## 3. Sémantique *(VIDEO-PPT 2.2)*

### 3.1. Chainages déductifs

Problème : trouver une méthode automatique de résolution donnant une réponse à une question.

Différentes stratégies peuvent être développées pour répondre à une question.

**Chainage avant :** on part des faits et on applique les règles pour déduire des conclusions (mécanisme de déduction) :

faits => …. => conclusions.

- Un médecin constate : fièvre + boutons => … => rougeole ou roséole ou scarlatine

**Chainage arrière :** on part d'une hypothèse (conclusion) et on « remonte » les règles pour trouver des faits qui confirment ou infirment l'hypothèse (mécanisme d'abduction) :

hypothèse <= …. <= faits.

- Un médecin émet l'hypothèse : supposons roséole <= …<= quel âge a le patient ?

- En pratique, le médecin procède à la fois en chainage avant (partant des symptômes, il en conclut les maladies possibles) puis en chainage arrière (partant d'une maladie possible, il interroge le patient ou fait des examens pour obtenir des données complémentaires (faits) infirmant ou confirmant la maladie).

Le système *Prolog* se base sur un mécanisme de chainage arrière (on part d'une question on remonte vers les faits).

### 3.2. Sémantique opérationnelle (SLD-résolution)

La SLD-résolution (SLD signifie Sélection Linéaire Définie) est un algorithme évaluant les buts à effacer en chaînage arrière.

- Reprenons l'exemple de la section précédente réduit à un sous-ensemble de clauses :

```
1. grand_mere(X,Y) :- mere(X,Z) , pere(Z,Y).
2. mere(rose, john).
3. mere(rose, robert).
4. pere(john, caroline).
```

Poser la question `grand_mere(rose, caroline)` avec une interprétation procédurale revient à effacer le but `grand_mere(rose, caroline).`

Pour cela, on cherche dans la base de connaissances une règle qui peut être appelée en fonction du but à résoudre :

- `1` et `{X = rose, Y = caroline }` exprime que résoudre `grand_mere(rose, caroline)` revient à résoudre la conjonction de buts `mere(rose,Z), pere(Z, caroline)`. Cette conjonction de buts est appelée **résolvante**.

L'algorithme SLD-résolution évalue la conjonction de buts `mere(rose,Z), pere(Z, caroline)` en séquence de la gauche vers la droite.

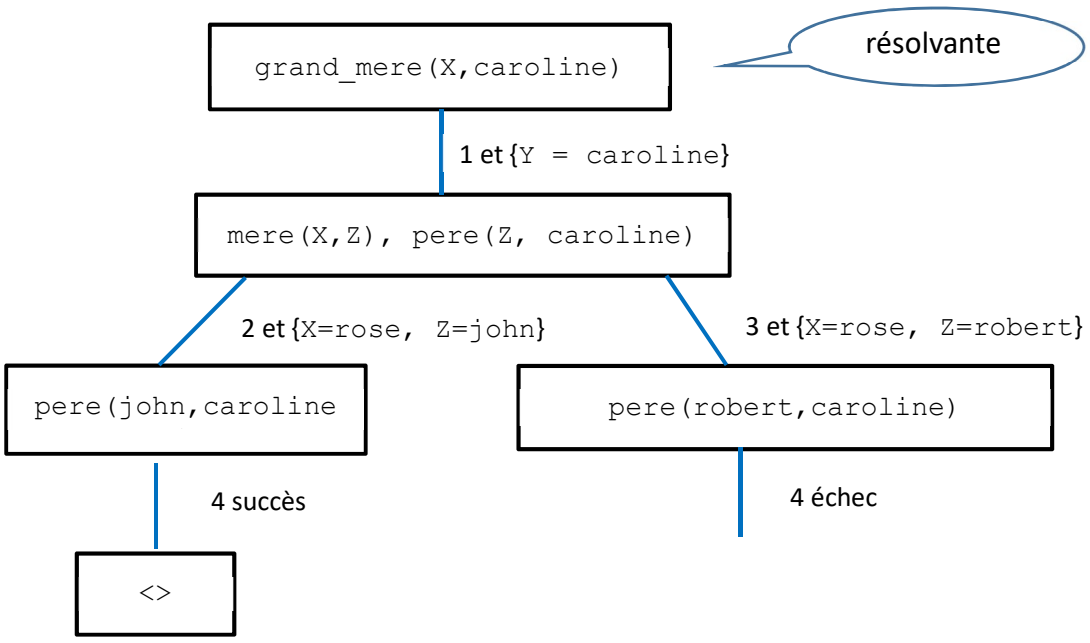
*NB* : cette évaluation gauche-droite est une spécificité par rapport à la logique mathématique où cette notion de séquencement n'a pas de sens (les buts pourraient être traités dans un ordre différent).

