Programmation (en) logique

Joseph Le Roux (d'après C. Rouveirol)

15/01/2014

Outline

Motivation

En Informatique, les logiques sont utilisées pour :

Modéliser les objets de manière formelle

- BDD,
- Pré/Post-conditions dans un programme. . .

Raisonner : utiliser la modélisation pour

- inférer de nouvelles connaissances,
- inférer de nouvelles propriétés,
- donner des preuves de correction

La logique est à la base de l'étude des raisonnements

raisonnement $\equiv$ des déductions que l'on peut faire sur les modèles formels

Bénéfices à l'utilisation des formalismes logiques

- modéles précis et non-ambigus,
- démarche rigoureuse,
- possibilité d'automatiser/semi-automatiser l'analyse des modèles (si logique axiomatisable)

Exemple 1

Soit une situation décrite de la manière suivante :

1. Si le train arrive en retard et qu'il n'y a pas de bus à la gare, alors j'arrive en retard;
2. Je ne suis pas en retard;
3. Le train est arrivé en retard.

On peut déduire

- Il y a avait un bus à la gare
- POURQUOI ?

Exemple 2

Considérons un autre exemple :

1. S'il pleut et que j'ai oublié mon parapluie alors je suis trempé;
2. Je ne suis pas trempé;
3. Il pleut;

et la déduction suivante :

- Je n'ai pas oublié mon parapluie
- Cette déduction est-elle fondée ?

Exemple 3

Jusqu'à présent, on pouvait modifier les raisonnements dans la logique propositionnelle.

- Un fait $\rightarrow$ une proposition
- Ce n'est pas toujours le cas $\rightarrow$ logique du premier ordre

Soit:

1. Les parents d'un enfant sont des ancêtres de cet enfant;
2. Les ancêtres d'une personne qui est l'ancêtre d'une deuxième personne sont également ancêtres de cette deuxième personne;
3. Deux personnes sont d'une même famille si et seulement si elles ont un ancêtre commun.

Utilisation de prédicats

Pour décrire les propriétés

- `enfant(X),...`

atomes ou variable propositionnelle: des énoncés dont nous ne cherchons pas pas connaître la structure interne. Notés $p, q, r, s, \ldots$.

connecteurs les connecteurs $\rightarrow$, $\leftrightarrow$, $\neg$, $\vee$, $\wedge$, $\ldots$

formules

- atomes
- si A est une formule, $\neg A$ est une formule
- si A et B sont des formules $A \leftrightarrow B$, $A \rightarrow B$, $A \vee B$ et $A \wedge B$ sont des formules
- formule vide notée $\Box$, et définie par $\perp$ (aburde, ou paradoxe, ou $\forall F, F \wedge \neg F$)

littéral positif = atome et littéral négatif = négation d'un atome

Sémantique : Interprétation

On veut associer à chaque formule une interprétation dans $\{v, f\}$ (ensembles des valeurs de vérité, ou *booléens*)

Valuation à chaque variable propositionnelle de la formule, on associe une valeur de vérité $\{v, f\}$.

Interprétation d'une formule une valuation pour les variables prop. de la formule. Notation $I(p)$ (resp. $I(F)$): la valeur de vérité de la variable p (resp. de la formule F) dans l'interprétation I.

Tables de vérités

- Si on peut librement assigner des valuations aux variables, il en va différemment des formules complexes, qui doivent obéir aux règles établies pour chaque connecteur
- On peut voir les connecteurs comme des opérations de composition interne sur les booléens:

A	B	$\neg A$	$A \vee B$	$A \wedge B$	$A \rightarrow B$	$A \leftrightarrow B$
v	v	f	v	v	v	v
v	f	f	v	f	f	f
f	v	v	v	f	v	f
f	f	v	f	f	v	v

Sémantique : satisfiabilité d'une formule

- Satisfiabilité notée $\models$
- Soit F une formule et I une interprétation de F.
- On définit la relation de satisfiabilité ($I \models F$) par:
  - F est une variable P: $I \models F$ ssi $I(P) = v$
  - F est une formule de la forme $\neg G$: $I \models F$ ssi $I(G) = f$
  - F est une formule de la forme $G \vee H$: $I \models F$ ssi $I(G) = v$ ou $I(H) = v$
  - F est une formule de la forme $G \wedge H$, $I \models F$ ssi $I(G) = v$ et $I(H) = v$

Sémantique

modèle : I est un modèle pour une formule si $I \models F$. I est un modèle pour un ensemble de formules $F_i, 1 \leq i \leq n$ si pour tout i $I \models F_i$

conséquence logique : La formule F est conséquence logique de la formule G (noté $G \models F$) si pour tout I tel que $I \models G$ alors $I \models F$

satisfiabilité :
- Une formule est *satisfiable* si il existe au moins une interprétation I telle $I \models F$. Une formule est dite *insatisfiable* s'il n'existe aucun I telle que $I \models F$.
- Une formule qui est vraie pour toute interprétation I est une *tautologie*. On dit aussi qu'elle est *valide*. Notation: $\models F$.

