test pour une branche différente de 2 niveaux de plus

dinosaure_est(cerapoda, stegosaurus).

false.

%Test pour une branche différente de 3 niveaux de plus

dinosaure_est(theropoda, diplodocus).

false.

%Test pour une branche différente de 4 niveaux de plus

dinosaure_est(ornithischia, spinosaurus).

false.

```
?- dinosaure_est(ornithopodesBasaux, iguanodon).
false.

?- dinosaure_est(ornithopoda, triceratops).
false.

?- dinosaure_est(cerapoda, stegosaurus).
false.

?- dinosaure_est(theropoda, diplodocus).
false.

?- dinosaure_est(ornithischia, spinosaurus).
false.
```

Figure 10: Sortie du test des connections non-permises avec le chaînage arrière de la fonction `dinosaure_est()`.

%Tests de fonction - `dinosaure_est(X,Y)`. (*ch_arriere*)

%X n'existe pas

dinosaure_est(ornithopoda, barney).

false.

%Y n'existe pas

dinosaure_est(indominusRex, iguanodon).

false.

%X et Y n'existe pas

dinosaure_est(indominusRex, lucy).

false.

```
?- dinosaure_est(ornithopoda, barney).
false.

?- dinosaure_est(indominusRex, iguanodon).
false.

?- dinosaure_est(indominusRex, lucy).
false.
```

Figure 11: Sortie du test de la fonction `"dinosaure_est()` en chaînage arrière.

%Tests de fonction - exemple_de(Classe(X)). (ch_avant)

%Valeurs incorrectes

exemple_de(ornithopoda).

false.

exemple_de(fait(ornithopoda(X))).

false.

%Une classe inconnue

exemple_de(indominusRex(X)).

false.

%Un dinosaure au lieu d'une classe

exemple_de(tyrannosaurus).

false.

```
?- exemple_de(ornithopoda).  
false.  
  
?- exemple_de(fait(ornithopoda(X))).  
false.  
  
?- exemple_de(indominusRex(X)).  
false.  
  
?- exemple_de(tyrannosaurus).  
false.
```

Figure 12: Sortie du cas-test : fonction "exemple_de()" en chaînage avant.

%Fait vrai, profondeur 0

exemple_de(herrerasauridae(X)).

X = herrerasaurus.

%Fait vrai, profondeur 1

exemple_de(marginocephalia(X)).

X = triceratops.

%Fait vrai, profondeur 2

exemple_de(thyreophora(X)).

X = euoplocephalus.

%Fait vrai, profondeur 3

exemple_de(theropoda(X)).

X = allosaurus.

%Fait vrai, profondeur 4

exemple_de(saurischia(X)).

X = brachiosaurus.

%Fait vrai, profondeur 5

exemple_de(dinosaure(X)).

X = iguanodon.

```
?- exemple_de(herrerasauridae(X)).
nouveau fait : ornithopoda(iguanodon)
nouveau fait : marginocephalia(triceratops)
nouveau fait : cerapoda(iguanodon)
nouveau fait : euryopoda(euoplocephalus)
nouveau fait : thyreophora(euoplocephalus)
nouveau fait : genasauria(iguanodon)
nouveau fait : ornithischia(iguanodon)
nouveau fait : dinosaure(iguanodon)
nouveau fait : neosauropoda(brachiosaurus)
nouveau fait : sauropoda(brachiosaurus)
nouveau fait : sauropodomorpha(brachiosaurus)
nouveau fait : tetanurae(allosaurus)
nouveau fait : neotheropoda(allosaurus)
nouveau fait : theropoda(allosaurus)
nouveau fait : saurischia(brachiosaurus)
Plus de nouveaux faits déduits, la BC est saturée.
X = herrerasaurus.

?- exemple_de(marginocephalia(X)).
Plus de nouveaux faits déduits, la BC est saturée.
X = triceratops.

?- exemple_de(thyreophora(X)).
Plus de nouveaux faits déduits, la BC est saturée.
X = euoplocephalus.

?- exemple_de(theropoda(X)).
Plus de nouveaux faits déduits, la BC est saturée.
X = allosaurus.

?- exemple_de(saurischia(X)).
Plus de nouveaux faits déduits, la BC est saturée.
X = brachiosaurus.

?- exemple_de(dinosaure(X)).
Plus de nouveaux faits déduits, la BC est saturée.
X = iguanodon.

?- ■
```

Figure 13: Sortie du cas-test : fonction `exemple_de()` en chaînage avant.

