Apéndice B

Software para Validación de Resultados

Una técnica muy utilizada en la actualidad es dada una propiedad de un objeto se realiza un "testing" de dicha propiedad para verificar la veracidad del enunciado. Si se llega a encontrar un contraejemplo es claro que todo intento de demostración formal sería en vano. Cuando los recursos computacionales no encuentran un contraejemplo existe la posibilidad que el enunciado sea cierto. Al tener una conjetura se procede a encontrar una prueba formal. La tarea anteriormente mencionada se puede compartir con un demostrador automático ya que si éste encuentra la prueba sólo es cuestión de verificar lo obtenido por la máquina.

El procedimiento anteriormente descrito fue ampliamente utilizado en muchas ocasiones en el trabajo de tesis. En este apéndice se muestra el código fuente de los programas utilizados en la investigación de tesis. Los programas utilizados son Clasp y Prover9.

B.1. Clasp

Clasp es un *Answer Set Programming* Solver introducido en [13], el cual computa los answer sets de programas lógicos. En [20] se muestra una aplicación de clasp para mostrar la existencia de una única lógica trivaluada paraconsistente. Todas las tablas generadas en la presente tesis fueran generadas con la ayuda del conjunto de instruc-

ciones presentados en [20]. Como ejemplo en el siguiente programa se encuentra una valuación en C_1 el cual exhibe que $\nvdash_{C_1} \neg(\alpha \vee \beta) \leftrightarrow (\neg\alpha \wedge \neg\beta)$.

```
1  % Experimentos para logicas sobre Cw caso particular C1 proposicional
       de Newton
2  % Autor: Mauricio Osorio Septimbre 2013
3  % Corre en clasp
4
5  % Dominio de Valuaciones
6  v(0..2).
7
8  % Valores Designados
9  sel(1..2).
10
11 %% Definicion general de conectivos primitivos
12 1 { impl(X, Y, Z) : v(Z) } 1 :- v(X), v(Y).
13 1 { and(X, Y, Z) : v(Z) }  1 :- v(X), v(Y).
14 1 { or(X, Y, Z) : v(Z) } 1 :- v(X), v(Y).
15 1 { neg(X, Y) : v(Y) } 1 :- v(X).
16
17 % Conectivos no primitivos
18 equ(X,Y,Z) :- impl(X,Y,L), impl(Y,X,R), and(L,R,Z).
19
20 % Bolita
21 bl(X, Z) :- neg(X,X1), and(X,X1,Y), neg(Y,Z).
22
23 % Reglas de inferencia MP
24 :- impl(X,Y,Z), sel(X), sel(Z), v(Y), not sel(Y).
25
26 % Axiomas positivos
27
28 % Axioma P1
29 :- impl(Y,X,Z), impl(X,Z,W), v(W), not sel(W).
30
31 % Axioma P2
32 :- impl(Y,Z,W1), impl(X,W1,L), impl(X,Y,W3), impl(X,Z,W4), impl(W3,W4,R
       ),
33    impl(L,R,Q), not sel(Q).
34
35 % Axioma P3
```

```
36 :- and(X,Y,Z), impl(Z,X,W), v(W), not sel(W).
37
38 %Axioma P4
39 :- and(X,Y,Z), impl(Z,Y,W), v(W), not sel(W).
40
41 %Axioma P5
42 :- and(X,Y,Z), impl(Y,Z,W), impl(X,W,Q), v(Q), not sel(Q).
43
44 %Axioma P6
45 :- or(X,Y,Z), impl(X,Z,W), v(W), not sel(W).
46
47 %Axioma P7
48 :- or(X,Y,Z), impl(Y,Z,W), v(W), not sel(W).
49
50 %Axioma P8
51 :- impl(X,Z,L), impl(Y,Z,W2), or(X,Y,W3), impl(W3,Z,W4), impl(W2, W4,R)
       ,
52  impl(L,R,Q), v(Q), not sel(Q).
53
54 %Axiomas Cw
55
56 :- neg(X,X1), or(X,X1,Q), v(Q), not sel(Q).
57 :- neg(X,X1), neg(X1,X2), impl(X2,X,Q), v(Q), not sel(Q).
58
59 %Axiomas propios de C1
60
61 %B^o -> ((A -> B) -> ((A -> \neg B) -> \neg A))
62 :- impl(A,B,X1), neg(B,B1), neg(A,A1), impl(A,B1,X2),impl(X2, A1,X3),
63    impl(X1,X3,Y), bl(B,Bbl), impl(Bbl,Y,Q), v(Q), not sel(Q).
64
65 %(A^o \land B^o) -> (A -> B)^o
66 :- impl(A,B,Y), bl(Y,Ybl), bl(A,Abl), bl(B,Bbl), and(Abl, Bbl,X), impl(
       X,Ybl,Q),
67   v(Q), not sel(Q).
68
69 %(A^o \land B^o) -> (A \land B)^o
70 :- and(A,B,Y), bl(Y,Ybl), bl(A,Abl), bl(B,Bbl), and(Abl, Bbl,X), impl(X
       ,Ybl,Q),
71   v(Q), not sel(Q).
```

```
72
73 %(A^o \land B^o) -> (A \lor B)^o
74 :- or(A,B,Y), bl(Y,Ybl), bl(A,Abl), bl(B,Bbl), and(Abl, Bbl,X), impl(X,
       Ybl,Q),
75   v(Q), not sel(Q).
76
77 %Constraint de C1
78 %-X -> (X -> Y)
79 fu1(W) :- neg(X,X1), impl(X,Y,R), impl(X1,R,W).
80
81 %-----------------------------------------------------------------
82 %Teorema: EN C1 no se cumple que |- ~(a v b) <-> (~a ^ ~b)
83 %-----------------------------------------------------------------
84 teo(C1) :- or(A, B, AB1), neg(AB1, NAB1), neg(A, NA), neg(B, NB), and(
       NA, NB, AB2), equ(NAB1, AB2, C1), v(C1).
85
86 :- not fu1(X) : not sel(X) : v(X).
87 :- not teo(X) : not sel(X) : v(X).
88
89 #hide v/1.
90 #hide sel/1.
91 #hide teo/1.
92 #hide equ/3.
93 #hide fu1/1.
94 #hide bl/2.
```

```
➜ Avance de Tesis  clingo C1
Answer: 1
impl(2,2,2) impl(2,1,1) impl(2,0,0) impl(1,2,1) impl(1,1,1) impl(1,0,0) impl(0,2
,2) impl(0,1,1) impl(0,0,2) and(2,2,2) and(2,1,1) and(2,0,0) and(1,2,1) and(1,1,
2) and(1,0,0) and(0,2,0) and(0,1,0) and(0,0,0) or(2,2,2) or(2,1,1) or(2,0,2) or(
1,2,1) or(1,1,1) or(1,0,1) or(0,2,2) or(0,1,1) or(0,0,0) neg(2,0) neg(1,1) neg(0
,2)
SATISFIABLE

Models       : 1+
Time         : 0.220
  Prepare    : 0.130
  Prepro.    : 0.090
  Solving    : 0.000
```