Sémantique et Syntaxe

Théorème de déduction L (métathéorème)

- $F_1, \ldots, F_n \models G$ ssi $F_1 \wedge \ldots F_n \rightarrow G$ est valide (est une tautologie).
- Autre notation : $F_1, \ldots, F_n \models G\ ssi \models F_1 \wedge \ldots F_n \rightarrow G$.

Corollaires

- Soit G une formule quelconque. Si E est un ensemble insatisfiable de formules, alors G est conséquence logique de E
- Si G est valide, G est conséquence logique d'un ensemble quelconque de formules
- Soit E un ensemble de formules et G une formule, $E \models G$ ssi $E \wedge \neg G$ est insatisfiable.

L'équivalence logique

- Soit F et G deux formules quelconques. F est logiquement équivalente à G, noté $F \equiv G$ ssi $F \models G$ et $G \models F$.
- $F \equiv G\ ssi \models F \leftrightarrow G$
- Quelques équivalences utiles
  1. $F \leftrightarrow G \equiv F \rightarrow G \wedge G \rightarrow F$
  2. $F \rightarrow G \equiv \neg F \vee G$
  3. $\neg(F \wedge G) \equiv \neg F \vee \neg G$ et $\neg(F \vee G) \equiv \neg F \wedge \neg G$ (loi de de Morgan)
  4. $\neg\neg F \equiv F$ (loi de double négation)
  5. $F \vee (G \wedge H) \equiv (F \vee G) \wedge (F \vee H)$ (distributivité de $\vee$ sur $\wedge$)
  6. $F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$ (distributivité de $\wedge$ sur $\vee$)

Preuve du dernier corollaire du TD

Soit E un ensemble de formules et G une formule, $E \models G$ ssi $E \wedge \neg G$ est insatisfiable.

$$
\begin{array}{rcll}
E \models & & G & \\
\models & & E \rightarrow G & \text{TD} \\
\models & & \neg E \vee G & \text{equiv.} \\
\models & & \neg E \vee G \vee \Box & \text{prop. de la disjonction} \\
\models & & \neg E \vee \neg\neg G \vee \Box & \text{equiv.} \\
\models & & (\neg E \vee \neg\neg G) \vee \Box & \text{assoc. disjonction} \\
\models & & \neg(E \wedge \neg G) \vee \Box & \text{de Morgan} \\
\models & & (E \wedge \neg G) \rightarrow \Box & \text{(équiv.)} \\
E, \neg G \models & & \Box & \text{TD}
\end{array}
$$

E,$\neg$ G est insatisfiable (prop. de $\models$)

Axiomatisation du calcul propositionnel

- Méthodes algorithmiques pour décider automatiquement et aussi efficacement que possible de la satisfiabilité d'un ensemble de formules.
- Un système de preuve est défini par un ensemble de règles (dites *d'inférence*) à appliquer *mécaniquement*, en ne tenant compte que de la syntaxe des formules traitées.
- Notation :

$$\frac{P_1 \qquad P_2}{C}$$

  où P_1 et P_2 sont les prémisses du séquent et C sa conclusion.
- Règle du modus ponens:

$$\frac{A \rightarrow B \qquad A}{B}$$

Propriétés d'un système d'inférence

On note classiquement $\vdash$ la notion de dérivation par un système d'inférence.

Correction
: Un système d'inférence est correct si, pour toute règle de ce système, la formule conclusion est conséquence logique de sa formule prémisse (de la conjonction de ses formules prémisses). Si $\vdash F$ alors $\models F$

Complétude
: Un système d'inférence est complet s'il suffit à prouver toute formule valide : pour toute formule F telle que $\models F$, alors $\vdash F$.

Formes normales

Pour simplifier la mécanisation, on ne s'intéressera qu'à des sous-types de formules logiques

- **Littéral** formule atomique ou négation d'une formule atomique
- **Clause** formule de la forme $l_1 \vee \ldots \vee l_n$ où chaque l_i est un littéral
- **FNC** formule de la forme $D_1 \wedge \ldots \wedge D_k$ où $k > 0$ et chaque D_i est une clause
- **Conjonction élémentaire** formule de la forme $l_1 \wedge \ldots \wedge l_n$ où chaque l_i est un littéral
- **FND** formule de la forme $C_1 \vee \ldots \vee C_k$ où $k > 0$ et chaque C_i est une conjonction élémentaires

Toute formule admet une forme normale conjonctive et disjonctive qui lui est logiquement équivalente

Mise sous forme normale (*algorithme*)

Utilisation des équivalences vues au transparent 14

1. Eliminer les $\rightarrow$ et les $\leftrightarrow$: application des équivalences 1 et 2
2. Limiter la portée des négations (faire “descendre” les négations): application des équivalences 3 et 4
3. Mettre sous forme clausale: application des équivalences 5 et 6

