Razonamiento automático

Tema 2: Cálculo semántico proposicional con MACE

J.A. Alonso, J. Borrego, A. Chávez y F.J. Martín

Dpto. Ciencias de la Computación e Inteligencia Artificial

Universidad de Sevilla

Sintaxis de la lógica proposicional

- El alfabeto proposicional:
  - *símbolos proposicionales*.
  - *conectivas lógicas*:
    - $\neg$ (negación),
    - $\wedge$ (conjunción),
    - $\vee$ (disyunción),
    - $\rightarrow$ (condicional),
    - $\leftrightarrow$ (equivalencia).
  - símbolos auxiliares: "(" y ")".
- Las fórmulas proposicionales:
  - símbolos proposicionales
  - $\neg F$, $(F \wedge G)$, $(F \vee G)$, $(F \rightarrow G)$, $(F \leftrightarrow G)$.
- Eliminación de paréntesis:
  - Paréntesis externos.
  - Precedencia: $\neg, \wedge, \vee \rightarrow, \leftrightarrow$
  - Asociatividad: $\wedge$ y $\vee$ asocian por la derecha
- Sintaxis en OTTER/MACE

Usual	$\neg$	$\wedge$	$\vee$	$\rightarrow$	$\leftrightarrow$
OTTER/MACE	-	&	|	->	<->

Semántica: Verdad e interpretación

- Los valores de verdad:
  - verdadero 1 (en MACE , T)
  - falso 0 (en MACE , F)
- Las funciones de verdad:

i	$\neg i$
1	0
0	1

i	j	i $\wedge$ j	i $\vee$ j	i $\rightarrow$ j	i $\leftrightarrow$ j
1	1	1	1	1	1
1	0	0	1	0	0
0	1	0	1	1	0
0	0	0	0	1	1

Modelo de una fórmula

- El significado de una fórmula en una interpretación
  - Fórmula: $F = (p \vee q) \wedge (\neg q \vee r)$
  - Interpretación: I, $I(p) = I(r) = 1, I(q) = 0$

$$
\begin{array}{ccccccc}
(p & \vee & q) & \wedge & (\neg q & \vee & r) \\
(1 & \vee & 0) & \wedge & (\neg 0 & \vee & 1) \\
 & 1 & & \wedge & (1 & \vee & 1) \\
 & 1 & & \wedge & & 1 & \\
 & & & 1 & & &
\end{array}
$$

  - F es válida en I
  - I es modelo de F
  - $I \models F$

Modelo de una fórmula

- Significado de una fórmula en una interpretación
  - Fórmula: $F = (p \lor q) \land (\neg q \lor r)$
  - Interpretación: J, $J(r) = 1, J(p) = J(q) = 0$

$$
\begin{array}{ccccccc}
(p & \lor & q) & \land & (\neg q & \lor & r) \\
(0 & \lor & 0) & \land & (\neg 0 & \lor & 1) \\
 & 0 & & \land & (1 & \lor & 1) \\
 & 0 & & \land & & 1 & \\
 & & & 0 & & &
\end{array}
$$

  - F no es válida en J
  - J no es modelo de F
  - $J \not\models F$

Comprobación de modelo con MACE

- Determinar si la la interpretación I definida por $I(p) = I(r) = 1$, $I(q) = 0$ es un modelo de la fórmula $(p \lor q) \land (\neg q \lor r)$
- Formalización en MACE

"modelo_fla_1.in"

```
formula_list(usable).
 (p | q) & (-q | r).
end_of_list.

list(mace_constraints).
 assign(p,T).
 assign(q,F).
 assign(r,T).
end_of_list.
```

- Ejecución en MACE :

```
mace2 -p < modelo_fla_1.in > modelo_fla_1.out
```

Comprobación de modelo con MACE

- Determinar si la la interpretación J definida por $J(r) = 1$, $J(p) = 0$ y $J(q) = 0$ es un modelo de la fórmula $(p \vee q) \wedge (\neg q \vee r)$
- Formalización en MACE

"modelo_fla_2.in"

```
formula_list(usable).
 (p | q) & (-q | r).
end_of_list.

list(mace_constraints).
 assign(p,F).
 assign(q,F).
 assign(r,T).
end_of_list.
```

