De la logique à la programmation logique

Cours « Découverte de l'intelligence artificielle »

Nicolas Delestre

INSA
ROUEN NORMANDIE

Plan

INSA ROUEN NORMANDIE

Exemples de résultats intriguants

- Trompette de Gabriel : trompette fondée sur une hyperbole qui a un volume fini, mais une surface infinie
- La diagonale biscornue : $\sqrt{2} = 2$?
- Fonction zêta de Riemann ($\zeta(s) = \sum_{i=0}^{+\infty} \frac{1}{i^s}$) avec $s = -1$: $\sum_{i=0}^{+\infty} i = -\frac{1}{12}$?
- Paradoxe du barbier
- Quadrature du cercle

Science4all : Top 8 des monstres mathématiques

`https://youtu.be/f13ZaZrfIUc`

INSA ROUEN NORMANDIE

Définitions 1 / 2

Système formel

≪ modélisation mathématique d'un langage en général spécialisé [...] Les éléments linguistiques, mots, phrases, discours, etc., sont représentés par des objets finis [...] Le propre d'un système formel est que la correction au sens grammatical de ses éléments est vérifiable algorithmiquement, c'est-à-dire que ceux-ci forment un ensemble récursif. ≫ (wikipédia, 2021)

Logique mathématique

≪ discipline des mathématiques introduite à la fin du XIXe siècle, qui s'est donné comme objet l'étude des mathématiques en tant que langage. Les objets fondamentaux de la logique mathématique sont les formules représentant les énoncés mathématiques [...] et les sémantiques qui donnent un "sens" mathématique générique aux formules. ≫ (wikipédia, 2021)

Cf. Science4All, "1+1=2" : `https://youtu.be/oKprCgIKWxo`

Définitions 2 / 2

Logique informatique

≪ [domaine de l'informatique qui] étudie l'automatisation des calculs et des démonstrations, les fondements théoriques de la conception des systèmes, la programmation et l'intelligence artificielle. L'approche informatique est aujourd'hui cruciale car, en essayant de mécaniser les raisonnements, voire de les automatiser, la logique et les mathématiques vivent une véritable révolution depuis la fin du XXe siècle. ≫ (wikipédia, 2021)

Domaines scientifiques connexes

- Calculabilité : quels problèmes peuvent être résolus par un algorithme ?
- Complexité : combien de temps, d'espace, sont-ils nécessaires pour résoudre un problème ?

INSA ROUEN NORMANDIE

Les logiques 1 / 3

Les trois composants

- Formule : phrase du langage de la logique, suites de symboles bien formées qui respectent syntaxiquement des règles de construction
- Interprétation : fonction qui associe à toute formule un objet dans un monde (nommé modèle) qui permet de définir la validité des formules
- Déduction : algorithme qui permet de créer de nouvelles formules (théorème, conclusion) à partir de formules connues (axiomes, prémisses) et de règles d'inférence :

$$\frac{A_1 \dots A_n}{B} \qquad A_1 \dots A_n \vdash B$$

Les axiomes sont les lois logiques, notés $\vdash A$

INSA ROUEN NORMANDIE

Les logiques 2 / 3

Plusieurs logiques

- Elles se distinguent par les éléments de base pour écrire des formules et par les systèmes de déduction
- Plus une logique est ≪ riche ≫, plus son pouvoir de représentation est grand, mais plus les algorithmes de déduction sont complexes (au sens de la complexité algorithmique)

Exemples de logique

- Logique des propositions
- Logique des prédicats du premier ordre : extension de la logique des propositions avec des variables, des fonctions, des prédicats et deux quantificateurs
- Logique d'ordre supérieur : extension de la logique des prédicats telles que les variables peuvent référencer des fonctions et des prédicats
- Logiques modales : contextualisations des formules, par exemple
  - temporelles : *demain*, il pleut
  - épistémiques : *Paul croit* qu'il pleut

Les logiques 3 / 3

Restreindre une logique

- Pour améliorer les performances des algorithmes de déduction : par exemple les clauses de Horn (toutes les formules sont de la formes $(\neg r_1 \vee \neg r_2 \cdots \vee \neg r_n) \vee h$)
- Pour mieux s'adapter au domaine à modéliser : par exemple les logiques de description pour représenter des connaissances

