Fundamentos de Programación Lógica

Paradigmas de Lenguajes de Programación

Departamento de Computación, FCEyN, UBA

28 de septiembre de 2017

Paradigma lógico

- Se basa en el uso de la lógica como un lenguaje de programación
- Se especifican
  - ciertos hechos y reglas de inferencia
  - un objetivo ("goal") a probar
- Un motor de inferencia trata de probar que el objetivo es consecuencia de los hechos y reglas
- Es declarativo: se especifican hechos, reglas y objetivo sin indicar cómo se obtiene éste último a partir de los primeros

Prolog

- Lenguaje de programación basado en este esquema que fue introducido a fines de 1971 (cf. "The Birth of Prolog", A. Colmerauer y P. Roussel, `www.lif-sud.univ-mrs.fr/~colmer/`)
- Los programas se escriben en un subconjunto de la lógica de primer orden
- El mecanismo teórico en el que se basa es el método de resolución

Prolog

Ejemplo de programa

```
habla(ale,ruso).
habla(juan,ingles).
habla(maria,ruso).
habla(maria,ingles).

seComunicaCon(X,Y):-habla(X,L),habla(Y,L),X\=Y
```

Ejemplo de goal

```
seComunicaCon(X,ale)
```

Nuestro enfoque

1. Lógica proposicional
2. Método de resolución para lógica proposicional
3. Repaso de lógica de primer orden
4. Método de resolución para lógica de primer orden
5. Cláusulas de Horn y resolución SLD, programación lógica

Sintaxis de la lógica proposicional

Dado un conjunto $\mathcal{V}$ de variables proposicionales, podemos definir inductivamente al conjunto de fórmulas proposicionales (o proposiciones) **Prop** de la siguiente manera:

1. Una variable proposicional $P_0, P_1, \ldots$ es una proposición
2. Si A, B son proposiciones, entonces:
   - $\neg A$ es una proposición
   - $A \wedge B$ es una proposición
   - $A \vee B$ es una proposición
   - $A \supset B$ es una proposición
   - $A \iff B$ es una proposición

Ejemplos: $A \vee \neg B$, $(A \wedge B) \supset (A \vee A)$

Semántica

- Una valuación es una función $v : \mathcal{V} \to \{\mathbf{T}, \mathbf{F}\}$ que asigna valores de verdad a las variables proposicionales
- Una valuación satisface una proposición A si $v \models A$ donde:

$$
\begin{array}{rcl}
v \models P & \textit{sii} & v(P) = \mathbf{T} \\
v \models \neg A & \textit{sii} & v \not\models A \ (\textit{i.e.}\ \text{no}\ v \models A) \\
v \models A \vee B & \textit{sii} & v \models A \ \text{o}\ v \models B \\
v \models A \wedge B & \textit{sii} & v \models A \ \text{y}\ v \models B \\
v \models A \supset B & \textit{sii} & v \not\models A \ \text{o}\ v \models B \\
v \models A \iff B & \textit{sii} & (v \models A \ \text{sii}\ v \models B)
\end{array}
$$

Tautologías y satisfactibilidad

Una proposición A es

- una tautología si $v \models A$ para toda valuación v
- satisfactible si existe una valuación v tal que $v \models A$
- insatisfactible si no es satisfactible

Un conjunto de proposiciones S es

- satisfactible si existe una valuación v tal que para todo $A \in S$, se tiene $v \models A$
- insatisfactible si no es satisfactible

Ejemplos

Tautologías

- $A \supset A$
- $\neg\neg A \supset A$
- $(A \supset B) \iff (\neg B \supset \neg A)$

Proposiciones insatisfactibles

- $(\neg A \vee B) \wedge (\neg A \vee \neg B) \wedge A$
- $(A \supset B) \wedge A \wedge \neg B$

Tautologías e insatisfactibilidad

Teorema

Una proposición A es una tautología sii $\neg A$ es insatisfactible

Dem.

$\Rightarrow$. Si A es tautología, para toda valuación v, $v \models A$. Entonces, $v \not\models \neg A$ (i.e. v no satisface $\neg A$).

$\Leftarrow$. Si $\neg A$ es insatisfactible, para toda valuación v, $v \not\models \neg A$. Luego $v \models A$.

