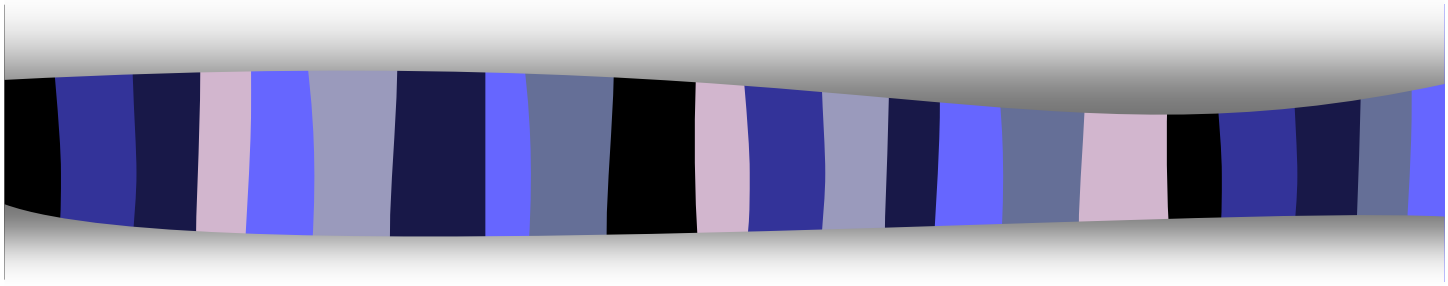
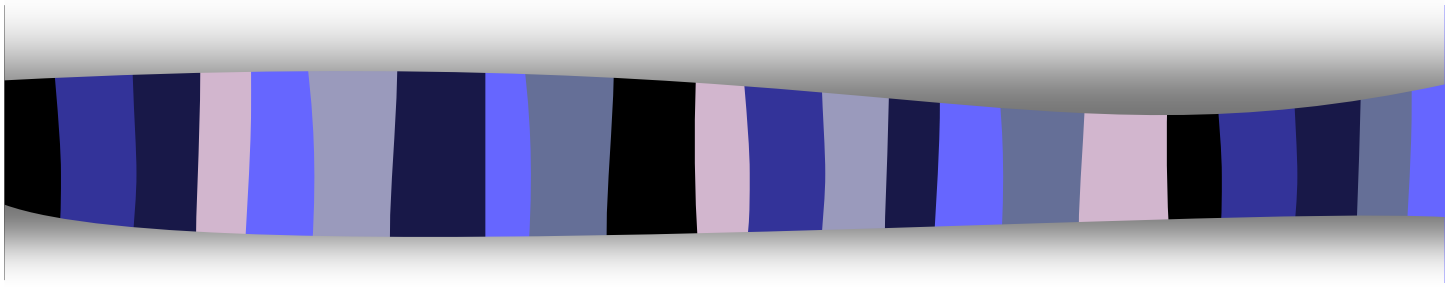


Introduction à la programmation logique



Plan

- **Programmes Prolog**
  - Clauses de Horn
  - Programmes et but définis
  - Représentation d'un problème
- **Stratégie des moteurs Prolog**
  - Principes
- **Bilan sur l'approche logique**

Notion de programme en logique

Clauses de Horn

Définitions

Une clause de *Horn* contient au plus un littéral positif : $\neg H_1 \vee \ldots \vee \neg H_n \vee C$, avec $n \geq 0$

- Alternative : $\forall x_1 \ldots \forall x_p \, (H_1 \wedge \ldots \wedge H_n \Rightarrow C)$, où $n \geq 0$, et $x_1 \ldots x_p$ variables libres de l'implication
- Sémantique : « si $H_1,..,H_n$ sont vrais, alors C est vrai »
- Une clause sans littéral positif est *négative*

Propriétés

Tout ensemble de clauses de Horn non négatives est satisfiable

Les stratégies de type linéaire avec entrées sont complètes pour les clauses de Horn

Programme Prolog

Programmes définis

Un *programme défini* est un ensemble fini de clauses de Horn :

$$\neg H_1 \vee \ldots \vee \neg H_n \vee C_1$$
$$\ldots\ldots\ldots\ldots$$
$$\neg T_1 \vee \ldots \vee \neg T_p \vee C_q$$

☞ Notation en programmation logique :

$C \leftarrow H_1, \ldots, H_n$ *règles*

tête *corps*

$C \leftarrow$ *faits*

Programme Prolog

But défini (requête, question)

