Towards a Declarative Approach to Model Human Reasoning with Nonmonotonic Logics

Christoph Wernhard
Technische Universität Dresden
christoph.wernhard@tu-dresden.de

Abstract. Stenning and van Lambalgen introduced an approach to model empirically studied human reasoning with nonmonotonic logics. Some of the research questions that have been brought up in this context concern the interplay of the open- and closed-world assumption, the suitability of particular logic programming semantics for the modeling of human reasoning, and the role of three-valued logic programming semantics and three-valued logics. We look into these questions from the view of a framework where logic programs that model human reasoning are represented declaratively and are mechanizable by classical formulas extended with certain second-order operators.

1 Introduction

When humans are presented with reasoning tasks, typical "fallacies" can be observed, conclusions that are not sound in a naive classical logic understanding. Such patterns of human reasoning are researched in cognitive science, with [1] being a landmark work. Recently, an approach to model such empirical observations by means of logic programs has been presented [18, 19]. It involves a two-step process, first the construction of a logic program from, for example, natural language statements and the choice of a logic programming semantics. Second, the actual reasoning, straightforwardly performed with respect to the program and chosen semantics. For modeling the findings from [1], a variant of the three-valued Fitting operator semantics had been originally suggested in [18] and corrected and developed further in [6–8, 2]. The following research questions were brought up, among others, by these works:

Q1: A particular variant of the Fitting operator semantics, along with a related variant of predicate completion, is required to model the results from [1]. What exactly are the roles of these variants? Are there analogue variants of other logic programming semantics?
Q2: Which logic programming semantics can be applied to model human reasoning according to the approach of [19]?
Q3: What are the roles of three-valued logic programming semantics and three-valued logics in the modeling of human reasoning?

We approach these questions here from a particular point of view where the logic programs that model human reasoning are represented by formulas of classical logic, extended by second-order operators for specific application patterns of predicate quantification [22]. A particular such pattern is predicate circumscription [14, 11], which allows to express formulas that have as models only those models of the argument formula which are minimal with respect to the extensions of a given set of predicates. For a formula with second-order operators, in many cases an equivalent formula without such operators can be computed by

eliminating these operators. With this approach, the relations between the nonmonotonic logic programming semantics and classical logic are established not by syntactic transformations, but by semantically defined second-order operators.

Typically, in logic programming the meaning of a predicate occurrence depends on its syntactic context, that is, whether it is in the head or in the body of a rule, or is subject to negation as failure. In our classical representations of programs, we distinguish these meanings by letting each original predicate p correspond to several predicates $p^0, p^1, \ldots$, one for each relevant such non-classical context. With this framework, several established logic programming semantics can be characterized as patterns in which circumscription and other second-order operators are applied, based on characterizations by means of syntactic translations, in particular, of predicate completion in terms of circumscription [10], stable models semantics in terms of circumscription [13, 12], and partial stable models semantics [16] in terms of two-valued stable models [9]. Consider for example the following normal logic program:

$$\begin{array}{l} l \leftarrow e,\ \mathsf{not}\ ab. \\ e \leftarrow \mathsf{true}. \end{array} \qquad \text{(i)}$$

It can be represented by the classical propositional formula $(l^0 \leftarrow e^0 \wedge \neg ab^1) \wedge e^0$, where its stable model semantics can be rendered by the second-order formula

$$\mathsf{rename}_{1\backslash 0}(\mathsf{circ}_{(0\cap \mathsf{POS})\cup 1}((l^0 \leftarrow e^0 \wedge \neg ab^1) \wedge e^0)). \qquad \text{(ii)}$$

The second-order operator $\mathsf{rename}_{1\backslash 0}$ expresses systematic renaming of predicates with superscript 1 to their counterparts with superscript 0. The other second-order operator $\mathsf{circ}_{(0\cap \mathsf{POS})\cup 1}$ expresses parallel predicate circumscription [11] of the predicates with superscript 0 while leaving predicates with superscript 1 fixed. (See [20] for precise specifications of these operators.) Eliminating the second-order operators in (ii) yields the classical propositional formula $(e^0 \wedge l^0 \wedge \neg ab^0)$ which is equivalent to (ii) and whose models correspond to the stable models of the original program (i), that is, the single stable model $\{e, l\}$.

