Revised Stable Models – A Semantics for Logic Programs

Luís Moniz Pereira and Alexandre Miguel Pinto

Centro de Inteligência Artificial – CENTRIA,
Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
{lmp, amp}@di.fct.unl.pt

Abstract. This paper introduces an original 2-valued semantics for Normal Logic Programs (NLP), which conservatively extends the Stable Model semantics (SM) to all normal programs. The distinction consists in the revision of one feature of SM, namely its treatment of odd loops, and of infinitely long support chains, over default negation. This single revised aspect, addressed by means of a *Reductio ad Absurdum* approach, affords a number of fruitful consequences, namely regarding existence, relevance and top-down querying, cumulativity, and implementation.

The paper motivates and defines the Revised Stable Models semantics (rSM), justifying and exemplifying it. Properties of rSM are given and contrasted with those of SM. Furthermore, these results apply to SM whenever odd loops and infinitely long chains over negation are absent, thereby establishing significant, not previously known, properties of SM. Conclusions, further work, terminate the paper.

Keywords: Logic program semantics, Stable Models, *Reductio ad absurdum.*

1 Introduction

The paper introduces a new 2-valued semantics for Normal Logic Programs (NLP), called Revised Stable Models semantics (rSM), cogent in the properties it enjoys. Its name intends to draw attention to being inspired by, and actually revising, Stable Model semantics (SM) [2]. Indeed SMs are just particular rSM models, and the definition of the SM is a specific instance or specialization of the rSM one. But its name also draws attention to that the definitional distinction between the two consists in the *revision* of one feature of SM, namely its treatment of odd loops over negation, as well as of infinite support chains over negation. Finally, this single revised aspect is addressed by means of a *Reductio ad Absurdum* approach, a form of belief *revision*, and affords a number of fruitful consequences, not shared by SM, the present 'de facto' standard for 2-valued semantics for NLP.

For one, rSM are guaranteed to *exist* for all NLP. The concrete examples below show that odd loops may be required to model knowledge. Moreover, this guarantee is crucial in program composition (say from knowledge originating in divers sources) so that the result has a semantics. It is also important to warrant the existence of semantics after external updating, or even SM based self-updating languages [1]. Two, rSM is *relevant*, meaning that there may exist purely top-down, program call-graph based, query driven methods to determine whether a literal belongs to some model or other. These methods can thus simply return a partial model, guaranteed

C. Bento, A. Cardoso, and G. Dias (Eds.): EPIA 2005, LNAI 3808, pp. 29 – 42, 2005.

extendable to a complete one, there existing no need to compute all models or even to complete models in order to answer a query. Relevance is also crucial for modelling abduction, it being query driven. Three, rSM is cumulative (and two kinds of cumulativity will be considered), so that lemmas may be stored and reused. These and other properties shall be examined in the sequel. These results apply to SM whenever odd loops over negation (OLONs) and infinitely long chains over default negation (ICONs), are absent, thereby establishing significant, not previously known, properties of SM.

Odd Loops Over Negation (OLONs)
In SM, programs such as a ← ~a, where '~' stands for default negation, do not have a model. One can easily perceive that the Odd Loop Over Negation is the trouble-maker. The single rSM model however is {a}. The reason is that if assuming '~a' leads to an inconsistency, namely by implying 'a', then in a 2-valued semantics 'a' should be true instead by *Reductio ad Absurdum*.

Example 1: The president of Morelandia is considering invading another country. He reasons thus: if I do not invade them now they are sure to deploy Weapons of Mass Destruction (WMD) sometime; on the other hand, if they shall deploy WMD I should invade them now. This is coded by his analysts as:

deploy_WMD ← ~ invade_now invade_now ← deploy_WMD

Under the SM semantics this program has no models. Under the rSM semantics invasion is warranted by the single model M={invade_now}, and no WMD will be deployed.

It is an apparently counter-intuitive idea to permit such loops to support a literal's truth value, because it means the truth of the literal is being supported on its own negation, and this seems self-inconsistent. SM does not go a long way in treating such OLON. It simply decrees there is no model (throwing out the baby along with the bath water), instead of opting for taking the next logical step: reasoning by absurdity or *Reductio ad Absurdum* (RAA). That is, if assuming a literal false (i.e. its default negation is true) leads to an inconsistency, then, in a 2-valued semantics, the literal must be true if that's consistent. SM does not do this – it requires a true literal to be supported by its rules, i.e. by a rule with true body. The solution proffered by rSM is to extend the notion of support to include reasoning by absurdity, i.e. one supported indeed by those rules creating the odd loop. That is why the single rSM of a ← ~a is {a}.

Example 2: During elections, the prime minister of Italisconia promises to lower taxes as soon as possible, justifying it as inevitable. Indeed, if taxes are not lowered the rich do not invest, the economy cools down, and the country is all the poorer. People thus cannot afford to pay taxes, and these must be lowered anyway:

no_investment ← ~ lower_taxes cool_economy ← no_investment
unaffordable_taxes ← cool_economy lower_taxes ← unaffordable_taxes

Under SM this program has no models. Under rSM lowering taxes is warranted by the single model M={lower_taxes}, and the economy does not cool, etc. These two examples are typical of political *reductio ad absurdum* inevitability arguments.