- Ejecución en MACE :
  `mace2 -p < modelo_fla_2.in > modelo_fla_2.out`

Fórmulas satisfacibles e insatisfacibles

- Def.: F es satisfacible syss F tiene modelo
- Def.: F es insatisfacible syss F no tiene modelo
- Ejemplo:

  $(p \to q) \land (q \to r)$ es satisfacible

  $$I(p) = I(q) = I(r) = 0$$

  $p \land \neg p$ es insatisfacible

p	$\neg p$	$p \land \neg p$
1	0	0
0	1	0

Satisfacibilidad con MACE

- Determinar si la fórmula $(p \to q) \land (q \to r)$ es satisfacible
- Formalización en MACE

"satisfacibilidad_1.in"

```
formula_list(sos).
 (p -> q) & (q -> r).
end_of_list.
```

- Ejecución en MACE :

```
mace2 -p < satisfacibilidad_1.in
         > satisfacibilidad_1.out
```

Satisfacibilidad con MACE

- Determinar si la fórmula $p \wedge \neg p$ es satisfacible
- Formalización en MACE

”satisfacibilidad_2.in”

```
formula_list(sos).
 p & -p.
end_of_list.
```

- Ejecución en MACE :

```
mace2 -p < satisfacibilidad_2.in
         > satisfacibilidad_2.out
```

Fórmulas válidas

- Def.: F es válida syss toda interpretación de F es modelo de F
- Fórmula: $F = (p \rightarrow q) \lor (q \rightarrow p)$

p	q	$(p \rightarrow q)$	$(q \rightarrow p)$	$(p \rightarrow q) \lor (q \rightarrow p)$
1	1	1	1	1
1	0	0	1	1
0	1	1	0	1
0	0	1	1	1

- $\models F$
- $\not\models (p \rightarrow q)$

Satisfacibilidad y validez

- Problemas de satisfacibilidad y validez
  - El problema de la satisfacibilidad: Dada F determinar si es satisfacible.
  - El problema de la validez: Dada F determinar si es válida.
- Relaciones entre validez y satisfacibilidad:
  - F es válida $\iff$ $\neg F$ es insatisfacible
  - F es válida $\implies$ F es satisfacible
  - F es satisfacible $\not\Longrightarrow$ $\neg F$ es insatisfacible
- El problema de la satisfacibilidad es NP–completo

Validez con MACE

- Determinar si la fórmula $(p \to q) \lor (q \to p)$ es válida
- Formalización en MACE

”validez_1.in”

```
formula_list(usable).
 -((p -> q) | (q -> p)).
end_of_list.
```

- Ejecución en MACE :
  `mace2 -p < validez_1.in > validez_1.out`

Validez con MACE

- Determinar si la fórmula $p \to q$ es válida
- Formalización en MACE

"validez_2.in"

```
formula_list(usable).
 -(p -> q).
end_of_list.
```

- Ejecución en MACE :

```
mace2 -p < validez_2.in > validez_2.out
```

Modelos de un conjunto de fórmulas

- Def.: I es modelo de S syss para toda $F \in S, I \models F$
- $I \models S$
- Ejemplo:
  $S = \{(p \vee q) \wedge (\neg q \vee r), q \rightarrow r\}$
  $I_1(p) = 1, I_1(q) = 0, I_1(r) = 1 \Longrightarrow I_1 \models S$
  $I_2(p) = 0, I_2(q) = 1, I_2(r) = 0 \Longrightarrow I_2 \not\models S$

Comprobación de modelos con MACE

- Determinar si la interpretación I definida por $I(p) = 1$,$I(q) = 0$, $I(r) = 1$ es un modelo del conjunto de fórmulas $S = \{(p \vee q) \wedge (\neg q \vee r), q \rightarrow r\}$
- Formalización en MACE

"modelo_cjt_1.in"

```
formula_list(usable).
 (p | q) & (-q | r).
 q -> r.
end_of_list.

list(mace_constraints).
 assign(p,T).
 assign(q,F).
 assign(r,T).
end_of_list.
```

Comprobación de modelos con MACE

- Determinar si la interpretación I definida por $I(p) = 0$,$I(q) = 1$, $I(r) = 0$ es un modelo del conjunto de fórmulas $S = \{(p \vee q) \wedge (\neg q \vee r), q \rightarrow r\}$
- Formalización en MACE