INSA ROUEN NORMANDIE

La logique des propositions

- Logique la plus simple, permet de représenter formellement ≪ S'il pleut ou il neige alors il y a des nuages ≫
- Constituants du langage :
  - variables propositionnelles
  - opérateurs : $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$
  - parenthèses pour enlever des ambiguïtés
- Formules propositionnelles (ou propositions) :
  - une variable est une proposition
  - Si p et q sont des propositions alors $\neg p$, $p \wedge q$, $p \vee q$, $p \rightarrow q$,$p \leftrightarrow q$ et (p) sont des propositions

Si A, B sont des propositions :

- $(A \rightarrow B) \rightarrow (\neg B \rightarrow \neg A)$ est une proposition
- $A \wedge B \wedge$ n'est pas une proposition

Interprétation ou modèle

Principe

- Associer une valeur de vérité (Vrai ou Faux) à chacune des variables propositionnelles
- Utiliser les tables de vérités de bases associées à chaque opérateur

Vocabulaire

- Une formule est *satisfaisable* s'il existe un modèle qui la rend vraie
- Une formule est une *tautologie* si tous les modèles la rendent vraie, notée $\models A$. A est alors appelé théorème
- Une formule est *falsifiable* s'il existe une modèle qui la rend fausse
- Une formule est une *antilogie* si tous les modèles la rendent fausse

INSA ROUEN NORMANDIE

Comment démontrer qu'une formule est un théorème ?

Première méthode : calculer toutes les interprétations = sa table de vérité

- Avantage : méthode décidable
- Inconvénient : très inefficace, si n propositions 2^n calculs

Deuxième méthode : démontrer la formule

- Théorème d'adéquation : Si $A \vdash B$ alors $A \vDash B$
- Théorème de complétude : Si $A \vDash B$ alors $A \vdash B$
- Quatre méthodes :
  - **Frege et Hilbert : une seule règle, le Modus ponens, des axiomes**
  - Déduction naturelle : un seul axiome, plusieurs règles
  - Par résolution : généralisation du Modus ponens, transformation de la formule en forme normale conjonctive (conjonctions de disjonctions, de clauses) afin de produire une disjonction
  - **Méthodes des tableaux : construction d'un arbre (un créant deux fils si $\vee$) à partir de la négation de la formule en essayant de fermer chaque branche (une branche est fermée lorsqu'il y a une contradiction)**

ROUEN NORMANDIE

Fredge et Hilbert

Modus ponens

$$\frac{\vdash A \quad \vdash (A \to B)}{\vdash B}$$

$\vdash A$ indique que A est un axiome ou un théorème

Axiomes [GRT02]

1. $\vdash A \to (B \to A)$
2. $\vdash (A \to (A \to B)) \to (A \to B)$
3. $\vdash (A \to (B \to C)) \to ((A \to B) \to (A \to C))$
4. $\vdash (A \leftrightarrow B) \to (A \to B)$
5. $\vdash (A \leftrightarrow B) \to (B \to A)$
6. $\vdash (\neg B \to \neg A) \to (A \to B)$
7. $\vdash A \vee B \leftrightarrow (\neg A \to B)$
8. $\vdash A \wedge B \leftrightarrow \neg(A \to \neg B)$

Exemple

Les deux premiers axiomes

1. $\vdash A \to (B \to A)$
2. $\vdash (A \to (A \to B)) \to (A \to B)$

Démonstration que $A \to A$ est un théorème [GRT02]

- Axiome 1 : $\vdash A \to (A \to A)$
- Axiome 2 : $\vdash (A \to (A \to A)) \to (A \to A)$
- Modus ponens :

$$\frac{\vdash A \to (A \to A) \quad \vdash (A \to (A \to A)) \to (A \to A)}{\vdash A \to A}$$

INSA ROUEN NORMANDIE

Méthode des tableaux

Principe

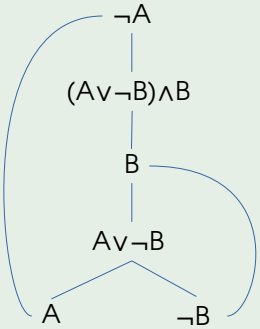
- Représenter la négation de la formule à démontrer à l'aide d'un arbre et essayer $\ll$ de fermer $\gg$ chaque branche de l'arbre, c'est-à-dire trouve une contradiction pour chaque branche
- Algorithme (simplifié) :
  - Représenter la formule uniquement avec les opérateurs $\neg$, $\wedge$ et $\vee$
  - Si la formule courante est l'opération binaire :
    - $\wedge$ les deux opérandes apparaissent dans la même branche (règle α)
    - $\vee$ la branche courante est décomposée en deux branches, chacune avec une opérande (règle β)

