Stable Model Semantics: From Theory to Implementations and Applications

CL2000 — Tutorial

Ilkka Niemelä
Helsinki University of Technology
Department of Computer Science and Engineering
Laboratory for Theoretical Computer Science
P.O.Box 5400, FIN-02015 HUT, Finland
`http://www.tcs.hut.fi/~ini/`
e-mail: `Ilkka.Niemela@hut.fi`

1 Overview

This handout contains supporting material for the CL2000 tutorial on stable model semantics. It consists of:

- information on how to obtain the SMODELS software discussed in the tutorial.
- examples on running SMODELS
- encoding of blocks world planning using SMODELS
- references to additional material
- complete set of slides

2 Where to get SMODELS?

SMODELS has been developed in Helsinki University of Technology by a group led by Dr. Ilkka Niemelä. For running SMODELS you need to obtained the latest versions of two programs (`smodels`+`lparse`) which can be down loaded at `http://www.tcs.hut.fi/Software/smodels/` . For further information on how to use the system, consult the `lparse` user's manual.

3 Running SMODELS

- Typical Unix command line:

```
UNIX> lparse [options] files | smodels [number]
```

- Some options:

```
-c const=n : define 'const' as number 'n'
-d : domain predicates to be output
-t : output plain text
number determines the number of stable models computed (0 indicates all)
```

- Computing all stable models of `demo.lp`:

```
UNIX> lparse demo.lp | smodels 0
smodels version 2.25. Reading...done
Answer: 1
Stable Model: r(a) p(a) s(a,b,c)
Answer: 2
Stable Model: r(a) q(a) s(a,b,c)
False
```

```
demo.lp:
s(a,b,c).
s(X,Y,Z):- t(Y,X,Z).
p(X):- not q(X), s(X,Y,Z).
q(X):- not p(X), s(X,Y,Z).
r(X):- p(X), s(X,Y,Z).
r(X):- q(X), s(X,Y,Z).
```

- Having a look at the grounding of `demo.lp` in plain text form with all evaluated domain predicates removed:

```
UNIX> lparse -d none -t demo.lp
r(a) :- p(a).
r(a) :- q(a).
q(a) :- not p(a).
p(a) :- not q(a).
```