*Prolog* cherche donc d'abord à effacer : `mere(rose, Z)` :

- La clause 2 + `{Z = john}` efface ce but. La résolvante se réduit à `pere(john, caroline)` qui se résout par 2 puis 3 :
  - 2 ne s'unifie pas à ce but et donc ne l'efface pas : c'est un **échec**.
  - 3 ne s'unifie pas à ce but et donc ne l'efface pas : c'est un **échec**.
  - 4 efface ce but. La résolvante est vide : c'est un **succès**.
- La clause 3 + `{Z = robert}` efface ce but. La résolvante se réduit à `pere(robert, caroline)` qui ne s'unifie sur aucune clause, c'est un **échec**
- La clause 4 ne s'unifie pas à `mere(rose, Z)`, c'est un **échec**.

**Arbre de résolution (de recherche)**

Posons la question ouverte `grand_mere(X,caroline)`.




Pour résoudre une question, Prolog construit l'arbre de recherche issu de la résolvante formée de la question. L'arbre de recherche est parcouru en profondeur d'abord et de gauche à droite. Les nœuds terminaux sont des nœuds de succès ou d'échec :

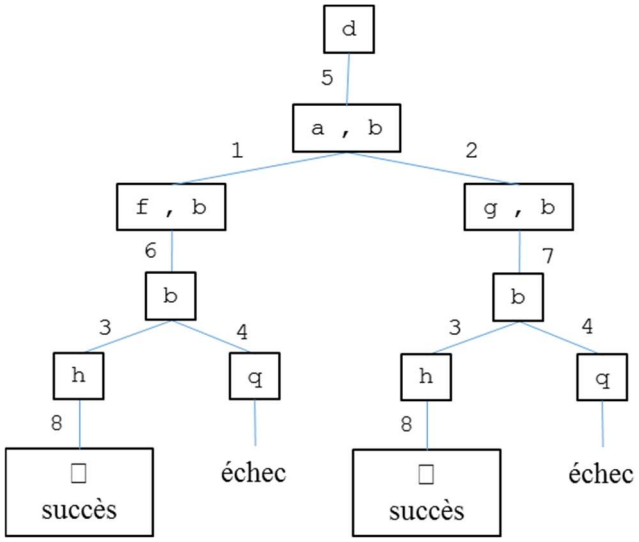
- nœud de succès : la résolvante est vide. Prolog est arrivé à une solution, il l'affiche et cherche d'autres solutions en remontant dans l'arbre jusqu'à un point de choix possédant des branches non explorées (retour arrière) ;
- nœud d'échec (notée □) : remontée dans l'arbre jusqu'à un point de choix possédant des branches non explorées.

➢ Autre exemple (synthétique), base de connaissances :

```
1. a :- f.
2. a :- g.
3. b :- h.
4. b :- q.
5. d :- a, b.
6. f.
7. g.
8. h.
```

Question : `|?- d.`

Arbre de recherche :




➢ *miniTP* : Soit le prédicat récursif ci-dessous qui définit le concept d'ancêtre :

```
1. ancetre(X, Y) :- mere(X, Y).
2. ancetre(X, Y) :- pere(X, Y).
3. ancetre(X, Y) :- mere(X, Z) , ancetre(Z, Y).
4. ancetre(X, Y) :- pere(X, Z) , ancetre(Z, Y).
```

a) Décrivez l'arbre de résolution correspondant à la question `ancetre(X, caroline).`
b) Pourquoi l'ordre des littéraux dans la queue de clause 4 est ici important ?

On rappelle les faits disponibles :

```
5. mere(rose, john).
6. mere(rose, robert).
7. pere(john, caroline).
```