Le Raisonnement Mécanisé

Joël Quinqueton,
jq@lirmm.fr
Deug MASS, 4e semestre

Mécaniser le raisonnement

- Systèmes formels
- Déduction, abduction, induction
- Logique des propositions
- Logique des prédicats
- Programmation par contraintes
- Apprentissage automatique

Système Formel (SF)

- Un alphabet fini de symboles
- Un procédé de construction des mots du SF
- Un ensemble d 'axiomes qui sont des mots
- Un ensemble fini de règles qui permettent de déduire d 'un ensemble fini de mots un autre ensemble de mots

Exemple de SF

- Alphabet = {a, b, }
- Mots: suite quelconque de symboles
- Axiome unique: a a
- Règle unique x y →bx by
- Tous les mots de la forme:
- bb…ba b…ba

Autre exemple: le MU-puzzle

- Alphabet: {M,I,U}
- Mots: suite quelconque de symboles
- Axiome unique: MI
- Règles:
  - xI → xIU (production)
  - Mx → Mxx (production)
  - III → U (réécriture)
  - UU → *(rien)* (réécriture)

Le raisonnement

- Démontrer une assertion:
  - LP: Enchainer des règles et des faits
  - PP: Unifier des variables et des constantes
- Mécanisme d'unification:
  - Construire la correspondance entre expressions logiques

Logique des propositions (LP)

- Lettres propositionnelles, opérateurs, parenthèses: {p, q, r, …, ¬, ⊃, (,) }
- Mots:
  - Une lettre propositionnelle est un mot
  - Si x est un mot alors (x) et ¬x sont des mots
  - Si x et y sont des mots alors x⊃y est un mot
- Axiomes:
  - (x ⊃(y ⊃x))
  - (x ⊃(y ⊃z)) ⊃ ((x ⊃y) ⊃(x ⊃z))
  - (¬y ⊃¬x) ⊃(x ⊃y)
- Règle: (x), (x ⊃y) →(y)

Exemple de démonstration

- Le principe d'identité d'Aristote

Axiome 1

x=p et y=(p ⊃p)

(p ⊃((p ⊃p) ⊃p))

Axiome 2

x=z=p, et y=(p ⊃p)

règle de détachement

(p ⊃ (p ⊃p)) ⊃ (p ⊃p)

x=y=p

(p ⊃ (p ⊃p))

règle de détachement

$\boxed{\text{p ⊃p}}$

Interprétation

- Application Φ des mots de LP dans {0,1}
- Telle que Φ(¬x) = 1 - Φ(x)
- et Φ(x ⊃y) = 1 ssi Φ(x) = 0 ou Φ(y) = 1
- Logique décidable (Post 1921):
  - Théorème = Tautologie
  - Logique non contradictoire, complète et résoluble (ou décidable)

Vérification de tautologie

- Le principe d'identité d'Aristote

p	0	1
p ⊃p	1	1

Logique des prédicats (PP)

- Lettres propositionnelles, prédicats, variables, opérateurs, parenthèses, quantificateur: {p, q, …, a, b,…, X, Y,…,¬, ⊃, (,), ∀}
- Mots:
  - Une lettre propositionnelle est un mot
  - a(X,Y,…) mot si nombre de variables=arité de a, ∀X a(X,Y,…) est un mot où x est dite variable libre et les autres variables liées
  - Si x est un mot alors (x) et ¬x sont des mots
  - Si x et y sont des mots alors x⊃y est un mot
- Axiomes:
  - (x ⊃(y ⊃x))
  - (x ⊃(y ⊃z)) ⊃ ((x ⊃y) ⊃(x ⊃z))
  - (¬y ⊃¬x) ⊃(x ⊃y)
  - ((∀X) a(X)) ⊃ a(Y)
  - ((∀X) (x ⊃ y)) ⊃ (x ⊃ ((∀X) y))
- Règle de détachement: (x), (x ⊃y) →(y)
- Règle de généralisation: (x) →((∀X) x) où X est libre dans x

Unification

- Exemple précédent:
  - « Socrate est un homme »: homme(Socrate)
  - « Les hommes sont mortels »: ∀x homme(x) ⊃ mortel(x)
  - donc « Socrate est mortel »: mortel(Socrate)
- Construire la correspondance x=Socrate.