"modelo_cjt_2.in"

```
formula_list(usable).
 (p | q) & (-q | r).
 q -> r.
end_of_list.

list(mace_constraints).
 assign(p,F).
 assign(q,T).
 assign(r,F).
end_of_list.
```

Conjuntos consistentes

- Def.: S es consistente syss S tiene modelo
- Def.: S es inconsistente syss S no tiene modelo
- Ejemplo: $\{(p \vee q) \wedge (-q \vee r), p \rightarrow r\}$ es consistente
- Ejemplo: $\{(p \vee q) \wedge (-q \vee r), p \rightarrow r, \neg r\}$ es inconsistente:

	p	q	r	$(p \vee q)$	$(\neg q \vee r)$	$(p \vee q) \wedge (\neg q \vee r)$	$p \rightarrow r$	$\neg r$
I_1	0	0	0	0	1	0	1	1
I_2	0	0	1	0	1	0	1	0
I_3	0	1	0	1	0	0	1	1
I_4	0	1	1	1	1	1	1	0
I_5	1	0	0	1	1	1	0	1
I_6	1	0	1	1	1	1	1	0
I_7	1	1	0	1	0	0	0	1
I_8	1	1	1	1	1	1	1	0

Consistencia con MACE

- Determinar si el conjunto $\{(p \vee q) \wedge (-q \vee r), p \rightarrow r\}$ es consistente
- Formalización en MACE

"consistencia.in"

```
formula_list(sos).
 (p | q) & (-q | r).
 p -> r.
end_of_list.
```

Consistencia con MACE

- Determinar si el conjunto $\{(p \vee q) \wedge (-q \vee r), p \rightarrow r, \neg r\}$ inconsistente
- Formalización en MACE

"inconsistencia.in"

```
formula_list(sos).
 (p | q) & (-q | r).
 p -> r.
 -r.
end_of_list.
```

Consecuencia lógica

- F es consecuencia de S syss todos los modelos de S son modelos de F
- Representación: $S \models F$
- Ejemplo: $\{p \rightarrow q, q \rightarrow r\} \models p \rightarrow r$

	p	q	r	$p \rightarrow q$	$q \rightarrow r$	$p \rightarrow r$
I_1	0	0	0	1	1	1
I_2	0	0	1	1	1	1
I_3	0	1	0	1	0	1
I_4	0	1	1	1	1	1
I_5	1	0	0	0	1	0
I_6	1	0	1	0	1	1
I_7	1	1	0	1	0	0
I_8	1	1	1	1	1	1

- Ejemplo: $\{p\} \not\models p \wedge q$

p	q	$p \wedge q$
1	1	1
1	0	0
0	1	0
0	0	0

Consecuencia, validez y consistencia

- Las siguientes condiciones son equivalentes:
  - $\{F_1, \dots, F_n\} \models G$
  - $\models F_1 \land \dots \land F_n \to G$
  - $\neg(F_1 \land \dots \land F_n \to G)$ es insatisfacible
  - $\{F_1, \dots, F_n, \neg G\}$ es inconsistente

Consecuencia lógica con MACE

- Determinar si $\{p \leftrightarrow q, r \leftrightarrow (p \wedge q)\} \not\models p \wedge r$
- Formalización en MACE

”no-consecuencia.in”

```
formula_list(sos).
 p <->q.
 r <-> (p & q).
 -(p & r).
end_of_list.
```

Consecuencia lógica con MACE

- Determinar si $\{p \leftrightarrow q, r \leftrightarrow (p \land q)\} \models p \leftrightarrow r$
- Formalización en MACE

"consecuencia.in"

```
formula_list(sos).
 p <->q.
 r <-> (p & q).
 -(p <-> r).
 end_of_list.
```

Problema de los animales con MACE

- Base de conocimiento
  - Base de reglas:
    1. Si el animal tiene pelos es mamífero
    2. Si el animal da leche es mamífero
    3. Si el animal es un mamífero y tiene pezuñas es ungulado
    4. Si el animal es un mamífero y rumia es ungulado
    5. Si el animal es un ungulado y tiene cuello largo es una jirafa
    6. Si el animal es un ungulado y tiene rayas negras es una cebra
  - Base de hechos:
    1. El animal tiene pelos
    2. El animal tiene pezuñas
    3. El animal tiene rayas negras
  - Consecuencia:

    El animal es una cebra

Problema de los animales con MACE

- Formalización en MACE

"animales_1.in"

```
formula_list(sos).
tiene_pelos | da_leche -> es_mamifero.
es_mamifero & (tiene_pezuñas | rumia) -> es_ungulado.
es_ungulado & tiene_cuello_largo -> es_jirafa.
es_ungulado & tiene_rayas_negras -> es_cebra.

tiene_pelos & tiene_pezuñas & tiene_rayas_negras.

-es_cebra.
end_of_list.
```

Problema de los animales con MACE

- Modelo del problema de los animales
- Formalización en MACE

"animales_2.in"

```
formula_list(sos).
tiene_pelos | da_leche -> es_mamifero.
es_mamifero & (tiene_pezuñas | rumia) -> es_ungulado.
es_ungulado & tiene_cuello_largo -> es_jirafa.
es_ungulado & tiene_rayas_negras -> es_cebra.

tiene_pelos & tiene_pezuñas & tiene_rayas_negras.

% -es_cebra.
end_of_list.
```

Problema del coloreado de pentágonos

- Demostrar que es imposible colorear los vértices de un pentágono de rojo o azul de forma que los vértices adyacentes tengan colores distintos
- Representación: Se numeran los vértices consecutivos del pentágono con los números 1, 2, 3, 4 y 5. Se usan los símbolos r_i $(1 \leq i \leq 5)$ para representar que el vértice i es rojo y los símbolos a_j $(1 \leq j \leq 5)$ para representar que el vértice j es azul

Problema del coloreado de pentágonos

- Formalización en MACE

"pentagono_2.in"

```
formula_list(usable).
% El vértice i (1 <= i <= 5) es azul o rojo:
a1 | r1.        a2 | r2.         a3 | r3.
a4 | r4.        a5 | r5.
% Dos vértices adyacentes no pueden ser azules:
-(a1 & a2).    -(a2 & a3).      -(a3 & a4).
-(a4 & a5).    -(a5 & a1).
% Dos vértices adyacentes no pueden ser rojos:
-(r1 & r2).    -(r2 & r3).      -(r3 & r4).
-(r4 & r5).    -(r5 & r1).
end_of_list.
```

Problema del coloreado de pentágonos

- Demostrar que es posible colorear los vértices de un pentágono de rojo, azul o negro de forma que los vértices adyacentes tengan colores distintos
- Representación: Se numeran los vértices consecutivos del pentágono con los números 1, 2, 3, 4 y 5. Se usan los símbolos r_i $(1 \leq i \leq 5)$ para representar que el vértice i es rojo, los símbolos a_j $(1 \leq j \leq 5)$ para representar que el vértice j es azul y los símbolos n_j $(1 \leq j \leq 5)$ para representar que el vértice j es negro

Problema del coloreado de pentágonos

- Formalización en MACE

"pentagono_3.in"

```
formula_list(usable).
% El vértice i (1 <= i <= 5) es azul, rojo o negro:
a1 | r1 | n1.       a2 | r2 | n2.       a3 | r3 | n3.
a4 | r4 | n4.       a5 | r5 | n5.
% Dos vértices adyacentes no pueden ser azules:
-(a1 & a2).         -(a2 & a3).         -(a3 & a4).
-(a4 & a5).         -(a5 & a1).
% Dos vértices adyacentes no pueden ser rojos:
-(r1 & r2).         -(r2 & r3).         -(r3 & r4).
-(r4 & r5).         -(r5 & r1).
% Dos vértices adyacentes no pueden ser negros:
-(n1 & n2).         -(n2 & n3).         -(n3 & n4).
-(n4 & n5).         -(n5 & n1).
end_of_list.
```

Bibliografía

- Bundy, A. *The computer modelling of mathematical reasoning*. (Academic Press, 1983)
    - Cap. 2 “Arguments about propositions”
    - Cap. 3: “The internal structure of propositions”
- Chang, C.L. y Lee, R.C.T *Symbolic logic and mechanical theorem proving*. (Academic Press, 1973)
    - Cap. 2 “The propositional logic”
- Genesereth, M.R. *Computational logic*
    - Cap. 2 “Propositional logic”
    - Cap. 3 “Truth table method”
- Nilsson, N.J. *Inteligencia artificial (Una nueva síntesis)*. (McGraw–Hill, 2000)
    - Cap. 13 “El cálculo proposicional”