Envisaged benefits of this approach, in particular in the context of modeling human reasoning, include the following:

- The second-order formulas that express logic programs which in turn model human reasoning tasks provide a *declarative view of human reasoning*. With these formulas, different operational methods to construct them and to perform reasoning can be associated, in analogy to calculi. Calculi follow different paradigms, such as "model-based" versus "rule-based". Similarly, different paradigms in human reasoning such as "mental models" versus "rules" or neural approaches can be related to a single declarative representation.
- The framework allows *mechanization at all levels*, not just execution of the logic program in a way that simulates human reasoning. Also the metalevel of the characterizations of nonmonotonic semantics is mechanizable. Automated deduction systems can be applied to reason about features of human reasoning and to systematize them.
- Humans apply different ways of reasoning. The proposed approach allows to take this into account by allowing to express *different nonmonotonic semantics within a single framework*, where they can be used together. In addition, features like disjunctive heads, negation as failure in the head and first-order quantification can straightforwardly be incorporated.

– With the proposed approach, nonmonotonic semantics are in essence *reduced to combinations of a few general operators and principles*, like circumscription applied to predicate occurrences. It may be of interest to investigate whether patterns at this level of combination of general operators match observed patterns of human reasoning.

The rest of the paper is organized around the three mentioned research questions: Question Q1 is addressed in Section 2, *Integration of Open- and Closed-World Reasoning*, question Q2 in Section 3, *Logic Programming Semantics for Modeling Human Reasoning*, and question Q3 in Sections 4 and 5 about three-valued logic programming semantics and three-valued logics, respectively, for modeling human reasoning.

2 Integration of Open- and Closed-World Reasoning

According to the approach of [19], it is essential for the application of logic programming to model human reasoning that some predicates are subject to the closed-world assumption and others to the open-world assumption. With predicate circumscription, selective closed-world reasoning can be expressed naturally by specifying which predicates are to be minimized and which are fixed. Other nonmonotonic semantics have to be generalized such that they allow selective closed-word reasoning. Consider the following program, which models the two conditionals *"if she has an essay to write she will study in the library"* and *"if she has a textbook to read she will study in the library"* from [1] according to [19], where l stands for *"she will study in the library"*, e for *"she has an essay to write"*, and t for *"she has a textbook to read"*.

$$\begin{array}{l} l \leftarrow e,\ \mathsf{not}\ ab_1. \\ l \leftarrow t,\ \mathsf{not}\ ab_2. \end{array} \qquad \text{(iii)}$$

According to [19], abnormality predicates (ab_1, ab_2) and predicates occurring in a rule head (l) are considered closed-world, and the remaining predicates (e, t) open-world. Now, the negative fact *"she does not have an essay to write"* is added to the human reasoning scenario. Thus e should be set to false. Since this can not be expressed in the syntax of normal logic programs, in [19] a special syntax for negative facts is provided that allows to write $e \leftarrow \mathsf{false}$. If we want to stay within normal logic programs, the negated e can be expressed simply by considering e as subject to the closed-world assumption, leaving just t open-world. The stable model of program (iii) with respect to t considered as open-world can be expressed by the following second-order formula:

$$\mathsf{rename}_{1\backslash 0}(\mathsf{circ}_{(0\cap \mathsf{POS})\cup 1\cup\{t^0\}}((l^0 \leftarrow e^0 \wedge \neg ab_1^1) \wedge (l^0 \leftarrow t^0 \wedge \neg ab_2^1))). \qquad \text{(iv)}$$

The open-world predicate t is passed as parameter to the circumscription operator to the effect that t is considered with respect to the circumscription as fixed. Elimination of the second-order operators in (iv) yields

$$(l^0 \wedge t^0 \wedge \neg e^0 \wedge \neg ab_1^0 \wedge \neg ab_2^0) \vee (\neg l^0 \wedge \neg t^0 \wedge \neg e^0 \wedge \neg ab_1^0 \wedge \neg ab_2^0), \qquad \text{(v)}$$

corresponding to two stable models: $\{l, t\}$ and $\{\}$. Thus $\neg l$, is not a consequence of program (iii) under stable models semantics with t considered open-world, matching the empiric results reported in [1], where in that scenario only 5% of the subjects conclude *she will not study in the library.*

Further logic programming semantics can be generalized to take open-world predicates into account analogously to the stable models semantics. This holds in particular for the supported models semantics and for three-valued generalizations of these two-valued semantics: the partial stable models semantics [16], the related well-founded semantics, and semantics based on the Fitting operator [4]. In all cases, the open-world predicates are passed as fixed predicates to an occurrence of the circumscription operator [20].