Figura 1. Ejecución del programa de Clasp.

B.2. Prover9

Prover9 es un demostrador automático de teoremas para lógica de primer orden con y sin igualdad [17]. El uso de este software fue como herramienta auxiliar en las demostraciones de tesis. Como muestra del uso se enseña en el siguiente código la lógica proposicional de C_1 para demostrar $\vdash_{C_1} \alpha \to \alpha$.

```
formulas(sos).

 % Positive Axioms ---------------------------------------------------
 % A1
 P( impl(x, impl(y, x)) ).
 % A2
 P( impl(impl(x, y), impl(impl(x, impl(y, z)), impl(x, z))) ).
 % A3
 P( impl(and(x, y), x) ).
 % A4
 P( impl(and(x, y), y) ).
 % A5
 P( impl(x, impl(y, and(x, y))) ).
 % A6
 P( impl(x, or(x, y)) ).
 % A7
 P( impl(y, or(x, y)) ).
 % A8
 P( impl(impl(x, z), impl(impl(y, z), impl(or(x, y), z))) ).
 % Positive Axioms ---------------------------------------------------

 % A9
 P( impl(neg(and(y, neg(y))), impl(impl(x, y), impl(impl(x, neg(y)),
   neg(x)))) ).

 % A10
 % o_and
 P( impl(and(neg(and(x, neg(x))), neg(and(y, neg(y)))), neg(and(and(x,
    y), neg(and(x, y))))) ).
 % o_or
 P( impl(and(neg(and(x, neg(x))), neg(and(y, neg(y)))), neg(and(or(x,
   y), neg(or(x, y))))) ).
```

```
30   %o_impl
31   P( impl(and(neg(and(x, neg(x))), neg(and(y, neg(y)))), neg(and(impl(x
       , y), neg(impl(x, y))))) ).
32
33   %A11
34   P( or(x, neg(x)) ).
35   %A12
36   P( impl(neg(neg(x)), x) ).
37
38   -P(impl(x,y)) | -P(x) | P(y).
39
40 end_of_list.
41
42 formulas(goals).
43
44   P( impl(x, x) ).
45
46 end_of_list.
```

```
➜ Avance de Tesis  prover9 -f C1.in | prooftrans
============================== prooftrans ============================
Prover9 (64) version 2009-11A, November 2009.
Process 833 was started by joseabelcastellanosjoo on Z,
Sun Oct 26 20:10:44 2014
The command was "prover9 -f C1.in".
============================== end of head ===========================

============================== end of input ==========================
-------- Proof 1 --------

THEOREM PROVED

------ process 833 exit (max_proofs) ------

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.10 (+ 0.01) seconds.
% Length of proof is 9.
% Level of proof is 3.
% Maximum clause weight is 14.000.
% Given clauses 207.


1 P(impl(x,x)) # label(non_clause) # label(goal).  [goal].
2 P(impl(x,impl(y,x))).  [assumption].
3 P(impl(impl(x,y),impl(impl(x,impl(y,z)),impl(x,z)))).  [assumption].
8 P(impl(x,or(y,x))).  [assumption].
16 -P(impl(x,y)) | -P(x) | P(y).  [assumption].
17 -P(impl(c1,c1)).  [deny(1)].
79 P(impl(impl(x,impl(or(y,x),z)),impl(x,z))).  [hyper(16,a,3,a,b,8,a)].
4585 P(impl(x,x)).  [hyper(16,a,79,a,b,2,a)].
4586 $F.  [resolve(4585,a,17,a)].

============================== end of proof ==========================
```

Figura 2. Ejecución del programa de Prover9.