The Role of Preferences in Logic Programming: Nonmonotonic Reasoning, User Preferences, Decision under Uncertainty

Dr. Roberto Confalonieri

Supervisor(s): Dr. Javier Vázquez-Salceda, Dr. Juan Carlos Nieves

Abstract— **Preferences have a multi-faceted relationship with several knowledge representation domains and they have been understood in different ways in the Artificial Intelligence literature. In this thesis, we study and clarify the role that preferences can play in the setting of Answer Set Programming.**

Keywords— **preferences, nonmonotonic reasoning, uncertain nonmonotonic reasoning, decision under uncertainty, answer set programming**

I. Introduction

PREFERENCES are a multi-disciplinary topic and they have been extensively been studied in economics, psychology, philosophy, logics and other human-centered disciplines. Nevertheless, they are a relative new topic in Artificial Intelligence (AI) [31], [24] and they have become of great interest for the development of reasoning mechanisms in intelligent systems [9].

Intelligent systems are usually studied and developed to assist users in fulfilling complex tasks. To act autonomously and to intelligently support users, such systems require a concise and processable representation of *incomplete* and *uncertain* information. To understand user choices and to choose among different options in a desirable way, such system also need a concise and processable representation of *preferences*.

In the development of reasoning models, preferences are intrinsically related to several domains such as *reasoning under incomplete and uncertain information*, *user preference modeling*, and *qualitative decision under uncertainty*. Indeed, preferences can provide an effective way to choose among outcomes when these stand for:

- the most plausible states of the world according to pieces of default knowledge such as in nonmonotonic reasoning [38], [22], [14] and in uncertain nonmonotonic reasoning [34], [39],
- the most satisfactory states of the world when expressing user preferences in a given context [11], [8], [6], or
- possible decisions when a decision is a matter of preference and uncertainty such as in qualitative decision making [7], [40], [27], [26].

For each of the above perspectives, several non-classical logics have been proposed. These formalisms usually focus on rich representation languages, axiomatizations, and sound and complete semantics. Most of the time, the computational aspects (and implementations) are left in the background. On the other hand, when one is interested in developing practical reasoning tools, a setting able to represent and to reason under incomplete and uncertain knowledge, to model preferences and qualitative decision making, and that can provide efficient computation must be taken into account.

In this respect, *Answer Set Programming* (ASP) [3] has shown to offer a good compromise between symbolic representation and computation of knowledge. ASP has been regarded to be expressive enough to address many knowledge representation problem in AI and to offer several efficient answer set solvers [36], [32], [29].

However, in the vast ASP literature available, the role played by preferences is not very clear. Although several ASP extensions have been proposed to deal with preferences [22], with uncertainty [35], [37], [4], [5] and with decision making [10], [30], it seems that not all the perspectives mentioned earlier can be cast in ASP, at least, in a direct way.

The main objective of this thesis is to tackle to role played by preferences in the ASP setting from the different perspectives of *nonmonotonic reasoning and uncertainty*, *user preferences*, and *qualitative decision making under uncertainty*. For each perspective, we analyse the role of preferences in the AI and ASP literature, and we propose (when needed) ASP frameworks and methods that allow to bridge the gap between the AI and ASP preference points of view.

II. Nonmonotonic Reasoning and Uncertainty

In nonmonotonic reasoning, the notion of preference is closely tied with the selection of default rules. Default rules, *i.e., normally, p's are q's*, are abstract representations of general properties which can admit exceptions. Given a set of defaults, one is interested in selecting the *preferred models* (in a logical sense) which are in agreement with those defaults [38], [22]. The selection is usually achieved using two types of preferences: *implicit* and *explicit* [14].

Explicit preferences among default rules have been proposed in extended versions of nonmonotonic logics. In these approaches, a partial order is explicitly assigned to the defaults. Instead, implicit preferences can be obtained from the information described by the rules themselves such as in System Z [38]. System Z lies on a *specificity principle* according to which the logical models which are supported by more specific rules are always preferred to those supported by less specific rules.

Concerning logic programming, ASP models default rules by making exceptions explicit and by capturing them using *nega-*

Current Affiliation: Institut de Recherche en Informatique Toulouse (IRIT), Universitè Paul Sabatier, 118 Route de Narbonne, 31062 Toulouse Cedex 9, France. E-mail: confalonieri@lsi.upc.edu.

