Razonamiento automático (2005–06)

Tema 7: Teorías de primer orden en PVS

José A. Alonso Jiménez

Grupo de Lógica Computacional
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

La teoría de grupo con definiciones

```
grupo : THEORY
 BEGIN
  T:        TYPE+
  x, y, z: VAR T
  id :      T
  *:        [T, T -> T]

  asociativa: AXIOM (x * y) * z = x * (y * z)
  identidad:  AXIOM x * id = x
  inversa:    AXIOM EXISTS y: x * y = id

  cuadrado(x): T =
    x * x

  cuadrado_de_la_identidad: LEMMA
    cuadrado(id) = id

 END grupo
```

Las tácticas `replace` y `lemma`

- Demostración de `cuadrado_de_la_identidad` con `replace` y `lemma`

```
cuadrado_de_la_identidad :

  |-------
{1}   cuadrado(id) = id

Rule? (lemma "cuadrado" ("x" "id"))
Applying cuadrado where x gets id, this simplifies to:
cuadrado_de_la_identidad :

{-1}  cuadrado(id) = id * id
  |-------
[1]   cuadrado(id) = id

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

Las tácticas `replace` y `lemma`

```
cuadrado_de_la_identidad :
[-1]  cuadrado(id) = id * id
  |-------
{1}   id * id = id

Rule? (lemma "identidad")
Applying identidad this simplifies to:
cuadrado_de_la_identidad :
{-1}  FORALL (x: T): x * id = x
[-2]  cuadrado(id) = id * id
  |-------
[1]   id * id = id

Rule? (inst?)
Found substitution: x: T gets id, Using template: x * id = x
Instantiating quantified variables,
Q.E.D.
```

La táctica `use`

- Demostración de `cuadrado_de_la_identidad` con `use` en lugar de `lemma` e `inst?`

```
cuadrado_de_la_identidad :

[-1]  cuadrado(id) = id * id
  |-------
{1}   id * id = id

Rule? (use "identidad")
Using lemma identidad,
Q.E.D.
```

Las tácticas `expand` y `rewrite`

- Demostración de `cuadrado_de_la_identidad` con `expand` y `rewrite`

```
cuadrado_de_la_identidad :

  |-------
{1}   cuadrado(id) = id

Rule? (expand "cuadrado")
Expanding the definition of cuadrado, this simplifies to:
cuadrado_de_la_identidad :

  |-------
{1}   id * id = id

Rule? (rewrite "identidad")
Found matching substitution: x: T gets id,
Rewriting using identidad, matching in *,
Q.E.D.
```

La táctica `rewrite` sobre definiciones

- Demostración de `cuadrado_de_la_identidad` con `rewrite`

```
cuadrado_de_la_identidad :

  |-------
{1}   cuadrado(id) = id

Rule? (rewrite "cuadrado")
Found matching substitution: x gets id,
Rewriting using cuadrado, matching in *, this simplifies to:
cuadrado_de_la_identidad :

  |-------
{1}   id * id = id

Rule? (rewrite "identidad")
Found matching substitution: x: T gets id,
Rewriting using identidad, matching in *,
Q.E.D.
```

Las tácticas `auto_rewrite` y `assert`

- Demostración de `cuadrado_de_la_identidad` con `auto_rewrite` y `assert`

```
cuadrado_de_la_identidad :
  |-------
{1}   cuadrado(id) = id

Rule? (auto-rewrite "cuadrado" "identidad")
Installing automatic rewrites from: cuadrado, identidad. This
simplifies to:
cuadrado_de_la_identidad :
  |-------
[1]   cuadrado(id) = id

Rule? (assert)
identidad rewrites id * id  to id
cuadrado rewrites cuadrado(id)  to id
Simplifying, rewriting, and recording with decision procedures,
Q.E.D.
```

La táctica auto_rewrite_theory

- Demostración de `cuadrado_de_la_identidad` con `auto_rewrite_theory`

```
cuadrado_de_la_identidad :
  |-------
{1}   cuadrado(id) = id

Rule? (auto-rewrite-theory "grupo")
Rewriting relative to the theory: grupo. This simplifies to:
cuadrado_de_la_identidad :
  |-------
[1]   cuadrado(id) = id

Rule? (assert)
identidad rewrites id * id to id
cuadrado rewrites cuadrado(id) to id
Simplifying, rewriting, and recording with decision procedures,
Q.E.D.
```

La táctica `grind`

- Demostración de `cuadrado_de_la_identidad` con `grind`

```
cuadrado_de_la_identidad :

  |-------
{1}   cuadrado(id) = id

Rule? (grind :theories "grupo")
identidad rewrites id * id
  to id
cuadrado rewrites cuadrado(id)
  to id
Trying repeated skolemization, instantiation, and if-lifting,
Q.E.D.
```

Extensión de teorías

- La teoría de grupos conmutativos

```
grupo_conmutativo : THEORY
 BEGIN
  IMPORTING grupo

  x, y, z: VAR T

  conmutatividad: AXIOM x * y = y * x

  identidad_izquierda: LEMMA
    id * x = x

 END grupo_conmutativo
```

La táctica `auto-rewrite!`

- Demostración de **`identidad_izquierda`** con **`auto-rewrite!`**

```
identidad_izquierda :
  |-------
{1}   FORALL (x: T): id * x = x

Rule? (auto-rewrite! "conmutatividad" "identidad")
Installing automatic rewrites from: (conmutatividad identidad),
this simplifies to:
identidad_izquierda :
  |-------
[1]   FORALL (x: T): id * x = x

Rule? (assert)
identidad rewrites x * id  to x
conmutatividad rewrites id * x  to x
Simplifying, rewriting, and recording with decision procedures,
Q.E.D.
```

Teorías parametrizadas

- Teoría de grupos parametrizada

```
grupo_parametrizado [T: TYPE+, * : [T, T -> T], id: T ] : THEORY
 BEGIN
  ASSUMING
   x, y, z: VAR T

   asociativa: ASSUMPTION (x * y) * z = x * (y * z)
   identidad:  ASSUMPTION x * id = x
   inversa:    ASSUMPTION EXISTS y: x * y = id
  ENDASSUMING

  cuadrado(x): T = x * x

  cuadrado_de_la_identidad: LEMMA cuadrado(id) = id

 END grupo_parametrizado
```

Instanciación de teorías

- Grupo aditivo de los reales

```
grupo_real_aditivo: THEORY
 BEGIN
  IMPORTING grupo_parametrizado[real, +, 0]
 END grupo_real_aditivo
```

- Condiciones de instanciación (generadas y probadas)

```
IMP_grupo_parametrizado_TCC1: OBLIGATION
  FORALL (x, y, z: real): (x + y) + z = x + (y + z);
IMP_grupo_parametrizado_TCC2: OBLIGATION
  FORALL (x: real): x + 0 = x;
IMP_grupo_parametrizado_TCC3: OBLIGATION
  FORALL (x: real): EXISTS (y: real): x + y = 0;
```

Bibliografía

- M. Hofmann *Razonamiento asistido por computadora (2001–02)*
- N. Shankar *Mechanized verification methodologies*