Notar

Este resultado sugiere un método indirecto para probar que una proposición A es una tautología, a saber probar que $\neg A$ es insatisfactible

Forma normal conjuntiva (FNC)

- Un Literal es una variable proposicional P o su negación $\neg P$
- Una proposición A está en FNC si es una conjunción

  $$C_1 \wedge \ldots \wedge C_n$$

  donde cada C_i (llamado cláusula) es una disyunción

  $$B_{i1} \vee \ldots \vee B_{in_i}$$

  y cada B_{ij} es un literal
- Una FNC es una "conjunción de disyunciones de literales"

Forma normal conjuntiva

Ejemplos

- $(P \vee Q) \wedge (P \vee \neg Q)$ está en FNC
- $(P \vee Q) \wedge (P \vee \neg\neg Q)$ no está en FNC
- $(P \wedge Q) \vee P$ no está en FNC

Teorema

Para toda proposición A puede hallarse una proposición A' en FNC que es lógicamente equivalente a A.

Nota

A es lógicamente equivalente a B sii $A \iff B$ es una tautología

Notación conjuntista para FNC

- Dado que tanto $\vee$ como $\wedge$

  1. son conmutativos (i.e. $(A \vee B) \iff (B \vee A)$)
  2. son asociativos (i.e. $((A \vee B) \vee C) \iff (A \vee (B \vee C))$)
  3. son idempotentes (i.e. $(A \vee A) \iff A$)

  Podemos asumir que

  1. Cada cláusula C_i es distinta
  2. Cada cláusula puede verse como un conjunto de literales distintos

Notación conjuntista para FNC

Consecuentemente para una FNC podemos usar la notación

$$\{C_1, \ldots, C_n\}$$

donde cada C_i es un conjunto de literales

$$\{B_{i1}, \ldots, B_{in_i}\}$$

Por ejemplo, la FNC $(P \vee Q) \wedge (P \vee \neg Q)$ se anota

$$\{\{P, Q\}, \{P, \neg Q\}\}$$

Validez por refutación

Principio de demostración por refutación:

Probar que A es válido mostrando que $\neg A$ es insatisfactible

- Hay varias técnicas de demostración por refutación
  - Tableaux semántico (1960)
  - Procedimiento de Davis-Putnam (1960)
  - Resolución (1965)
- Nos vamos a concentrar en Resolución

Resolución

- Introducido por Alan Robinson en 1965

  A machine-oriented logic based on the resolution principle, J. of the ACM (12).

- Es simple de implementar
- Popular en el ámbito de demostración automática de teoremas
- Tiene una única regla de inferencia: la regla de resolución
- Si bien no es imprescindible, es conveniente asumir que las proposiciones están en forma normal conjuntiva

Principio fundamental del método de resolución

- Se basa en el hecho de que la siguiente proposición es una tautología

$$(A \vee P) \wedge (B \vee \neg P) \iff (A \vee P) \wedge (B \vee \neg P) \wedge (A \vee B)$$

- En efecto, el conjunto de cláusulas

$$\{C_1, \ldots, C_m, \{A, P\}, \{B, \neg P\}\}$$

es lógicamente equivalente a

$$\{C_1, \ldots, C_m, \{A, P\}, \{B, \neg P\}, \{A, B\}\}$$

Resolución

- En consecuencia, el conjunto de cláusulas

$$\{C_1, \ldots, C_m, \{A, P\}, \{B, \neg P\}\}$$

  es insatisfactible sii

$$\{C_1, \ldots, C_m, \{A, P\}, \{B, \neg P\}, \{A, B\}\}$$

  es insatisfactible
- La cláusula $\{A, B\}$ se llama resolvente de las cláusulas $\{A, P\}$ y $\{B, \neg P\}$
- El resolvente de las cláusulas $\{P\}$ y $\{\neg P\}$ es la cláusula vacía y se anota $\square$

Regla de resolución

- Dado un literal L, el opuesto de L (escrito $\overline{L}$) se define como:
  - $\neg P$ si $L = P$
  - P si $L = \neg P$
- Dadas dos cláusulas C_1, C_2, una cláusula C se dice resolvente de C_1 y C_2 sii, para algún literal L, $L \in C_1$, $\overline{L} \in C_2$, y

$$C = (C_1 - \{L\}) \cup (C_2 - \{\overline{L}\})$$

Ejemplos

Las cláusulas $\{A, B\}$ y $\{\neg A, \neg B\}$ tienen dos resolventes: $\{A, \neg A\}$ y $\{B, \neg B\}$.
Las cláusulas $\{P\}$ y $\{\neg P\}$ tienen a la cláusula vacía como resolvente

Regla de resolución

$$\frac{\{A_1, \ldots, A_m, Q\} \quad \{B_1, \ldots, B_n, \neg Q\}}{\{A_1, \ldots, A_m, B_1, \ldots, B_n\}}$$

El método de resolución

El proceso de agregar a un conjunto S la resolvente C de dos cláusulas C_1, C_2 que pertenecen a S (i.e. de aplicar la regla de resolución a S) se llama un paso de resolución.

