Tema 2: Lógica de primer orden en PVS

José A. Alonso Jiménez

Jose-Antonio.Alonso@cs.us.es
http://www.cs.us.es/~jalonso

Dpto. de Ciencias de la Computación e Inteligencia Artificial

Universidad de Sevilla

Reglas y tácticas para cuantificadores

Izquierda		Derecha	
$\dfrac{\Gamma, A[x/t] \Longrightarrow \Delta}{\Gamma, \forall x A \Longrightarrow \Delta}$	[∀I]	$\dfrac{\Gamma \Longrightarrow A[x/c], \Delta}{\Gamma \Longrightarrow \forall x A, \Delta}$	[∀D]
$\dfrac{\Gamma, A[x/c] \Longrightarrow \Delta}{\Gamma, \exists x A \Longrightarrow \Delta}$	[∃I]	$\dfrac{\Gamma \Longrightarrow A[x/t], \Delta}{\Gamma \Longrightarrow \exists x A, \Delta}$	[∃D]

- **La constante c es nueva (i.e. no aparece en el secuente de la conclusión) y se llama constante de Skolem**

- **Tácticas para cuantificadores**
  - **Para ∀D y ∃I:** `skolem`, `skolem!` y `skosimp`
  - **Para ∃D y ∀I:** `inst` e `inst?`
  - **Estrategia para proposicional y cuantificadores:** `reduce`

Las tácticas `skolem` e `inst`

- **Teorema** (`ej1`)**:** $\forall x(P(x) \rightarrow (\exists x P(x)))$
- **Teoría** (`lpo.pvs`)

```
lpo: THEORY
 BEGIN
  T: TYPE
  P: [T -> bool]
  x: VAR T

  ej1: THEOREM
    FORALL x: (P(x) IMPLIES (EXISTS x: P(x)))
 END lpo
```

- **Prueba del** `ej1` **con** `skolem` **e** `inst`

```
ej1 :
  |-------
{1}   FORALL x: (P(x) IMPLIES (EXISTS x: P(x)))
```

Las tácticas skolem e inst

```
Rule? (skolem 1 "a")
For the top quantifier in 1, we introduce Skolem constants: a, this simplifies to:
ej1 :

  |-------
{1}   (P(a) IMPLIES (EXISTS x: P(x)))

Rule? (flatten)
Applying disjunctive simplification to flatten sequent, this simplifies to:

ej1 :

{-1}  P(a)
  |-------
{1}   (EXISTS x: P(x))

Rule? (inst 1 "a")
Instantiating the top quantifier in 1 with the terms: a,
Q.E.D.
```

Las tácticas `skolem!` e `inst?`

- **Prueba del `ej1` con `skolem!` e `inst?`**

```
ej1 :

  |-------
{1}   FORALL x: (P(x) IMPLIES (EXISTS x: P(x)))

Rule? (skolem!)
Skolemizing, this simplifies to:
ej1 :

  |-------
{1}   (P(x!1) IMPLIES (EXISTS x: P(x)))

Rule? (flatten)
Applying disjunctive simplification to flatten sequent, this simplifies to:
```

Las tácticas `skolem!` e `inst?`

```
ej1 :

{-1}  P(x!1)
  |-------
{1}   (EXISTS x: P(x))

Rule? (inst?)
Found substitution:
x gets x!1,
Using template: P(x)
Instantiating quantified variables,
Q.E.D.
```

La táctica `skosimp`

- **Prueba del `ej1` con `skosimp`**

```
ej1 :

  |-------
{1}   FORALL x: (P(x) IMPLIES (EXISTS x: P(x)))

Rule? (skosimp)
Skolemizing and flattening, this simplifies to:
ej1 :

{-1}  P(x!1)
  |-------
{1}   (EXISTS x: P(x))

Rule? (inst?)
Found substitution: x gets x!1,
Using template: P(x)
Instantiating quantified variables,
Q.E.D.
```

La táctica `reduce`

- **Prueba del `ej1` con `reduce`**

```
ej1 :

  |-------
{1}   FORALL x: (P(x) IMPLIES (EXISTS x: P(x)))

Rule? (reduce)
Repeatedly simplifying with decision procedures, rewriting,
  propositional reasoning, quantifier instantiation, skolemization,
  if-lifting and equality replacement,
Q.E.D.
```