There are alternate ways of expressing the incorporation of the required open-world reasoning into logic programs: In [19], a variant of predicate completion is used, called *weak completion* in [6], where predicates that do not occur in a head are exempt from completion. This works for the completion based semantics such as supported models and the three-valued semantics based on the Fitting operator, however it is not straightforwardly adaptable to rules with first-order variables [20]. Another possibility is to use the standard versions of the logic programming semantics and extended programs by special rules to encode that certain predicates are open-world: For the completion based semantics this can be achieved by adding $p \leftarrow p$ for each open-world predicate p. For the stable and the partial stable models semantics, and for the completion based semantics as well, this can be achieved by adding two rules $p \leftarrow \neg not\_p$ and $not\_p \leftarrow \neg p$, where not_p is a fresh predicate, for each open-world predicate p. Existential predicate quantification can be applied to the newly generated $not\_$ predicates, such that they do not occur in the final results. For all these variants, equivalence to the circumscription based representation can be shown [20].

3 Logic Programming Semantics for Modeling Human Reasoning

In the literature, so far only a single logic programming semantics has been investigated in the context of modeling human reasoning according to [19]: The least fixed point of the Fitting operator (modified to take open-world predicates into account) and its rendering as least model of the program completion viewed as formula in a three-valued logic. The program representation of a human reasoning scenario is considered adequate with respect to the empiric results if certain conclusions drawn or not drawn by a significant number of human subjects correspond to facts that are entailed or not entailed, respectively, by the program, like $\neg l$ in the example above.

In Section 2 we already have seen an example that has been evaluated with a different logic programming semantics, the two-valued stable models semantics, generalized to take open-world predicates into account. It can be shown that for the scenarios from [1] studied in [19] with this semantics the same adequacy results as for the three-valued Fitting operator based semantics can be obtained. For the logic programs according to [19], the supported models semantics, again generalized to take open-world predicates into account, yields exactly the same models as the stable models semantics. Considering three-valued semantics, the adequacy results obtained for the Fitting operator based semantics can also be obtained with the partial stable models semantics and the well-founded semantics, when these are generalized to take open-world predicates into account. These correspondences have been shown for "forward-reasoning" tasks, that is,

modus ponens and *denial of the antecedent* in [20]. Combining this with results from [8, 21], they are expected also to hold for abduction based "backward reasoning", that is, *modus tollens* and *affirmation of the consequent*.

That the stable models semantics yields the same models as the supported models semantics does not come as a surprise, since by Fages' theorem [3] both semantics are identical for programs that are *tight*, that is, do not involve "positive loops". The experiments discussed in [19] all lead to tight programs. It is not difficult to transfer Fages' theorem to a three-valued setting, where for tight programs the models represented by the fixed-points of the Fitting operator are exactly the partial stable models. Accordingly, the model represented by the least fixed-point of the Fitting operator is the well-founded model.

It seems currently unclear whether there are interesting scenarios of human reasoning that would lead to logic programs which are not tight or to programs of richer classes, for example by permitting disjunctive rules, where the completion based semantics might differ from those based on stable models.

4 Three-Valued Logic Programming Semantics for Modeling Human Reasoning

In human reasoning positive and negative knowledge is apparently not handled symmetrically. The Fitting operator [4] represents a particular asymmetric way of inferring positive information (heads of rules whose body is verified) and negative information (negated heads in case the bodies of all rules with the atom as head are falsified). As shown in [19], this matches the empirical results from [1] about the suppression task when open-world predicates are properly considered.