Une question est une fbf *G* de la forme :

$$\exists x_1 \ldots \exists x_s \ (G_1 \wedge \ldots \wedge G_r)$$

La négation de ce *but G*, une fois ajoutée aux clauses du programme, est une clause de Horn négative $\neg G_1 \vee \neg G_2 \vee \ldots\ldots\ldots\ldots \vee \neg G_r$

Notation en programmation logique :

$$\leftarrow G_1 , \ldots , G_r$$

La méthode de réfutation par résolution fournit une réponse à la question sous la forme d'une substitution

Programme Prolog

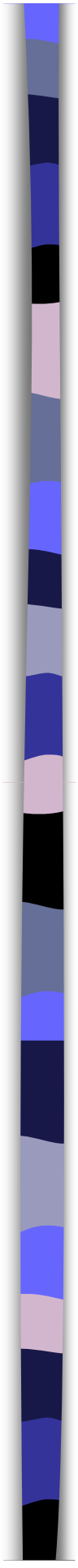
Expression Prolog d'un problème

Cette représentation comprend donc des règles, des faits et une question, qui résulte de l'ajout de la négation du but aux clauses du programme défini

$$\neg H_1 \vee \ldots \vee \neg H_n \vee C_1$$
$$\ldots\ldots\ldots\ldots$$
$$\neg T_1 \vee \ldots \vee \neg T_p \vee C_q$$
$$\neg G_1 \vee \ldots \vee \neg G_r$$

Limites de la représentation :

- Toutes les variables sont quantifiées universellement
- Pas de négations dans le corps des clauses, donc pas de disjonction (astuces)



Stratégie Prolog

Stratégie Prolog

La stratégie des divers langages Prolog est appelée *SLD Résolution*, pour "Linear Definite with Selection function Resolution"

Linéaire : la clause C_0 est issue de $\neg G$ et l'un des parents est le dernier résolvant produit (au cas où c'est possible)

A entrées : l'une des clauses parentes est choisie parmi les clauses du programme défini

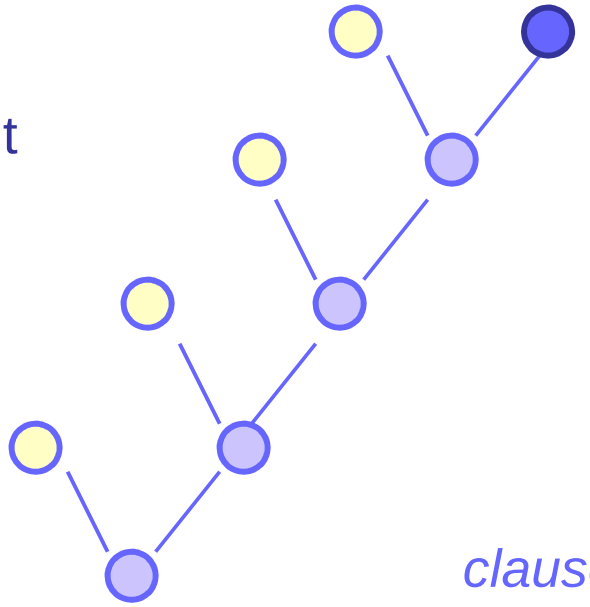
Ordonnée : un ordre existe sur le choix des clauses ainsi que sur celui des littéraux

La stratégie SLD-résolution est complète pour l'expression Prolog de tout problème

Stratégie Prolog

En profondeur d'abord

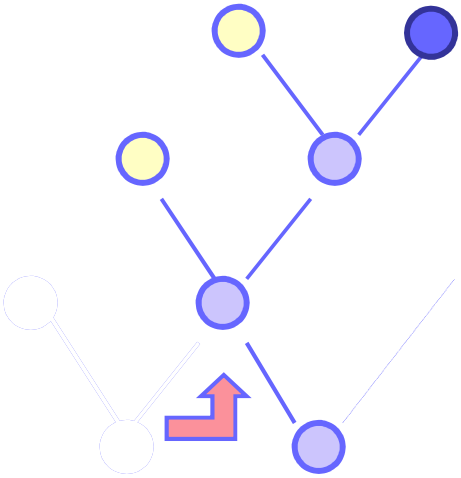
A partir du dernier
résolvant, évidemment
si c'est possible



clause
but

Bactracking

Que se passe-t'il si
le développement d'une
branche conduit
à une impasse ?



Stratégie Prolog

Choix de la clause d'entrée

Les clauses d'entrée étant considérées ordonnées, la clause d'entrée choisie est toujours la 1ère clause trouvée en état de l'être

Question : que se passe-t'il si une branche infinie est développée ? Il y a échec : aucun arbre de réfutation ne peut être trouvé, alors que pourtant il en existe au moins un !!!