INSA ROUEN NORMANDIE

Exemple

Démontrer que $((A \vee \neg B) \wedge B) \rightarrow A$ est un théorème (Wikipédia)

- $((A \vee \neg B) \wedge B) \rightarrow A$ peut s'écrire $((\neg A \wedge B) \vee \neg B) \vee A$
- Sa négation est donc $((A \vee \neg B) \wedge B) \wedge \neg A$



Limites de la logique des propositions

- Les propositions sont indépendantes les unes des autres, même si dans une interprétation elles caractérisent un même individu
- Les propositions ne peuvent pas caractériser un ensemble d'individus

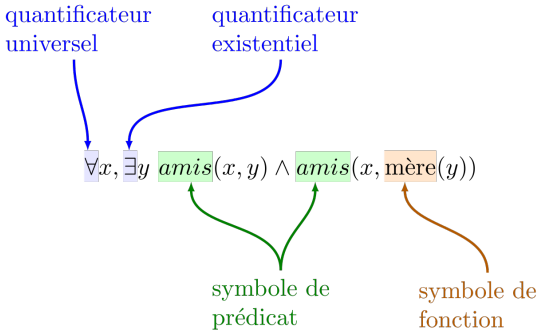
- Tous les hommes sont mortels
- Socrate est un homme
- Socrate est mortel

INSA ROUEN NORMANDIE

La logique des prédicats

- Logique mathématique qui remplace les variables propositionnelles par des prédicats et permet l'utilisation de deux quantificateurs : ≪ tous les hommes sont mortels ≫
- Constituants du langage :
  - constantes : désignent certains éléments du monde
  - fonctions : relient les éléments du monde
  - variables : représentent les éléments du monde sans les désigner. Une variable peut être liée ou libre
  - prédicats : caractérisent les éléments du monde
  - quantificateurs : $\forall$ et $\exists$
  - opérateurs : $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$
  - parenthèses pour enlever des ambiguïtés
- Formules :
  - $p(t_1, t_2, \ldots, t_n)$ où p est un prédicat, et t_i des termes (constante, fonction, variable) est une formule
  - Si e et f sont des formules alors $\neg e$, $e \wedge f$, $e \vee f$, $e \rightarrow f$,$e \leftrightarrow f$, (e), $\forall x\ e$, $\exists x\ e$ sont des formules

Exemples



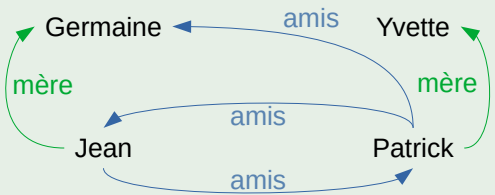
Source wikipédia (2021)

- $\forall x\ homme(x) \rightarrow mortel(x)$
- $homme(Socrate)$
- $mortel(Socrate)$

Interprétation 1 / 2

- Ensemble d'individus caractérisés par les prédicats (faits) et possiblement reliés par les fonctions
- Associer les constantes et variables aux individus
- Comme pour la logique des propositions, une formule peut être **satisfaisable**, **falsifiable**, une tautologie, une antologie

$\forall x, \exists y \ amis(x, y) \wedge amis(x, mere(y))$



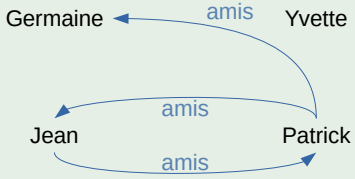
Qu'en est il pour :

- $\exists x, \forall y \ amis(x, y) \wedge amis(x, mere(y))$
- $\exists x, \exists y \ amis(x, y) \wedge amis(x, mere(y))$

Interprétation 2 / 2

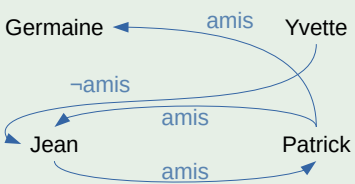
Monde fermé

Tout fait non déclaré est considéré comme Faux



Monde ouvert

Il faut déclarer un fait Faux pour qu'il le soit



Déduction

Plusieurs algorithmes

- Méthodes des tableaux
- Chaînage avant ou arrière

Utilisation en informatique

- Pour une interprétation donnée (faits), trouver les individus telle qu'une formule (question) soit vrai