Nota:

- Asumiremos que el resolvente C que se agrega a S no pertenecía ya a S
- Pasos de resolución preservan insatisfactibilidad

  S es insatisfactible sii $S \cup \{C\}$ es insatisfactible

El método de resolución

- Un conjunto de cláusulas se llama una refutación si contiene a la cláusula vacía (i.e. a $\square$).
- El método de resolución trata de construir una secuencia de conjuntos de cláusulas, obtenidas usando pasos de resolución hasta llegar a una refutación.

$$S_1 \Rightarrow S_2 \Rightarrow \ldots \Rightarrow S_{n-1} \Rightarrow S_n \ni \square$$

- En ese caso se sabe que el conjunto inicial de cláusulas es insatisfactible dado que
  1. cada paso de resolución preserva insatisfactibilidad
  2. el último conjunto de cláusulas es insatisfactible (contiene la cláusula vacía)

Ejemplo

Objetivo: mostrar que el conjunto de cláusulas
$\{\{P, Q\}, \{P, \neg Q\}, \{\neg P, Q\}, \{\neg P, \neg Q\}\}$ es insatisfactible.

1. $\{\{P, Q\}, \{P, \neg Q\}, \{\neg P, Q\}, \{\neg P, \neg Q\}\}$
2. $\{\{P, Q\}, \{P, \neg Q\}, \{\neg P, Q\}, \{\neg P, \neg Q\}, \{P\}\}$
3. $\{\{P, Q\}, \{P, \neg Q\}, \{\neg P, Q\}, \{\neg P, \neg Q\}, \{P\}, \{Q\}\}$
4. $\{\{P, Q\}, \{P, \neg Q\}, \{\neg P, Q\}, \{\neg P, \neg Q\}, \{P\}, \{Q\}, \{\neg P\}\}$
5. $\{\{P, Q\}, \{P, \neg Q\}, \{\neg P, Q\}, \{\neg P, \neg Q\}, \{P\}, \{Q\}, \{\neg P\}, \square\}$

Ejemplo

Objetivo: mostrar que el conjunto de cláusulas $\{\{A, B, \neg C\}, \{A, B, C\}, \{A, \neg B\}, \{\neg A\}\}$ es insatisfactible.

1. $\{\{A, B, \neg C\}, \{A, B, C\}, \{A, \neg B\}, \{\neg A\}\}$
2. $\{\{A, B, \neg C\}, \{A, B, C\}, \{A, \neg B\}, \{\neg A\}, \{A, B\}\}$
3. $\{\{A, B, \neg C\}, \{A, B, C\}, \{A, \neg B\}, \{\neg A\}, \{A, B\}, \{A\}\}$
4. $\{\{A, B, \neg C\}, \{A, B, C\}, \{A, \neg B\}, \{\neg A\}, \{A, B\}, \{A\}, \square\}$

Ejemplo

Objetivo: mostrar que el conjunto de cláusulas
$S = \{\{A, B, C\}, \{A\}, \{B\}\}$ es insatisfactible.

- No podemos aplicar ningún paso de resolución a S
- Por lo tanto, no puede llegarse a una refutación a partir S
- S debe ser satisfactible
- En efecto, tomar por ejemplo $v(A) = v(B) = \mathbf{T}$

Ejemplo

Objetivo: probar que R es consecuencia de $P \vee Q$, $P \supset R$, $Q \supset R$.

- Queremos probar que $((P \vee Q) \wedge (P \supset R) \wedge (Q \supset R)) \supset R$ es tautología.
- Esto es equivalente a probar que $\neg[((P \vee Q) \wedge (P \supset R) \wedge (Q \supset R)) \supset R]$ es insatisfactible.
- Para ello usamos resolución sobre su FNC.