Notes:

- Pour une formule f donnée, il existe plusieurs f' telles que $f \equiv f'$ et f' en FNC.
- La taille de f' peut croître (exponentiellement)

Exemple: mise sous FNC

$$(p \rightarrow (q \leftrightarrow \neg r)) \wedge (r \leftrightarrow \neg s) \equiv$$

$$\neg p \vee ((q \rightarrow \neg r) \wedge (\neg r \rightarrow q)) \wedge (r \leftrightarrow \neg s) \equiv$$

$$\neg p \vee ((q \rightarrow \neg r) \wedge (\neg r \rightarrow q)) \wedge (r \rightarrow \neg s) \wedge (\neg s \rightarrow r) \equiv$$

$$\neg p \vee ((\neg q \vee \neg r) \wedge (r \vee q)) \wedge (r \rightarrow \neg s) \wedge (\neg s \rightarrow r) \equiv$$

$$(\neg p \vee \neg q \vee \neg r) \wedge (\neg p \vee r \vee q) \wedge (\neg r \vee \neg s) \wedge (s \vee r)$$

Résolution en calcul propositionnel

Méthode de démonstration par réfutation

- Ensemble de formules insatisfiable: un ensemble de formules $\{F_i\}, 1 \leq i \leq n$ est insatifiable ssi il n'existe aucun modèle M tel que pour tout i, $M \models F_i$.
- Principe de déduction: par le théorème de déduction, une formule G est conséquence logique d'un ensemble de formules $F = \{F_i, 1 \leq i \leq n\}$ ssi $F \cup \{\neg G\}$ est insatisfiable

Intuition

- $F_1, \ldots, F_n \models G$
- $F_1, \ldots, F_n, \neg G \models \Box$

Résolution en calcul propositionnel

2 règles

Résolution

Soient C_1 et C_2 deux clauses, soit p un littéral.

$$\frac{C_1 \vee p \qquad C_2 \vee \neg p}{R = C_1 \vee C_2}$$

On appelle R la résolvante de $C_1 \vee p$ et $C_2 \vee \neg p$

Factorisation

Soient C une clause et p une littéral

$$\frac{C \vee p \vee p}{C \vee p}$$

Dérivation par résolution (propositionnelle)

Soient E un ensemble de clauses et c une clause.

Une *dérivation* par résolution

de c à partir des *hypothèses* E est une suite de clause $r_1, \ldots, r_n$ telles que pour tout i, $1 \leq i \leq n$:

- $r_i \in E$
- il existe un $j \leq i$ tel que $\dfrac{r_j}{r_i}$ par factorisation
- il existe $j, k \leq i$ tels que $\dfrac{r_j \qquad r_k}{r_i}$ par résolution (r_i est appelé résolvante de r_j et r_k
- $r_n = c$

Une *réfutation* de E

est une dérivation par résolution de $\square$ à partir des hypothèses de E

Exemple de réfutation

$$E = \{p \lor q \lor r; \quad \neg p \lor q \lor r; \quad \neg q \lor r; \quad \neg r\}$$

$$\dfrac{\dfrac{\dfrac{\neg q \lor r \qquad \neg r}{\neg q} \qquad \dfrac{\dfrac{\dfrac{p \lor q \lor r \qquad \neg p \lor q \lor r}{q \lor q \lor r \lor r}}{q \lor r \lor r}}{q \lor r}}{r} \qquad \neg r}{\square}$$

Propriétés de la résolution

Correction Soient A, B et C des clauses. $\{A \vee C, B \vee \neg C\} \models A \vee B$

Non complétude Il existe une théorie clausale F et une clause C telle que $F \models C$, mais il n'existe pas de dérivation par résolution de C à partir de F. (par exemple $p \wedge q \models p \vee q$)

Complétude pour la réfutation Un ensemble de formules F est insatisfiable ssi on peut dériver la clause vide $\square$ de F par résolution. Si $F \models C$, il existe une réfutation à partir de $F \cup \{\neg C\}$.

Résolution unitiare et clauses de Horn

Clause de Horn clause contenant au plus un littéral positif

Résolution unitaire au moins un des deux parents est un littéral (ℓ)

$$\frac{\ell \qquad L \vee \neg \ell}{L}$$

Stratégie de résolution

1. Si $\square$ est dans F alors F est insatisfiable
2. Sinon, choisir une clause C et p une clause unitaire positive et C contient $\neg p$
3. Calculer la résolvante R de p et C
4. Remplacer dans F C par R

Résolution et clause de Horn

- La résolution unitaire n'est pas complète par réfutation pour les théories clausales quelconques. Exemple: $\{a \vee b; \neg a \vee b; a \vee \neg b; \neg a \vee \neg b\}$
- La résolution unitaire est complète par réfutation pour les ensembles de clauses de Horn. Exemple: $\{a; c; \neg a \vee b; \neg b \vee c\}$