Incompletitud de la táctica `reduce`

- **Conjetura** (`ej2`)**:** $(\forall x P(x)) \rightarrow (\exists x P(x))$
- **Ampliación de la teoría `lpo.pvs`**

```
ej2: THEOREM
  (FORALL x: P(x)) IMPLIES (EXISTS x: P(x))
```

- **Intento de prueba con `reduce`**

```
ej2 :
  |-------
{1}   (FORALL x: P(x)) IMPLIES (EXISTS x: P(x))

Rule? (reduce)
Repeatedly simplifying ... this simplifies to:
ej2 :
{-1}  (FORALL x: P(x))
  |-------
{1}   (EXISTS x: P(x))
```

Incompletitud de la táctica `reduce`

- **La conjetura es falsa, ya que el tipo `T` puede ser vacío**
- **Teorema (`ej3`): $(\forall x_1 P_1(x_1)) \rightarrow (\exists x_1 P_1(x_1))$, donde x_1 es una variable en un dominio T_1 no vacío y P_1 es un predicado sobre T_1**
- **Ampliación de la teoría `lpo.pvs`**

```
T1: NONEMPTY_TYPE
a1: T1
P1: [T1 -> bool]
x1: VAR T1

ej3: THEOREM
  (FORALL x1: P1(x1)) IMPLIES (EXISTS x1: P1(x1))
```

Incompletitud de la táctica `reduce`

- **Intento de prueba con `reduce`**

```
ej3 :

  |-------
{1}   (FORALL x1: P1(x1)) IMPLIES (EXISTS x1: P1(x1))

Rule? (reduce)
Repeatedly simplifying ... this simplifies to:
ej3 :

{-1}  (FORALL x1: P1(x1))
  |-------
{1}   (EXISTS x1: P1(x1))

Rule? q
Do you really want to quit?  (Y or N): y
```

Incompletitud de la táctica `reduce`

- **Prueba del `ej3` con `inst`**

```
ej3 :

  |-------
{1}   (FORALL x1: P1(x1)) IMPLIES (EXISTS x1: P1(x1))

Rule? (flatten)
Applying disjunctive simplification to flatten sequent,
this simplifies to:
ej3 :

{-1}  (FORALL x1: P1(x1))
  |-------
{1}   (EXISTS x1: P1(x1))

Rule? (inst - "a1")
Instantiating the top quantifier in - with the terms:
 a1,
this simplifies to:
```

Incompletitud de la táctica `reduce`

```
ej3 :

{-1}  P1(a1)
  |-------
[1]   (EXISTS x1: P1(x1))

Rule? (inst?)
Found substitution:
x1 gets a1,
Using template: P1(x1)
Instantiating quantified variables,
Q.E.D.
```

Incompletitud de la táctica `reduce` (II)

- **Teorema** (`ej4`) $(\exists y \in T_1)[(\forall z \in T_1)[Q(y) \rightarrow Q(z)]]$, **con** $T_1 \neq \emptyset$
- **Ampliación de la teoría** `lpo.pvs`

```
Q: [T1 -> bool]
ej4: THEOREM
  EXISTS (y:T1): FORALL (z:T1): Q(y) IMPLIES Q(z)
```

Incompletitud de la táctica `reduce` (II)

- **Intento de prueba con `reduce`**

```
ej4 :

  |-------
{1}   EXISTS (y: T1): FORALL (z: T1): Q(y) IMPLIES Q(z)

Rule? (reduce)
Repeatedly simplifying with decision procedures, rewriting,
  propositional reasoning, quantifier instantiation, skolemization,
 if-lifting and equality replacement,
this simplifies to:
ej4 :

  |-------
[1]   EXISTS (y: T1): FORALL (z: T1): Q(y) IMPLIES Q(z)
```

Incompletitud de la táctica `reduce` (II)

- **Prueba del `ej4`**

```
ej4 :
  |-------
{1}   EXISTS (y: T1): FORALL (z: T1): Q(y) IMPLIES Q(z)

Rule? (CASE "FORALL (y:T1): Q(y)")
Case splitting on FORALL (y: T1): Q(y), this yields  2 subgoals:

ej4.1 :
{-1}  FORALL (y: T1): Q(y)
  |-------
[1]   EXISTS (y: T1): FORALL (z: T1): Q(y) IMPLIES Q(z)

Rule? (inst 1 "a1")
Instantiating the top quantifier in 1 with the terms: a1,
this simplifies to:
```