The Fitting operator semantics and the partial stable models semantics can be expressed with second-order operators in two-valued classical logic [20], where the representation of the partial stable models semantics is based on a translation into the two-valued stable models semantics [9]. The following formulas show how the Fitting operator semantics is rendered in our framework for the example program (iii), with t considered as open-world:

$$\begin{aligned}
&\mathsf{circ}_{\mathsf{INFMIN}}(\mathsf{fitting}_{\{t^0,t^1,t^2\}}((l^0 \leftarrow e^2 \wedge \neg ab_1^1) \wedge (l^0 \leftarrow t^2 \wedge \neg ab_2^1))) && (1)\\
\equiv\ &\mathsf{circ}_{\mathsf{INFMIN}}(\mathsf{CONS} \wedge (l^0 \leftarrow e^0 \wedge \neg ab_1^1) \wedge (l^0 \leftarrow t^0 \wedge \neg ab_2^1) \wedge && (2)\\
&\qquad (l^1 \rightarrow (e^1 \wedge \neg ab_1^0) \vee (t^1 \wedge \neg ab_2^0)) \wedge\\
&\qquad (e^1 \rightarrow \mathsf{false}) \wedge (ab_1^1 \rightarrow \mathsf{false}) \wedge (ab_2^1 \rightarrow \mathsf{false}))\\
\equiv\ &\mathsf{circ}_{\mathsf{INFMIN}}(\neg ab_1^0 \wedge \neg ab_1^1 \wedge \neg ab_2^1 \wedge \neg ab_2^0 \wedge \neg e^0 \wedge \neg e^1 \wedge && (3)\\
&\qquad ((l^0 \wedge l^1 \wedge t^0 \wedge t^1) \vee && \text{(vi)}\\
&\qquad (l^0 \wedge l^1 \wedge \neg t^0 \wedge t^1) \vee\\
&\qquad (\neg l^0 \wedge l^1 \wedge \neg t^0 \wedge t^1) \vee\\
&\qquad (\neg l^0 \wedge \neg l^1 \wedge \neg t^0 \wedge t^1) \vee\\
&\qquad (\neg l^0 \wedge \neg l^1 \wedge \neg t^0 \wedge \neg t^1)))\\
\equiv\ &\neg ab_1^0 \wedge \neg ab_1^1 \wedge \neg ab_2^1 \wedge \neg ab_2^0 \wedge \neg e^0 \wedge \neg e^1 \wedge \neg l^0 \wedge l^1 \wedge \neg t^0 \wedge t^1. && (4)
\end{aligned}$$

In the classical representation of the logic program in formula (1), three roles of predicate occurrences are distinguished: In the head, subjected to negation as failure, and in the positive body, superscripted by 0, 1, 2, respectively. The operator $\mathsf{fitting}$ is a shorthand for a certain combination of second-order operators that renders the semantics of the Fitting operator. Its subscript argument in (1) specifies that t is to be considered open-world. (See [20] for precise definitions.)

Step (2) shows the formula (1) after eliminating fitting, which can be considered as the result of a program transformation: CONS, a shorthand for an axiom that excludes certain unwanted models as discussed below, a classical representation of the program with head and positive body superscripted with 0 and the negative body with 1, and the converses of the completion of the latter variant of the program with superscripts flipped.

The predicate superscripts now indicate the contribution to three-valued models: An interpretation assigns the "three-valued" truth value TRUE to an atom p if it is a model of $p^0 \wedge p^1$. It assigns FALSE to p if it is a model of $\neg p^0 \wedge \neg p^1$, and it assigns UNDEFINED to p if it a model of $\neg p^0 \wedge p^1$. The remaining possibility that it is a model $p^0 \wedge \neg p^1$ is excluded by the axiom CONS. In step (3) the argument formula of the circumscription is syntactically simplified, indicating its five three-valued models. Finally, in step (4) the circumscription operator is eliminated, rendering the selection of the *least* model of the Fitting operator. The symbol INFMIN is a shorthand for a parameter that specifies that predicates with superscript 0 are minimized while those with superscript 1 are maximized (our version of predicate circumscription allows maximization, dual to minimization [22]). A model of a formula circumscribed in this way is an *informationally minimal* model of the circumscribed formula, since viewed as three-valued, there is no other model of the formula whose assignments of atoms to FALSE and to TRUE (but not to UNDEFINED) are properly contained in those assignments of the first model.