Terminación de la regla de resolución

- La aplicación reiterada de la regla de resolución siempre termina (suponiendo que el resolvente que se agrega es nuevo)
- En efecto, notar que
  1. El resolvente (i.e. la cláusula nueva que se agrega) se forma con los literales distintos que aparecen en el conjunto de cláusulas de partida S
  2. Hay una cantidad finita de literales en el conjunto de cláusulas de partida S
- En el peor de los casos, la regla de resolución podrá generar una nueva cláusula por cada combinación diferente de literales distintos de S

Corrección y completitud

- El siguiente resultado establece la corrección y completitud del método de resolución

Teorema

Dado un conjunto finito S de cláusulas,

S es insatisfactible sii tiene una refutación

Recapitulando

Para probar que A es una tautología hacemos lo siguiente:

1. Calculamos la forma normal conjuntiva de $\neg A$
2. Aplicamos el método de resolución
3. Si hallamos una refutación:
   - $\neg A$ es insatisfactible,
   - Y, por lo tanto, A es una tautología
4. Si no hallamos ninguna refutación:
   - $\neg A$ es satisfactible,
   - Y, por lo tanto, A no es una tautología

Lenguaje de primer orden

Un lenguaje de primer orden (LPO) $\mathcal{L}$ consiste en:

1. Un conjunto numerable de constantes $c_0, c_1, \ldots$
2. Un conjunto numerable de símbolos de función con aridad $n > 0$ (indica el número de argumentos) $f_0, f_1, \ldots$
3. Un conjunto numerable de símbolos de predicado con aridad $n \geq 0$, $P_0, P_1, \ldots$. La aridad indica el número de argumentos que toma (si $n = 0$, es una variable proposicional)

Ejemplo: Lenguaje de primer orden para la aritmética

Constantes: 0; Símbolos de función: $S, +, *$; Símbolos de predicado: $<$, $=$.

Términos de primer orden

Sea $\mathcal{V} = \{x_0, x_1, \ldots\}$ un conjunto numerable de variables y $\mathcal{L}$ un LPO. El conjunto de $\mathcal{L}$-términos se define inductivamente como:

1. Toda constante de $\mathcal{L}$ y toda variable es un $\mathcal{L}$-término
2. Si $t_1, \ldots, t_n \in \mathcal{L}$-términos y f es un símbolo de función de aridad n, entonces $f(t_1, \ldots, t_n) \in \mathcal{L}$-términos

Ejemplo: Aritmética (cont.)

$S(0), +(S(0), S(S(0))), *(S(x_1), +(x_2, S(x_3)))$

Fórmulas atómicas

Sea $\mathcal{V}$ un conjunto numerable de variables y $\mathcal{L}$ un LPO. El conjunto de $\mathcal{L}$-fórmulas atómicas se define inductivamente como:

1. Todo símbolo de predicado de aridad 0 es una $\mathcal{L}$-fórmula atómica
2. Si $t_1, \ldots, t_n \in \mathcal{L}$-términos y P es un símbolo de predicado de aridad n, entonces $P(t_1, \ldots, t_n)$ es una $\mathcal{L}$-fórmula atómica

Ejemplo: Aritmética (cont.)

$< (0, S(0)), < (x_1, +(S(0), x_2))$

Fórmulas de primer orden

Sea $\mathcal{V}$ un conjunto numerable de variables y $\mathcal{L}$ un LPO. El conjunto de $\mathcal{L}$-fórmulas se define inductivamente como:

1. Toda $\mathcal{L}$-fórmula atómica es una $\mathcal{L}$-fórmula
2. Si $A, B \in \mathcal{L}$-fórmulas, entonces $(A \wedge B), (A \vee B), (A \supset B), (A \iff B)$ y $\neg A$ son $\mathcal{L}$-fórmulas
3. Para toda variable x_i y cualquier $\mathcal{L}$-fórmula A, $\forall x_i.A$ y $\exists x_i.A$ son $\mathcal{L}$-fórmulas

Ejemplo: Aritmética (cont.)

- $\forall x.\forall y.(x < y \supset \exists z.y = x + z)$
- $\forall x.\forall y.((x < y \vee y < x) \vee x = y)$

Variables libres y ligadas

Las variables pueden ocurrir libres o ligadas.