INSA ROUEN NORMANDIE

Paradigmes de programmation 1 / 2

Rappels

- Un paradigme de programmation est la façon d'aborder la résolution d'un problème
- Un paradigme de programmation repose sur des principes, des méthodes et outils

Paradigme de programmation impérative

- Principe : problème et solution sont représentés par des états (état initial et état final) et un programme représente le **comment** passer de l'état initial à l'état final
- Outil : l'affectation permet de passer d'un état i à un état $i + 1$
- Méthode : les schémas (séquentiel, conditionnel et itératif) permettent d'organiser les affectations

INSA ROUEN NORMANDIE

Paradigmes de programmation 2 / 2

Paradigme de programmation structurée

- Sous paradigme de la programmation impérative, il ajoute deux nouveaux outils et une nouvelle méthode
- Outil : sous-programmes (fonction et procédure)
- Méthode : passage de paramètre (association entre paramètres effectifs et paramètres formels)

INSA ROUEN NORMANDIE

Paradigme de la programmation logique

Paradigme de programmation déclarative

- Principe : on décrit le problème, le **quoi** et pas le comment
- Outil : un moteur interprète le problème pour trouver/construire la solution
- Méthode : *dépendant du sous paradigme*

Paradigme de programmation logique

- Sous paradigme de la programmation déclarative
- Principe : on décrit le problème en terme de faits (formules logiques avec uniquement des constantes), de règles (formules logiques faisant intervenir des variables), et une question (formule logique)
- Outil : trouver pour quelles constantes (solution(s)) l'interprétation de la question est vraie
- Méthode : algorithmes de démonstration

Prolog 1 / 2

- Langage du paradigme de la PROgrammation LOGique
- Langage inventé par Alain Colmerauer et Philippe Roussel vers 1972 (Marseille)
- « [...]le but n'était pas de faire un langage de programmation mais de traiter les langages naturels, en l'occurrence le Français. »

Fondement théorique

- Fondé sur les clauses de Horn de la logique des prédicats du premier ordre
- Les concepts fondamentaux sont le chaînage arrière, l'unification, la récursivité et le backtracking

INSA ROUEN NORMANDIE

Prolog 2 / 2

Principe

- Prolog permet au programmeur de déclarer des faits, des règles et de répondre à des questions

Éléments de base du langage

- Les clauses de Horn :
  - $r_1 \wedge r_2 \wedge \cdots \wedge r_n \rightarrow h$ avec $n \in \mathbb{N}$
  - En prolog, elles peuvent représenter aussi bien des faits que des règles
- Les atomes (commencent par une minuscule, ou entre simples côtes si utilisation de l'espace ou ne commençant pas par une minuscule) et les variables (commencent par une majuscule)
- Une question

- Il existe plusieurs syntaxes (ISO-Prolog, Edinburgh prolog, etc.)
- Tous les exemples de ce cours utilisent *swi-prolog*

Introduction de la syntaxe 1 / 3

- Fait :

```
fait(constante1,constante2,...,constante3).
```

- Règle :

```
regle(Var_ou_cons01, Var_ou_cons02, ...):-
                cond1(Var_ou_cons11, Var_ou_cons12, ...),
                cond2(Var_ou_cons21,Var_ou_cons22,..),
                ...,
                condn(Var_ou_consn1,Var_ou_consn2,...).
```

Remarques

- Un prédicat est un ensemble de clauses (faits et/ou règles) ayant le même nom et la même arité
- Deux prédicats de même nom mais d'arités différentes sont différents

Introduction de la syntaxe 2 / 3

Exemple : des faits

```
1 /* homme(X) est vrai si X est un homme */
2 homme(patrick).
3 homme(gerard).
4 homme(louis).
5 homme(pierre).
6
7 /* femme(X) est vrai si X est une femme */
8 femme(therese).
9 femme(sandrine).
10 femme(muriel).
11 femme(germaine).
12 femme(yvette).
13
14 /* enfant_parents(Enfant,Parent1,Parent2) est vrai si Enfant est un enfant de Parent1 et Parent2 */
15 enfant_parents(gerard,germaine,louis).
16 enfant_parents(therese,yvette,pierre).
17 enfant_parents(patrick,gerard,therese).
18 enfant_parents(muriel,gerard,therese).
19 enfant_parents(sandrine,gerard,therese).
20 enfant_parents(astride,jean,therese).
```