With this approach, "positive" and "negative" aspects correspond to differently superscripted predicates. In combination they yield three-valued truth values. The same combinations are obtained for the partial stable models semantics based on the translation in [9], where it is suggested to consider the predicates superscripted with 1 as representing potential truth. It may be of interest to investigate whether there are correspondences to observed human reasoning at the level of this "lower layer" of components with different status, representing "true" and "potentially true" knowledge.

5 Three-Valued Logics for Modeling Human Reasoning

The least model of the Fitting operator applied to a normal logic program is the unique informationally minimal model of the program's completion under a certain three-valued logic [4]. This also applies to the considered generalizations in which open-world predicates are taken into account. In the corresponding three-valued logic the semantics of the biconditional must yield TRUE if and only if both argument formulas have the same truth value. This is the case for the semantics of the biconditional considered in [4], where FALSE is the value in all other cases, and also for the biconditional derived from Łukasiewicz's implication, considered in [6], where UNDEFINED is the value if the truth value of exactly one argument is UNDEFINED, and FALSE is the value in the remaining cases. With both variants of the three-valued biconditional, formulas with the syntactic form of a completed normal logic program have exactly the same three-valued models. Similarly, if normal logic programs themselves are considered as three-valued formulas, the models with respect of the implication seq_3 [5], the analogue to the mentioned biconditional considered in [4], used in [15] for logic programs, are exactly the models obtained with Łukasiewicz's implication.

In Section 4 we have seen an assignment of three-valued truth values to *atomic* formulas, by two-valued interpretations over atoms decorated with superscripts 0 and 1. This principle can be extended to *complex* formulas of certain three-valued logics. In [20] such an extension is given for the three-valued logic S_3 [17], the logic applied in [4] to render the semantics of the Fitting operator: A valuation function from S_3 formulas and two-valued interpretations over atoms with superscripts 0 and 1 onto three truth values is specified, where the values of atoms are assigned as described in Section 4 and the values of complex formulas according to the involved three valued connectives. The valuation function is complemented by a translation function that maps an S_3-formula to a classical propositional formula whose predicates are decorated with superscripts 0 and 1 in a compatible way. That is, an interpretation over the superscripted atoms is a model of the translated S_3 formula if and only if the valuation function applied to the formula and the interpretation yields TRUE. In the following example, the part of the argument formula of the circumscription that follows CONS is the value of this translation function, applied to the completion of the program (iii), extended by the rule $t \leftarrow t$ which expresses that t is open-world, viewed as an S_3-formula:

$$\begin{aligned}\mathsf{circ}_{\mathsf{INFMIN}}(&\mathsf{CONS} \wedge \\ &(l^0 \leftrightarrow (e^0 \wedge \neg ab_1^1) \vee (t^0 \wedge \neg ab_2^1)) \wedge (t^0 \leftrightarrow t^0) \wedge (e^0 \leftrightarrow \mathsf{false}) \wedge \\ &(ab_1^0 \leftrightarrow \mathsf{false}) \wedge (ab_2^0 \leftrightarrow \mathsf{false}) \wedge \\ &(l^1 \leftrightarrow (e^1 \wedge \neg ab_1^0) \vee (t^1 \wedge \neg ab_2^0)) \wedge (t^1 \leftrightarrow t^1) \wedge (e^1 \leftrightarrow \mathsf{false}) \wedge \\ &(ab_1^1 \leftrightarrow \mathsf{false}) \wedge (ab_2^1 \leftrightarrow \mathsf{false})).\end{aligned} \tag{vii}$$

The translation function yields two instances of classical representations of the completed program, with superscripts distinguishing predicate occurrences subjected to negation as failure, flipped in the two instances. It can be shown that the formula (vii) is equivalent to (vi) [20]. The translation extends the compositional view of three truth values in terms of two values and superscripted atoms outlined in Section 4 to complex logic formulas.

6 Conclusion

We have looked into the modeling of human reasoning tasks by logic programs proposed in [18, 19] and developed further in [6–8, 2] from the perspective of a framework where nonmonotonic semantics are represented in classical logic extended by some second-order operators. These classical formulas provide a view on logic programming and the modeled human reasoning that is not tied to a particular way of processing, that allows to represent the interplay of different nonmonotonic semantics, and that allows formalization and mechanization not only on the "object-level" by computing outcomes of human reasoning, but also on the "meta-level" of reasoning about features and principles of human reasoning.