- Los cuantificadores ligan variables
- Usamos $FV(A)$ y $BV(A)$ para referirnos a las variables libres y ligadas, resp., de A
- $FV(A)$ y $BV(A)$ se pueden definir por inducción estructural en A

Ejemplo

Si $A = \forall x.(R(x, y) \supset P(x))$, entonces $FV(A) = \{y\}$ y $BV(A) = \{x\}$

Variables libres y ligadas

- Una fórmula A se dice rectificada si
  - $FV(A)$ y $BV(A)$ son disjuntos y
  - Cuantificadores distintos de A ligan variables distintas
- Toda fórmula se puede rectificar (renombrando variables ligadas) a una fórmula lógicamente equivalente
- Una sentencia es una fórmula cerrada (i.e. sin variables libres).

Estructura de primer orden

Dado un lenguaje de primer orden $\mathcal{L}$, una estructura para $\mathcal{L}$, $\mathbf{M}$, es un par $\mathbf{M} = (M, I)$ donde

- M (dominio) es un conjunto no vacío
- I (función de interpretación) asigna funciones y predicados sobre M a símbolos de $\mathcal{L}$ de la siguiente manera:
  1. Para toda constante c, $I(c) \in M$
  2. Para todo f de aridad $n > 0$, $I(f) : M^n \to M$
  3. Para todo predicado P de aridad $n \geq 0$, $I(P) : M^n \to \{\mathbf{T}, \mathbf{F}\}$

Satisfactibilidad

Asignación

Sea $\mathbf{M}$ una estructura para $\mathcal{L}$. Una asignación es una función $s : \mathcal{V} \to M$

- Si s es una asignación y $a \in M$, usamos la notación $s[x \leftarrow a]$ para denotar la asignación que se comporta igual que s salvo en el elemento x, en cuyo caso retorna a

Satisfactibilidad

La relación $s \models_{\mathbf{M}} A$ establece que la asignación s satisface la fórmula A en la estructura $\mathbf{M}$

- Vamos a definir la relación $s \models_{\mathbf{M}} A$ usando inducción estructural en A

Satisfactibilidad

La relación $s \models_{\mathbf{M}} A$ se define inductivamente como:

$$
\begin{array}{rcl}
s \models_{\mathbf{M}} P(t_1, \ldots, t_n) & \textit{sii} & P_{\mathbf{M}}(s(t_1), \ldots, s(t_n)) \\
s \models_{\mathbf{M}} \neg A & \textit{sii} & s \not\models_{\mathbf{M}} A \\
s \models_{\mathbf{M}} (A \wedge B) & \textit{sii} & s \models_{\mathbf{M}} A \text{ y } s \models_{\mathbf{M}} B \\
s \models_{\mathbf{M}} (A \vee B) & \textit{sii} & s \models_{\mathbf{M}} A \text{ o } s \models_{\mathbf{M}} B \\
s \models_{\mathbf{M}} (A \supset B) & \textit{sii} & s \not\models_{\mathbf{M}} A \text{ o } s \models_{\mathbf{M}} B \\
s \models_{\mathbf{M}} (A \iff B) & \textit{sii} & (s \models_{\mathbf{M}} A \text{ sii } s \models_{\mathbf{M}} B) \\
s \models_{\mathbf{M}} \forall x_i.A & \textit{sii} & s[x_i \leftarrow a] \models_{\mathbf{M}} A \text{ para todo } a \in M \\
s \models_{\mathbf{M}} \exists x_i.A & \textit{sii} & s[x_i \leftarrow a] \models_{\mathbf{M}} A \text{ para algún } a \in M
\end{array}
$$

Validez

- Una fórmula A es satisfactible en $\mathbf{M}$ sii existe una asignación s tal que

$$s \models_{\mathbf{M}} A$$

- Una fórmula A es satisfactible sii existe un $\mathbf{M}$ tal que A es satisfactible en $\mathbf{M}$. En caso contrario se dice que A es insatisfactible.
- Una fórmula A es válida en $\mathbf{M}$ sii

$$s \models_{\mathbf{M}} A, \text{ para toda asignación } s$$

- Una fórmula A es válida sii es válida en toda estructura $\mathbf{M}$.
- **Nota:** A es válida sii $\neg A$ es insatisfactible.

Teorema de Church

No existe un algoritmo que pueda determinar si una fórmula de primer orden es válida

- Como consecuencia el método de resolución que veremos para la lógica de primer orden no es un procedimiento efectivo (i.e. un algoritmo)
- Es un procedimiento de semi-decisión:
  - si una sentencia es insatisfactible hallará una refutación,
  - pero si es satisfactible puede que no se detenga