Terme

- Constante
- Variable
- Constante fonctionnelle suivie d'une suite de termes entre parenthèses
- Exemples:
  - A: terme clos (ne contient aucune variable)
  - x
  - F(x, a, g(x), f(a,b)): arité 4

Substitution

- Application de l'ensemble des variables dans l'ensemble des termes
- Un terme u est une instanciation du terme t ss'il existe une substitution Sub telle que Sub(t) = u

Unificateur

- Unificateur de A et B:
- Substitution sub telle que sub(A) = sub(B)
- Un unificateur peut etre plus général qu'un autre
- Trouver l'unificateur le plus général

Algorithme d'unification

- Théorème de Robinson:
- Il existe un algorithme (appelé algorithme d'unification) qui produit, pour deux termes, un unificateur le plus général, s'ils sont unifiables, ou un message indiquant l'absence d'un unificateur.

Unifier un terme et un atome

- UnifierAtome(a, t) {
  - SI a et t sont identiques, RETOURNER NIL.
  - SI a est une variable {
    - SI a apparaît dans t, RETOURNER echec.
    - RETOURNER la substitution {a/t}}.
  - SI t est une variable, RETOURNER la substitution {t/a}.
  - RETOURNER échec}.

Unifier deux termes

- Unifier (t1, t2)
  - SI t1 est un atome, UnifierAtome(t1, t2).
  - SI t2 est un atome, UnifierAtome(t2, t1).
  - F1 <- premier élément de t1, T1 <- reste de t1
  - F2 <- premier élément de t2, T2 <- reste de t2
  - Z1 <- Unifier (F1, F2)
  - Si Z1 = échec, RETOURNER échec.
    - G1 <- remplacer les variables de T1 selon les substitutions de Z1.
    - G2 <- remplacer les variables de T2 selon les substitutions de Z1.
    - Z2 <- Unifier (G1, G2)
    - Si Z2 = échec, RETOURNER échec.
      - RETOURNER une liste contenant les substitutions de Z1 et Z2.

Exemples

- Unifier(monkey(X), monkey(cheetah))
  - Z1 = Unifier(monkey, monkey) = NIL
  - Z2 = Unifier(X, cheetah) = (X/cheetah)
- Unifier(est(X,monkey), est(cheetah,Y))
  - Z1 = Unifier(est,est) = NIL
  - Z2 = Unifier((X,monkey), (cheetah,Y))
    - Z1 = Unifier(X, cheetah) = (X/cheetah)
    - Z2 = Unifier(monkey, Y) = (Y/monkey)
  - Retour: (X/cheeath, Y/monkey)

Exemple de termes complexes

- Unifier(sur(X, route(Y, lyon)), sur(lyon, route(paris, Z)))
  - Z1 = Unifier(sur, sur) = NIL
  - Z2 = Unifier((X, route(Y, lyon)), (lyon, route(paris, Z)))
    - Z1 = Unifier(X, lyon) = (X/lyon)
    - Z2 = Unifier(route(Y, lyon), route(paris, Z))
      - Z1 = Unifier(route, route) = NIL
      - Z2 = Unifier((Y, lyon), (paris, Z)) …etc…
  - Retour: (X/lyon, Y/paris, Z/lyon)

Exercices

- Unifier(sur(X, route(Y, lyon)), sur(lyon, route(paris, X)))
- Unifier(sur(X, route(Y, macon)), sur(lyon, route(paris, X)))
- Unifier(pres(X, route(Y, lyon)), pres(route(paris, lyon), X))

Limitations de la logique des prédicats

- Complétude: les théorèmes coïncident avec les formules vraies dans toute interprétation (1er th. de Gödel)
- Gödel construit un énoncé: « je ne suis pas un théorème »
- Si l'arithmétique formelle est non-contradictoire, alors elle n'est pas complète

Référence(s)

- Le raisonnement en Intelligence Artificielle: modèles, techniques et architectures pour les systèmes à base de connaissances
- Par Jean Paul Haton et al, Intereditions, 1991