Universitat Politècnica de Catalunya, Dept. Llenguatges i Sistemes Informàtics, C/ Jordi Girona Salgado 1-3, E - 08034 Barcelona. E-mail: {jvazquez,jcnieves}@lsi.upc.edu.

tion as failure not. A default rule is represented by a normal rule $q \leftarrow p, not\ ab$, in which the piece of knowledge $not\ ab$ captures an exception ab *w.r.t.* the default rule. In ASP, the concept of preference is already present at semantics level, since the concept of preferred models is intrinsically defined in the *answer set semantics* definition of an answer set of a logic program. In fact, answer sets of a logic program correspond to the preferred models and tend to identify those models in which everything is *as normal as possible* [3]. However, this basic approach of assuming normality still faces a difficulty [14]. When rules are in conflict, a criterion for the selection of rules is needed. To this end, several ASP extensions for handling preferences have been developed [22]. In most ASP approaches, preferences are used to select rules in an explicit way by assigning a preference order to the defaults.

Selecting default rules explicitly is not always possible. For instance, when different pieces of knowledge encoded by default rules are integrated, possible exceptions must be made explicit before to be able to select default rules. When such rules contain different levels of specificity, an implicit preference order among the rules can be drawn in order to identify the exceptions to default rules and to deal with incomplete information. To this end, our first contribution has been:

C1: An approach to nonmonotonic reasoning using implicit preferences

The proposed logic programming procedure [21] is able to consider implicit preferences based on the specificity of the defaults and to handle incomplete information in extended definite programs (*i.e.,* without not). Rules' specificity is obtained by means of the Z-ordering of System Z [38] (adapted to logic program). Based on the partitions induced by the Z-ordering, we have designed a rewriting procedure which rewrites default rules into rules in which exceptions are made explicit. In particular, we have been able to show that exceptions to logic program rules can be caught in terms of strong negation ($\neg$) and a set of completion rules aiming at completing the incomplete information in the program. This result strengthens an earlier result by Dimipoulos and Kakas [23], showing how, in logic programming, nonmonotonic reasoning can be handled with preferences rather than negation as failure.

The representation of simple default statements encoded as extended definite rules is of quite limited applicability for representing user knowledge in an intelligent system. In particular, having only one option in each context does not give so much flexibility in modeling different choices. The Logic Programs with Ordered Disjunction (LPODs) framework [12] allows to specify more than one option in a given context by means of ordered disjunction rules. Ordered disjunction rules are rules which are augmented with an ordered disjunction connective $\times$. The $\times$ allows to express a preference order among atoms such as: *normally, in context c, p is preferred to q* (encoded as $p \times q \leftarrow c, not\ ab$). The answer sets of an LPOD can be selected by specifying a comparison criterion which takes the preference order induced by $\times$ into account. LPODs is designed to specify *context-dependent preferences* which can be used to do two things [14]: to specify a preference order among exceptions for the selection of default rules and to specify a preference order among several alternatives for encoding *user preferences* (as we will discuss in Section III).

In the former view, an ordered disjunction rule can model a preference order between the exceptions related to other defaults. For instance, given the classical nonmonotonic reasoning example of *normally, birds can fly* (encoded as $f \leftarrow b$), *normally, penguins cannot fly* (encoded as $\neg f \leftarrow p$) and *penguins are birds* ($b \leftarrow p$), an ordered disjunction rule such as $\neg f \times f \leftarrow p$ can state that *in case of penguins it is preferred to assume that penguins cannot fly rather than they can* (thus, avoiding conflicting conclusions). This point of view can be extended to the case in which default rules are not totally certain.

Uncertain nonmonotonic reasoning amounts to handle uncertain default rules. Handling uncertain default rules basically involves two tasks, that are, the selection of default rules and the management of the uncertainty. This can be done in different ways which depend on the setting considered. For instance, in possibilistic logic [39], the selection of defaults is independent from the uncertainty management. This is due to the fact that, in possibilistic logic, incomplete information cannot be represented in an explicit way. As such, default rules need to be first transformed into rules in which exceptions are made explicit, before to be able to process the uncertainty associated with them. Instead, in ASP, such transformation is not required and the selection of uncertain default rules can be a matter of incomplete and uncertain information at the same time. Indeed, one can be interested in selecting default rules based on the most certain rules in a logic program. Most of the approaches in logic programming able to deal with uncertainty do not consider nonmonotonic reasoning or, if they do [35], [37], [4], [5], they do not contemplate the selection of uncertain default rules in terms of explicit preference. Therefore, our second contribution has been:

C2: A framework for dealing with explicit preferences and uncertainty

The proposed possibilistic logic programming framework, called Logic Programs with Possibilistic Ordered Disjunction (LPPODs) [18], supports the selection of uncertain default rules. The framework is able to model explicit preferences about rules having exceptions and certainty degrees in terms of necessity values according to possibilistic logic [25]. An LPPOD consist of a set of possibilistic ordered disjunction rules such as *normally, it is α-certain that in context c, p is preferred to q* (encoded as $\alpha : p \times q \leftarrow c, not\ ab$). LPPODs is the first logic programming specification able to consider explicit preferences and uncertainty together based on possibilistic logic and answer set semantics. LPPODs extends two existing logic programming frameworks: LPODs and possibilistic normal programs [35]. On the one hand, from LPODs, the framework inherits the distinctive feature of expressing explicit context-dependent preferences among different exceptions to default rules (modeled as atoms of a logic program). On the other hand, possibilistic normal programs allow to capture certainty degrees to express to what extent a rule is certain (modeled as necessity values according to possibilistic logic) and to associate certainty degrees with the atoms in the possibilistic answer sets (inferred in terms of possibilistic answer set semantics). As in the case of LPODs, the $\times$ can induce an order among the possibilistic answer sets of an LPPOD. In order to define such ordering, we have defined a possibilistic comparison criterion which considers both preference and certainty degrees. It is worthy to point out that, although LPPODs can deal with preference and uncertainty, they cannot be used for dealing with qualitative decision making in a direct way. This is given to the fact that qualitative decision making requires a setting in which preferences and uncertainty are kept separate (as we will discuss in Section IV). Finally, one important aspect of the LPPODs framework is that LPPODs semantics is computable and that its complexity belongs to the same complexity class of its classical part, that is, LPODs. This represent an important result, since it shows that LPPODs can yield a more expressive framework without increasing the complexity of the underpinning semantics. The LPPODs semantics has been implemented in an ASP-based solver called *posPsmodels*.[1]

III. User Preferences

Preferences are not only a matter of (uncertain) default rule selection. Preferences are mainly understood as a means for representing user choices. As such, they are an essential aspect in any intelligent system that offers personalised content to users. In the modeling of user preferences, preferences have to be compactly represented for encoding different options which can depend on contextual information. This concern has fostered the study of *compact preference representation languages* such as [11], [8], [6], just to cite few of them. Generally speaking, these specifications define a preference model in a compact way and a preference relation to rank the outcomes of the model. The way in which preferences are encoded and the ranking is obtained mainly depends on the specification used.

ASP is also a suitable setting for modeling user preferences. Several ASP proposals not only can encode user preferences in a compact way, but they can also draw an order between the outcomes of the preference model [13], [2], [12]. Among them, LPODs is a suitable specification for capturing context-dependent preferences (*e.g., at lunch, Roberto prefers to eat meat over fish unless he is vegetarian,* encoded as $meat \times fish \leftarrow lunch, not\ vegetarian$) and for specifying an order among several preferences of the users (modeled in terms of the atoms in the answer sets of an LPOD). Capturing context-dependent preferences and drawing an order among the outcomes of a preference model are important features when building a preference handling method in context-aware (information) systems [1]. Indeed, these systems need to model users in order to be able to filter big amount of information in a personalised way. Users' models can be built by modeling user preferences in terms of *user profiles*. The modeling of user profiles bring several issues related to knowledge representation. User profiles should be rich enough to represent preferences that can depend both on contextual and incomplete information, but sufficiently compact to be processed in a fast way. Moreover, user preferences can change over the time. This means that preferences in one context can be more important than others or that some preferences can become obsolete. These concerns suggest that the enhancement of the personalisation capabilities of a context-aware system by means of a logic programming-based preference model can be valuable. Then, our third contribution has been:

C3: A logic programming-based preference model for handling user preferences in a context-aware system