3.2 Résultats obtenues

Nous visions ici deux approches. Une fonction nous permettant de vérifier si un dinosaure appartient à une classe précise. Cette fonction est `dinosaure_est(Classe,Dinosaure)`. Cette fonction utilise le chaînage arrière pour donner une réponse. Voici un exemple de son utilisation.

Questions avec `dinosaure_est()`

```
?- dinosaure_est(genasauria,ankylosaurus).
true.

?- dinosaure_est(heterodontosauridae,ankylosaurus).
false.
```

Figure 14: Sortie de la fonction "dinosaure_est()".

Comme deuxième opération, nous désirions une fonction qui nous retourne un exemple de dinosaure d'une classe donnée. Cette fonction est `exemple_de(Classe)`. Cette fonction utilise le chaînage avant pour nous donner le premier dinosaure correspondant à cette catégorie. Voici un exemple de son utilisation.

Questions avec `exemple_de()`

```
?- exemple_de(herrerasauridae(X)).
nouveau fait : ornithopoda(iguanodon)
nouveau fait : marginocephalia(triceratops)
nouveau fait : cerapoda(iguanodon)
nouveau fait : thyreophora(euoplocephalus)
nouveau fait : genasauria(iguanodon)
nouveau fait : ornithischia(iguanodon)
nouveau fait : neosauropoda(brachiosaurus)
nouveau fait : sauropoda(brachiosaurus)
nouveau fait : sauropodomorpha(brachiosaurus)
nouveau fait : tetanurae(allosaurus)
nouveau fait : neotherapoda(allosaurus)
nouveau fait : therapoda(allosaurus)
nouveau fait : saurischia(brachiosaurus)
Plus de nouveaux faits d'aduits, la BC est saturée.
X = herrerasaurus
```

Figure 15: Sortie de la fonction "exemple_de()".

3.3 Discussion sur les résultats

Ce système à base de connaissances a été conçu de façon à agir comme un outil de révision des acquis pour un cours (comme paléontologie) que comme un outils donnant les réponses directement.

Notre système a bien répondu aux attentes. La base de connaissances nous permet de vérifier si un dinosaure appartient à une classe choisie et on peut également avoir un exemple de dinosaure appartenant à une classe. Aucune réponse ne sera donnée directement à l'étudiant.

Il est clair cependant que notre système, dans son état actuel, ne peut pas servir à accélérer un processus de classement où d'autres tâches du domaine. Cependant, la base de connaissances est fonctionnelle et il suffirait de rajouter les fonctions appropriées pour répondre à de nouvelles exigences.

Pour amélioration, les possibilités sont nombreuses. Voici un exemple.

Un système de questions simples comme: "Ce dinosaure a-t-il des pieds d'oiseau?" nous aurait permis directement de remonter à la classe ornithopoda. En fait, chacune de ces classes sont toutes décomposables en parties qui nous indique la caractéristique permettant de dire si oui ou non un dinosaure appartient à cette classe. Par exemple, "ornithopoda" se décompose en "ornitho" qui veut dire oiseau et "poda" qui veut dire pieds. Donc cette catégorie de dinosaure a des pattes ressemblant à celles d'un oiseau.

Ce système aurait pu se brancher par-dessus notre réseau existant et donnerait un système beaucoup plus accessible à un public moins spécialisé.

4. Bilan de l'expérimentation

4.1 Avantage d'utiliser un SBC

Plusieurs avantages justifient l'utilisation d'un système à base de connaissances.

La première entraîne littéralement le deuxième nom donné à ce type de système: Systèmes Experts. Dans ce type de système on utilise directement les connaissances que l'expert a pu accumulée tout au long de sa carrière et on les utilise directement dans la base de connaissance. Le système sera alors capable de répondre à une grande partie des réponses que notre expert peut répondre.

Dans un tel système, on sépare les connaissances et le raisonnement en 2 parties distinctes. Il nous est alors plus facile d'intégrer de nouvelles connaissances sans avoir à modifier la partie raisonnement et vice-versa.