4 Blocks world planning

```
% (operator moveop
% (params (<X> OBJECT) (<Y> OBJECT) )
% (preconds  (clear <X>) (clear <Y>))
% (effects   (on <X> <Y>)  (clear <X>)))

% We consider plans executable in at most n steps where  multiple
% non-conflicting instances of moveop can be executed in a single step.
%
% A valid plan is given by a stable model:
% facts moveop(x,y,t) in a model ordered by the argument t
% where facts with the same t can be taken in any linear order.

% An example
% anomaly %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                          %                 b
block(a).                 % a               c
block(b).                 % b  c            a
block(c).                 % ---------
                          % initially      goal
% Initial conditions
on(a,b,0).
on(b,table,0).
on(c,table,0).

% Goal conditions
goal(T) :- time(T),
on(b,c,T),
on(c,a,T).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A solution: moveop(a,table,0), moveop(c,a,1), moveop(b,c,2)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% The actual encoding
% can be used with different initial and goal conditions
% bw.lp %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Available situations 0..n
time(0..n).

nextstate(Y,X) :- time(X), time(Y),
        Y = X + 1.

% Auxiliary predicates
object(table).
object(X) :- block(X).

covered(X,T) :- block(Z), block(X), time(T),
        on(Z,X,T).

on_something(X,T) :- block(X), object(Z), time(T),
        on(X,Z,T).

available(table,T) :- time(T).
available(X,T) :- block(X), time(T),
        on_something(X,T).

% Generator
% An operator can be applied if preconditions hold
{ moveop(X,Y,T) } :- time(T), block(X), object(Y),
        X != Y, on_something(X,T),
        available(Y,T),
        not covered(X,T),
        not covered(Y,T).
```

```
% Operator effects:
on(X,Y,T2) :- block(X), object(Y), nextstate(T2,T1),
        moveop(X,Y,T1).

% Frame axiom (as a rule with exceptions):
on(X,Y,T2) :- block(X), object(Y), nextstate(T2,T1),
        on(X,Y,T1),
        not moving(X,T1).

% The exceptions
moving(X,T) :-  time(T), block(X), object(Y),
        moveop(X,Y,T).

% Eliminating the possibility to execute conflicting instances of moveop
% in the same step.

% A block cannot be moved to two destinations
:- 2 { moveop(X,Y,T):object(Y) },
       block(X), time(T).
% The destination cannot be moving
:- block(X), object(Y), time(T),
        moveop(X,Y,T),
        moving(Y,T).
% No two blocks moved onto the same block
:- 2 { moveop(X,Y,T):block(X) },
       block(Y), time(T).

% Tester
% excludes models where the goal has not been reached.
:- not goal.
goal :- time(T), goal(T).
goal(T2) :- nextstate(T2,T1), goal(T1).
```

```
% optimizations

% Stop when the goal has been reached
:- block(X), object(Y), time(T),
        moveop(X,Y,T),
        goal(T).

% No move from table to table
:- block(X), time(T),
       moveop(X,table,T), on(X,table,T).

% No move on something and then to table
:- nextstate(T2,T1), block(X), object(Y),
        moveop(X,Y,T1), moveop(X,table,T2).

% Hiding from the output everything else but moveop
hide object(_).
hide block(_).
hide time(_).
hide on_something(_,_).
hide ne_tstate(_,_).
hide on(_,_,_).
hide available(_,_).
hide covered(_,_).
hide moving(_,_).
hide goal(_).
hide goal.


% Typical command line
% lparse -d none -c n=3 bw.lp anomaly | smodels
```

References

[AAdR99] Rafael Accorsi, Carlos Areces, and Maarten de Rijke. Towards feature interaction via stable models. In *Proceedings of the Second Workshop on Formal Methods*, Florianpolis, Brasil, October 1999.

[ABS00] T. Aura, M. Bishop, and D. Sniegowski. Analyzing single-server network inhibition. In *Proceedings of teh IEEE Computer Security Foundations Workshop*, Cambridge, UK, July 2000. IEEE Computer Society Press.

[DNK97] Y. Dimopoulos, B. Nebel, and J. Koehler. Encoding planning problems in non-monotonic logic programs. In *Proceedings of the Fourth European Conference on Planning*, pages 169–181, Toulouse, France, September 1997. Springer-Verlag.

[ET00] D. East and M. Truszczyński. DATALOG with Constraints. In *Proccedings of the Seventeenth National Conference on Artificial Intelligence(AAAI-2000)*, July 2000.

[ELM+98] Thomas Eiter, Nicola Leone, Cristinel Mateis, Gerald Pfeifer, and Francesco Scarnello. The KR system dlv: Progress report, comparisons and benchmarks. In *Proceedings of the 6th International Conference on Principles of Knowledge Representation and Reasoning*, pages 406–417, Trento, Italy, June 1998. Morgan Kaufmann Publishers.

[Hel99] K. Heljanko. Using logic programs with stable model semantics to solve deadlock and reachability problems for 1-safe Petri nets. In *Proceedings of the 5th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 240–254, Amsterdam, the Netherlands, March 1999. Springer-Verlag.

[HMN00] M. Hietalahti, F. Massacci, and I. Niemelä. DES: a challenge problem for nonmonotonic reasoning systems. In *Proceedings of the 8th International Workshop on Non-Monotonic Reasoning*, Breckenridge, Colorado, USA, April 2000. cs.AI/0003039.

[JNSY00] T. Janhunen, I. Niemelä, P. Simons, and J. You. Unfolding partiality and disjunctions in stable model semantics. In A.G. Cohn, F. Guinchiglia, and B. Selman, editors, *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning*, Breckenridge, Colorado, USA, April 2000. Morgan Kaufmann Publishers. Accepted for publication.