The framework lets general features and requirements of the approach to model human reasoning by logic programming become apparent, such as the interplay of open- and closed-world assumption, which can be straightforwardly expressed by parameterizing circumscription appropriately, from where it transfers to several other logic programming semantics by expressing them in terms of circumscription. The inspection of three-valued logic programming semantics and three-valued logics applied to model human reasoning leads to "lower

level" representations, where combinable pieces of knowledge (atomic formulas) are associated with different aspects, like being *true* versus being *potentially true*. This suggests future investigations in the area of human reasoning to see whether some of its features can be analogously explained by combination of related primitives.

References

1. R. M. J. Byrne. Suppressing valid inferences with conditionals. *Cognition*, 31:61–83, 1989.
2. E.-A. Dietz, S. Hölldobler, and M. Ragni. A computational logic approach to the suppression task. In *CogSci 2012*, 2012. To appear.
3. F. Fages. Consistency of Clark's completion and existence of stable models. *Journal of Methods of Logic in Computer Science*, (1):51–60, 1994.
4. M. Fitting. A Kripke-Kleene semantics for logic programs. *Journal of Logic Programming*, 2(4):295–312, 1985.
5. S. Gottwald. *A Treatise on Many-Valued Logics*, volume 9 of *Studies in Logic and Computation*. Research Studies Press, Baldock, UK, 2001.
6. S. Hölldobler and C. D. P. Kencana Ramli. Logic programs under three-valued Łukasiewicz semantics. In *ICLP 2009*, volume 5649 of *LNCS*, pages 464–478. Springer, 2009.
7. S. Hölldobler and C. D. P. Kencana Ramli. Logics and networks for human reasoning. In *ICANN'09*, pages 85–94, 2009.
8. S. Hölldobler, T. Philipp, and C. Wernhard. An abductive model for human reasoning (poster paper). In *Logical Formalizations of Commonsense Reasoning*, AAAI Spring Symposium Series, pages 135–138. AAAI Press, 2011.
9. T. Janhunen, I. Niemelä, D. Seipel, P. Simons, and J.-H. You. Unfolding partiality and disjunctions in stable model semantics. *ACM Transactions on Computational Logic*, 7(1):1–37, 2006.
10. J. Lee and F. Lin. Loop formulas for circumscription. *Artificial Intelligence*, 170:160–185, 2006.
11. V. Lifschitz. Circumscription. In *Handbook of Logic in AI and Logic Programming*, volume 3, pages 298–352. Oxford University Press, Oxford, 1994.
12. V. Lifschitz. Twelve definitions of a stable model. In *ICLP 2008*, volume 5366 of *LNCS*, pages 37–51. Springer, 2008.
13. F. Lin. *A Study of Nonmonotonic Reasoning*. PhD thesis, Stanford Univ., 1991.
14. J. McCarthy. Circumscription – a form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.
15. T. Przymusinski. Every logic program has a natural stratification and an iterated fixed point model. In *PODS 89*, pages 11–21. ACM SIGACT-SIGMOD, 1989.
16. T. Przymusinski. Well-founded semantics coincides with three-valued stable semantics. *Fundamenta Informaticae*, 13(4):445–464, 1990.
17. N. Rescher. *Many-Valued Logic*. McGraw-Hill, New York, 1969.
18. K. Stenning and M. van Lambalgen. Semantic interpretation as computation in nonmonotonic logic: The real meaning of the suppression task. *Cognitive Science*, (29):916–960, 2005.
19. K. Stenning and M. van Lambalgen. *Human Reasoning and Cognitive Science*. MIT Press, Cambridge, MA, 2008.
20. C. Wernhard. Forward human reasoning modeled by logic programming modeled by classical logic with circumscription and projection. Technical Report Knowledge Representation and Reasoning 11-07, Technische Universität Dresden, 2011.
21. C. Wernhard. The globally weakest sufficient condition as basis for abduction in logic programming. Unpublished manuscript, 2012.
22. C. Wernhard. Projection and scope-determined circumscription. *Journal of Symbolic Computation*, 47:1089–1108, 2012.