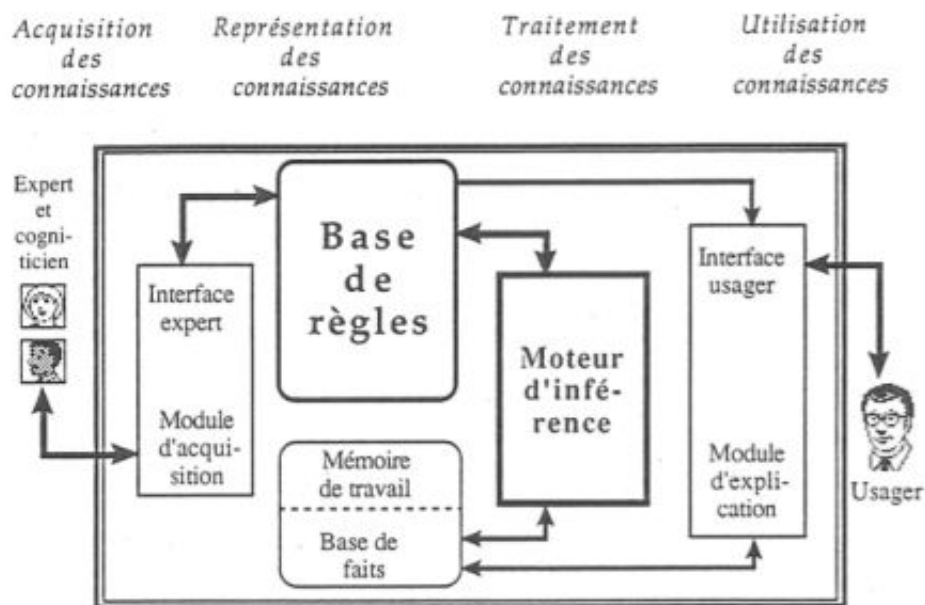


Figure 16 : Architecture d'un système à base de règles [3]

L'implantation d'un SBC, surtout par l'intermédiaire d'une coquille, est facile et rapide. Le système peut également continuer à ajouter des connaissances sans l'apport du programmeur. C'est une solution rapide pour rassembler toutes les connaissances des experts sans avoir besoin d'un support lourd côté informatique.

Il est également facile de recréer le chemin pour atteindre une réponse et de s'en servir pour donner justifier une réponse et donner des explications supplémentaires.

4.2 Difficulté pour déterminer l'expertise

L'identification scientifique internationale devient rapidement un casse tête lorsque l'on parle biologie des espèce. Comme chaque langue donne un nom différent à chaque créature, il a fallu trouver une solution au problème. Les sources latines au grecque fut la solution adopté par le domaine de la biologie.

Cependant cela entraîne un autre problème car la connaissances des racines grecques n'est pas une connaissance commune dans la population. Pour la majorité d'entre nous, le mot "Sauripoda" ne veut pas dire "pieds de lézard". Il a autant de sens qu'un livre pour un chat.

Dans ce domaine, la classification s'est développée au fil du temps car il est difficile d'avoir un consensus de tous les experts. Un nouveau fossile découvert peut soudainement forcer une reclassification de tous les spécimens. Notre système deviendrait alors obsolète.