[Lif99] V. Lifschitz. Answer set planning. In *Proceedings of the 16th International Conference on Logic Programming*, pages 25–37, Las Cruces, New Mexico, December 1999. The MIT Press.

[Lif00] V. Lifschitz. Missionaries and cannibals in the causal calculator, A.G. Cohn, F. Giunchiglia and B. Selman", In *Principles of Knowledge Representation and Reasoning, Proceedings of the Seventh International Conference (KR2000)*, April 2000, pp. 85–96. Morgan Kaufmann Publishers.

[LRS98] Xinxin Liu, C.R. Ramakrishnan, and Scott A. Smolka. Fully local and efficient evaluation of alternating fixed points. In Bernhard Steffen, editor, *Proceedings of the 4th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 5–19, Lisbon, Portugal, March/April 1998. Springer-Verlag.

[MT99] Marek, W., and Truszczyński, M. Stable models and an alternative logic programming paradigm. In Apt, K.; Marek, W.; Truszczyński, M.; and Warren, D., eds., *The Logic Programming Paradigm: a 25-Year Perspective*. Springer Verlag, 1999. 375–398.

[Nie99] I. Niemelä. Logic programming with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, 25(3,4):241–273, 1999.

[NS96] I. Niemelä and P. Simons. Efficient implementation of the well-founded and stable model semantics. In M. Maher, editor, *Proceedings of the Joint International Conference and Symposium on Logic Programming*, pages 289–303, Bonn, Germany, September 1996. The MIT Press.

[NS97] I. Niemelä and P. Simons. Smodels – an implementation of the stable model and well-founded semantics for normal logic programs. In *Proceedings of the 4th International Conference on Logic Programming and Non-Monotonic Reasoning*, pages 420–429, Dagstuhl, Germany, July 1997. Springer-Verlag.

[NSS99] I. Niemelä, P. Simons, and T. Soininen. Stable model semantics of weight constraint rules. In *Proceedings of the 5th International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 317–331, El Paso, Texas, USA, December 1999. Springer-Verlag.

[Sim97] P. Simons. Towards constraint satisfaction through logic programs and the stable model semantics. Research report A47, Helsinki University of Technology, Helsinki, Finland, August 1997. Available at http://www.tcs.hut.fi/pub/reports/A47.ps.gz.

[Sim99] P. Simons. Extending the stable model semantics with more expressive rules. In *Proceedings of the 5th International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 305–316, El Paso, Texas, USA, December 1999. Springer-Verlag.

[Sim00] P. Simons. Extending and implementing the stable model semantics. Doctoral Dissertation. Research report A58, Helsinki University of Technology, Helsinki, Finland, April 2000.

[SN99] T. Soininen and I. Niemelä. Developing a declarative rule language for applications in product configuration. In Gopal Gupta, editor, *Proceedings of the First International Workshop on Practical Aspects of Declarative Languages*, pages 305–319, San Antonio, Texas, January 1999. Springer-Verlag.

[SNG99] T. Soininen, I. Niemelä, and E. Gelle. A fixpoint definition of dynamic constraint satisfaction. In Joxan Jaffar, editor, *Proceedings of the Fifth International Conference on Principles and Practice of Constraint Programming*, pages 419–433, Alexandria, Virginia, USA, October 1999. Springer-Verlag.

[Syr99] T. Syrjänen. A rule-based formal model for software configuration. Research report A55, Helsinki University of Technology, Laboratory for Theoretical Computer Science, Helsinki, Finland, December 1999.

[Var99] K. Varpaaniemi. Stable models for stubborn sets. In H.-D. Burkhard, L. Czaja, H.S. Nguyen, and P.H. Starke, editors, *Concurrency, Specification and Programming (Proceedings of the CS&P'99 Workshop)*, pages 263–274, Warsaw, Poland, 1999.