Incompletitud de la táctica `reduce` (II)

```
ej4.1 :

[-1]  FORALL (y: T1): Q(y)
  |-------
{1}   FORALL (z: T1): Q(a1) IMPLIES Q(z)

Rule? (skolem!)
Skolemizing, this simplifies to:
ej4.1 :
[-1]  FORALL (y: T1): Q(y)
  |-------
{1}   Q(a1) IMPLIES Q(z!1)

Rule? (inst - "z!1")
Instantiating the top quantifier in - with the terms:  z!1,
this simplifies to:
```

Incompletitud de la táctica `reduce` (II)

```
ej4.1 :
{-1}  Q(z!1)
  |-------
[1]   Q(a1) IMPLIES Q(z!1)

Rule? (prop)
Applying propositional simplification,

This completes the proof of ej4.1.

ej4.2 :
  |-------
{1}   FORALL (y: T1): Q(y)
[2]   EXISTS (y: T1): FORALL (z: T1): Q(y) IMPLIES Q(z)

Rule? (skolem!)
```

Incompletitud de la táctica `reduce` (II)

```
Skolemizing, this simplifies to:
ej4.2 :
  |-------
{1}   Q(y!1)
[2]   EXISTS (y: T1): FORALL (z: T1): Q(y) IMPLIES Q(z)

Rule? (reduce)
Repeatedly simplifying with decision procedures, rewriting,
  propositional reasoning, quantifier instantiation, skolemization,
 if-lifting and equality replacement,

This completes the proof of ej4.2.

Q.E.D.
```

Incompletitud de la táctica `reduce` (II)

- **Comparación con OTTER**
  - **Entrada:** `ej4.in`

```
set(auto2).

formula_list(usable).
-(exists y (all z (Q(y) -> Q(z)))).
end_of_list.
```

  - **Prueba**

```
1 [] -Q($f1(x)).
2 [] Q(x).
3 [binary,2.1,1.1] $F.
```

Reglas y tácticas de la igualdad

- **Reglas de la igualdad: reflexiva, simétrica, transitiva y congruencia**
- **Tácticas de la igualdad: `replace` y `assert`**
- **Estrategia proposicional y ecuacional: `ground`**

La táctica `replace`

- **Teorema** (`ej5`)**:** $f(f(f(a))) = f(a) \rightarrow f(f(f(f(f(a))))) = f(a)$
- **Ampliación de la teoría** `lpo.pvs`

```
a: T
f: [T -> T]
ej5: THEOREM
  f(f(f(a))) = f(a) IMPLIES f(f(f(f(f(a))))) = f(a)
```

- **Prueba con** `replace`

```
ej5 :
  |-------
{1}   f(f(f(a))) = f(a) IMPLIES f(f(f(f(f(a))))) = f(a)

Rule? (flatten)
Applying disjunctive simplification to flatten sequent, this simplifies to:
```

La táctica replace

```
ej5 :

{-1}  f(f(f(a))) = f(a)
  |-------
{1}   f(f(f(f(f(a))))) = f(a)

Rule? (replace -1)
Replacing using formula -1,
this simplifies to:
ej5 :

[-1]  f(f(f(a))) = f(a)
  |-------
{1}   f(f(f(a))) = f(a)

which is trivially true.
Q.E.D.
```

La táctica assert

- **Prueba del `ej5` con `assert`**

```
ej5 :

  |-------
{1}   f(f(f(a))) = f(a) IMPLIES f(f(f(f(f(a))))) = f(a)

Rule? (flatten)
Applying disjunctive simplification to flatten sequent,
this simplifies to:
ej5 :

{-1}  f(f(f(a))) = f(a)
  |-------
{1}   f(f(f(f(f(a))))) = f(a)

Rule? (assert)
Simplifying, rewriting, and recording with decision procedures,
Q.E.D.
```

La estrategia `ground`

- **Prueba del `ej5` con `ground`**

```
ej5 :

  |-------
{1}   f(f(f(a))) = f(a) IMPLIES f(f(f(f(f(a))))) = f(a)

Rule? (ground)
Applying propositional simplification and decision procedures,
Q.E.D.
```