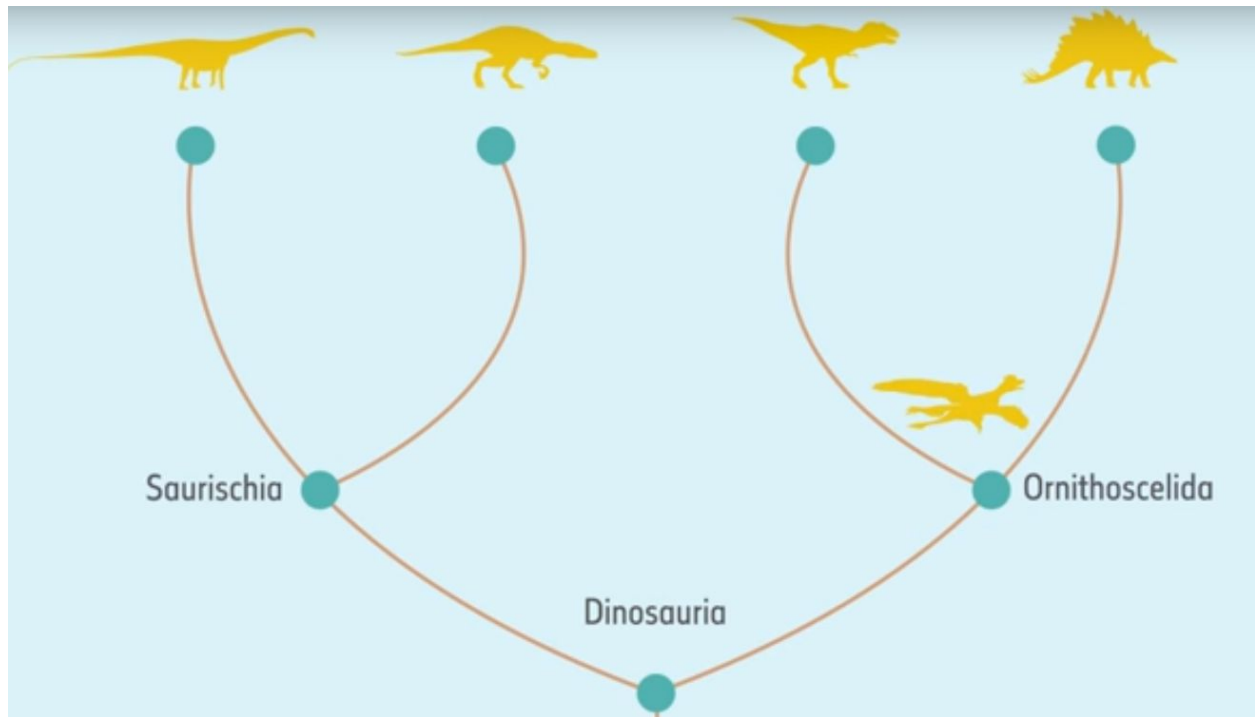


Figure 17: Image représentant un nouvel arbre possible de classification des dinosaures avec les oiseaux dans la catégorie "ornithoscelida" et non celle des "saurischia" [4].

Nous avons choisi d'utiliser le code de nomenclature zoologique international pour son côté reconnu et simple d'utilisation. Plusieurs autres classements furent utilisés aux cours des années.

Notre référence: <https://fr.wikipedia.org/wiki/Dinosaure#Classification>

Cette classification, bien que plus simple que plusieurs autres, se base sur des racines grecques et latines ce qui peut rendre les noms particulièrement complexes pour quelqu'un qui n'a pas l'habitude à ce genre de vocabulaire.

4.3 Avantages et contraintes de la coquille

Le principal avantage de l'utilisation d'une coquille est de permettre de construire un système à base de connaissance plus rapidement. La coquille sert de squelette au système. Il ne suffit que de rajouter les connaissances (faits) et les règles qui les gèrent pour avoir un système fonctionnel.

De plus, les experts du domaine sur lequel sera basé le SBC n'est pas nécessairement un programmeur et l'utilisation d'une coquille permet au programmeur de travailler sur la coquille alors que l'expert s'occupe de rentrer les connaissances dans le système.

Cependant, les limites d'une coquille s'avère souvent être les propre limite du programmeur qui l'utilise. Comme une coquille est conçue souvent à l'externe. La solution semble instantanée, mais si la coquille est complexe, le programmeur tentant de la modifier pourra passer un temps considérable à adapter cette dernière au besoin de l'entreprise.

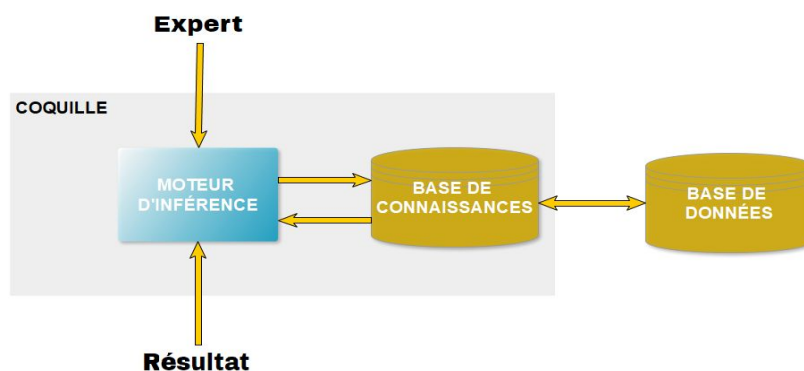


Figure 18: Image représentant le fonctionnement d'une coquille[5].