Raison : c'est la combinaison de la recherche en profondeur d'abord avec la règle de choix de la première clause qui fait que la stratégie de Prolog n'est pas directement complète

Que faire ? Bien ordonner les clauses du programme. De "l'Art de la programmation logique" !

Stratégie Prolog

Choix du littéral

Le littéral choisi dans la clause issue de la question est toujours le littéral négatif le plus à gauche

clause d'entrée choisie

littéral le plus à gauche

$\{ \neg H_1 \vee ..... \vee \neg H_n \vee C \}$

$\{ \neg G_1 \vee \neg G_2 \vee ..... \vee \neg G_s \}$

$\sigma(C) = \sigma(G_1)$

$\{ \neg\sigma(H_1) \vee .... \vee \neg\sigma(H_n) \vee \neg\sigma(G_2) \vee ..... \vee \neg\sigma(G_s) \}$

nouveau littéral à considérer

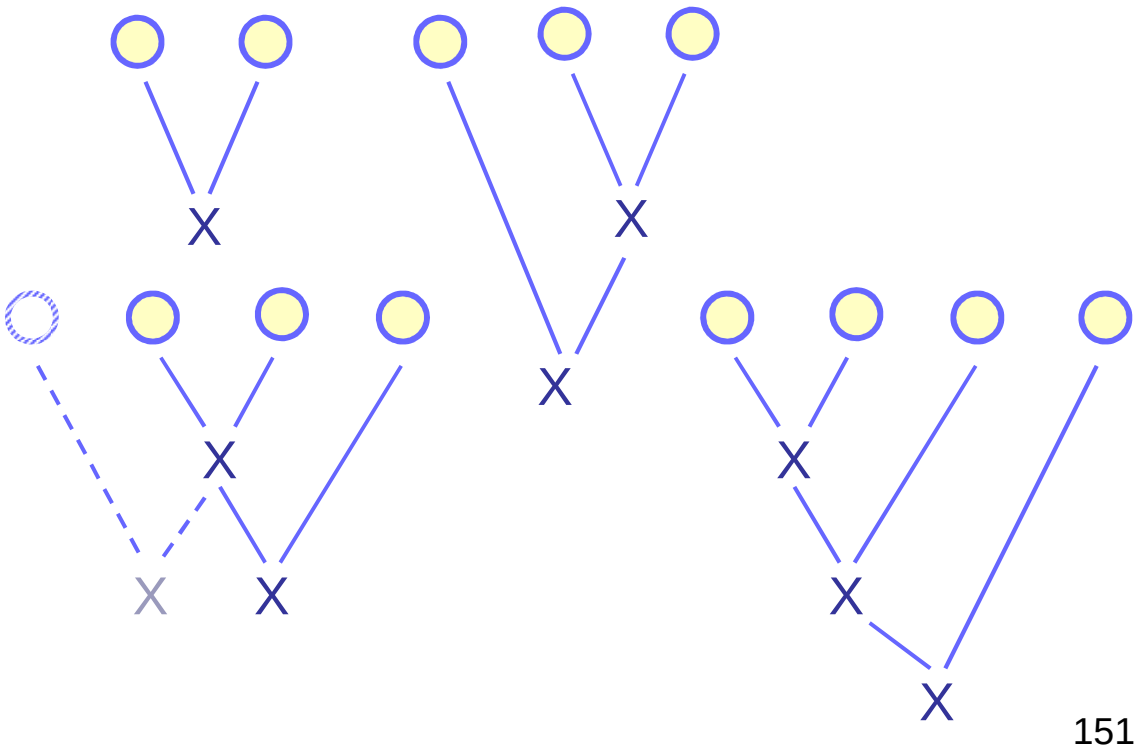
nouvelle clause-but

Stratégie Prolog

Recherche exhaustive

Un Prolog “pur” (sans le cut) recherche toutes les réponses possibles

- Interprétation procédurale de la recherche "en-arrière" : décomposition de but en sous-buts jusqu'à l'obtention du but vide
- **Un arbre Prolog** :



Approche logique

Avantages

Liés à la logique des prédicats

- représentation puissante et concise
- outils de vérification du raisonnement
- cadre théorique de référence

Liés à la règle de résolution

- uniformisation des connaissances
- solution non spécifiée à l'avance
- réalisations informatiques

Limites

- risque d'inefficacité
- approche réservée à certains problèmes
- représentation abstraite
- pas de contrôle du raisonnement
- perte d’information heuristique