INSA ROUEN NORMANDIE

Introduction de la syntaxe 3 / 3

Exemple : des règles

```
22 /* enfant_parent(Enfant,Parent) est vrai si Enfant est un enfant de Parent */
23 enfant_parent(Enfant,Parent) :- enfant_parents(Enfant,Parent,_).
24 enfant_parent(Enfant,Parent) :- enfant_parents(Enfant,_,Parent).
25
26 /* pere_enfant(Pere,Enfant) est vrai si Pere est pere de Enfant */
27 pere_enfant(Pere,Enfant) :- homme(Pere), enfant_parent(Enfant,Pere).
28
29 /* mere_enfant(Mere,Enfant) est vrai si Mere est mere de Enfant */
30 mere_enfant(Mere,Enfant) :- femme(Mere), enfant_parent(Enfant,Mere).
31
32 /* grandpere_enfant(GrandPere,Enfant) vrai si GrandPere est grand-pere de Enfant */
33 grandpere_enfant(GrandPere,Enfant) :- pere_enfant(GrandPere,Parent), pere_enfant(Parent,Enfant).
34 grandpere_enfant(GrandPere,Enfant) :- pere_enfant(GrandPere,Parent), mere_enfant(Parent,Enfant).
35
36 /* grandmere_enfant(GrandMere,Enfant) vrai si GrandMere est grand-mere de Enfant */
37 grandmere_enfant(GrandMere,Enfant) :- mere_enfant(GrandMere,Parent), pere_enfant(Parent,Enfant).
38 grandmere_enfant(GrandMere,Enfant) :- mere_enfant(GrandMere,Parent), mere_enfant(Parent,Enfant).
```

INSA ROUEN NORMANDIE

L'interpréteur de swi-prolog 1 / 3

- On lance l'interpréteur à l'aide de la commande *swipl* (ou *pl*)
- On pose alors des « questions » en utilisant un prédicat ou plusieurs prédicats (séparés par des virgules) suivis d'un point
- Un programme prolog (faits et règles) est stocké dans un fichier dont le nom commence par une minuscule et a l'extension `.pl`
- On charge un programme en mettant le nom du fichier entre accolade suivi d'un point
- Quelques prédicats particuliers :
  - *halt/0* pour quitter
  - *help/1* pour obtenir de l'aide

INSA ROUEN NORMANDIE

L'interpréteur de swi-prolog 2 / 3

- Lorsque plusieurs solutions existent, l'interpréteur en affiche une et demande ce qu'il doit faire pour le reste
  - ? pour plus d'information
  - ; ou espace pour la solution suivante
  - a ou entrée pour arrêter l'affichage des solutions

```
$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- [genealogie].
true.

?- pere_enfant(gerard,patrick).
true .

?- pere_enfant(therese,patrick).
false.
```

L'interpréteur de swi-prolog 3 / 3

```
?- pere_enfant(X,patrick).
X = gerard ;
false.

?- pere_enfant(gerard,X).
X = patrick ;
X = muriel ;
X = sandrine ;
false.

?- grandpere_enfant(louis,patrick).
true .

?- grandpere_enfant(germaine,patrick).
false.

?- grandpere_enfant(X,patrick).
X = louis ;
X = pierre ;
false.

9 ?- grandpere_enfant(louis,X).
X = patrick ;
X = muriel ;
X = sandrine ;
false.
```

Conclusion

Ce qu'il faut retenir

- La logique permet de mécaniser le raisonnement
- La logique informatique étudie l'automatisation des démonstrations
- Il faut distinguer la syntaxe de la sémantique
- Il existe plusieurs logiques avec des pouvoirs d'expression différents
- La programmation logique est un paradigme de programmation
- Prolog est un langage de programmation logique

INSA ROUEN NORMANDIE

Références

[GRT02] Benoit Gugger, Denis Richard, and Jerzy Tomasik.
Calcul propositionnel.
http://www.lama.univ-savoie.fr/pagesmembres/saber/calcul-pro.pdf, 2002.

[Tri22] Markus Triska.
The power of prolog.
https://www.metalevel.at/prolog, 2022.

INSA ROUEN NORMANDIE