Les coquilles ne sont pas nécessairement adaptées au besoin du domaine non plus. Dans notre cas, il s'agit d'un système de classification simple, mais pour un système compliqué avec beaucoup de règles sur les connaissances, il est possible que la coquille ne puisse pas fournir les outils nécessaire à la construction d'un système adéquat.

L'expert n'est pas toujours une ressource facile à trouver non plus et sans son apport, le système expert a très peu de valeur.

4.4 Avantages et limites du système

La principale limite de notre système est l'utilisation des noms compliqués pour les connaissances. Ceci réduit considérablement le public cible ayant la capacité de se servir du système.

Notre système est simple, mais il manque de raffinement. Avec une base de connaissance de ce type, plusieurs autres utilisations auraient pu être faites. Sortir une liste de choix potentiel à une question, sortir une hiérarchie de classe. Pour le moment notre système ne sort qu'une seule réponse à la question et toujours la même. Si jamais plus d'un dinosaure convient à la question, ça sera toujours le même qui reviendra avec `exemple_de`.

L'avantage principal est la simplicité du système. Deux questions seulement et une réponse claire. `"dinsaure_est"` retourne `"true"` ou `"false"`. `"exemple_de"` retourne un dinosaure si la catégorie existe, sinon rien. Aucune confusion dans les réponses.

Un autre avantage est la possibilité de faire une préparation d'examen pour un étudiant sans avoir la hiérarchie visible en tout temps pour vérifier la réponse. Il est facile de s'imaginer qu'on a bien retenue une matière avec le diagramme en main, mais rendu à l'examen sans ce même diagramme, la chose sera peut-être différente.

4.5 Amélioration à apporter au système

Nous aurions voulu améliorer notre système en ajoutant une couche plus simple que les noms avec racines grecques ou latines. Pouvoir poser des questions sur le dinosaure aurait permis d'éviter l'usage des racines grecques et latine et aurait rendu le programme beaucoup plus facile d'utilisation pour un public moins spécialisé.

Il aurait été aussi bien de pouvoir afficher une hiérarchie de classe pour un dinosaure. De pouvoir demander des caractéristiques sur les classes ou de rechercher une classe et/ou dinosaure par ses caractéristiques.

5. Bibliographie

- [1] Image de Wikipedia, [Classification des dinosaures](#)
- [2] Inspiré de l'Image du Blog [vers le KLM en ingénierie système](#)
- [3] Site Teluq, image [Architecture d'un système à base de règles](#)
- [4] Image tirée de la vidéo [Dinosaur family tree gets major makeover de la chaîne Youtube The Natural History Museum](#)
- [5] Inspiré de l'Image du site ResearchGate, [Working process of KBS](#)

6. Annexes

ANNEXE 2 – COQUILLE DE SBC EN LANGAGE PROLOG

```
% Definition des operateurs
:- op( 800, fx, si ),
   op( 700, xfx, alors ),
   op( 300, xfy, ou ),
   op( 200, xfy, et ).
:- dynamic(fait/1).

% données du problème : fait( X ) - à ajouter

% Règles de la base de connaissances : si ... alors ... - à ajouter

% ch_arriere/1 : moteur d inference fonctionnant en chainage
arriere ch_arriere( But ) :- est_vrai( But ).
est_vrai( Proposition ) :- fait( Proposition ).
est_vrai( Proposition ) :- si Condition alors Proposition, est_vrai( Condition ).
est_vrai( Cond1 et Cond2 ) :- est_vrai( Cond1 ), est_vrai( Cond2 ).
est_vrai( Cond1 ou Cond2 ) :- est_vrai( Cond1 ) ; est_vrai( Cond2 ).

% ch_avant/0 : moteur d inference fonctionnant en chainage avant
ch_avant :-
    nouveau_fait( Nouveau ), !, write( 'Nouveau fait : ' ), write( Nouveau ), nl, assert( fait( Nouveau ) ), ch_avant.
ch_avant :-
    write( 'Plus de nouveaux faits déduits, la BC est saturée.' ), nl.
nouveau_fait( NouvFait ) :-
    si Condition alors NouvFait, not( fait(NouvFait) ), recherche_fait( Condition ).
recherche_fait( Condition ) :-
    fait( Condition ).
recherche_fait( Cond1 et Cond2 ) :-
    recherche_fait( Cond1 ), recherche_fait( Cond2 ).
recherche_fait( Cond1 ou Cond2 ) :-
    recherche_fait( Cond1 ) ; recherche_fait( Cond2 ).
```