In [16], we have proposed to model user preferences in a specific category of context-aware systems, Interactive Community Displays (ICDs) [15], by means of a preference handling method based on the LPPODs framework (Section II). Although LPPODs was introduced in the context of uncertain nonmonotonic reasoning, its preference representation capabilities have fit particularly well for representing user profiles in an ICD. Possibilistic ordered disjunction rules such as *we are fully certain that Roberto prefers to eat in an Italian restaurant over a Mexican restaurant if time is not between 15 and 17* (encoded as $fully - certain : Italian \times Mexican \leftarrow restaurant, not\ 15 - 17h$), or *we are almost certain that he prefers to pay by Master Card over using his Visa* (encoded as $almost - certain : MasterCard \times Visa$) can meet different requirements imposed by the representation of user profiles: (i) they capture contextual information such as *date* and *time, user location, weather condition*, etc. (according to the first rule above, the restaurants suggested to Roberto will be different depending on the time of his petition); (ii) they represent incomplete information by means of negation as failure (Roberto prefers to go to an Italian restaurant unless time is between 15 and 17); (iii) they measure to what extent preferences in a given context are more important than others by means of qualitative (uncertainty) labels associated with preference rules (in the above example, preferences about restaurants can be considered more important than the ones about the paying method); (iv) they model user preferences in a compact way and they can be used in practical applications, since they can be processed by the *posPsmodels* solver. A demonstration of how LPPODs has been used in the ICD system is available at `http://research.tmtfactory.com/index.php/tmtresearch/projects/49/`. Moreover, in [20], we have shown how LPPODs can also accommodate different kinds of preference queries, originally proposed in the possibilistic logic setting [**?**].

Nevertheless, preference expressions supported by LPPODs are rather simple. Since LPPODs generalizes LPODs, LPPODs inherits the expressivity of LPODs. This basically means that the LPPODs syntax only allows to express alternatives between single preferences. In reality, user preferences are usually more complex and they demand a richer syntax in order to express equalities and/or combinations between several options. Indeed, a user can prefer to have $Italian$ over $Mexican$ food over not having any of them, *i.e.,* expressions such as $Mediterranean \times Mexican \times (not\ \ Mediterranean \wedge not\ \ Mexican) \leftarrow restaurant, 13 - 15h$; or (s)he may prefer to pay either by Master Card or by Visa over paying cash, *i.e.,* expressions such as $(MasterCard \vee Visa) \times cash$, or even more complex expressions such as $(pub \vee bar) \times (cinema \wedge \neg tv) \leftarrow night$ or $(pub \wedge (expensive \times cheap)) \times bar \times (not\ pub \wedge not\ bar) \leftarrow night \vee (not\ \ busy \wedge afternoon)$. These examples suggest that studying a less restricted syntax could be valuable. To model such complex context-dependent preference expressions, we have proposed:

[1] `https://github.com/rconfalonieri/posPsmodels/wiki`

C4: A framework for handling nested preferences

The proposed framework, called Nested Logic Programs with Ordered Disjunction (LPODs$^+$) [17], is an extension of LPODs and it allows to capture nested preference expressions built by means of connectives $\{\vee, \wedge, \neg, not, \times\}$. We have shown how LPODs$^+$ can be defined in an easy way by reusing and extending some definitions related to the syntax and semantics of nested logic programs [33] (indeed LPODs$^+$ properly generalizes nested logic programs). Since the syntax of LPOD$^+$ is too complex to be handled by existent answer set solvers, we have reduced the problem of inferring the answer sets of a nested logic program with ordered disjunction (OD$^+$-program) to the problem of inferring the answer sets of an equivalent disjunctive logic program. To this end, we have specified a set of transformation rules which can transform any OD$^+$-program into a disjunctive logic program by preserving the LPODs$^+$ semantics. This is an important result. In fact, despite the flexibility added at the syntactic level, the complexity of deciding whether an OD$^+$-program has at least one answer set corresponds to the complexity class of disjunctive logic programs, that is, Σ_2^P [28]. In this way, complexity results of LPODs$^+$ can be lifted by the Σ_2^P-completeness of disjunctive logic programs. The LPODs$^+$ framework represents a first step towards an extension of the LPPODs framework.

IV. Decision Making under Uncertainty

Preferences are also important for selecting an optimal decision in qualitative decision making under uncertainty (DMU). In qualitative DMU, preferences are used to mark the states of the world that a user or decision maker prefers to achieve when taking a decision. Then, making an optimal decision amounts to find a decision (if it exists) that satisfies as many preferences as possible taking into account that the knowledge about the world is not fully certain. Choosing an optimal decision is not an easy task. It requires a theory able to represent preferences and uncertainty and to rank decisions according to decision rules.

Several qualitative decision making frameworks have been proposed in the literature [7], [40], [27], [26]. Among them, possibility theory has shown to provide a convenient setting for modeling qualitative DMU [27], [26]. According to this approach, knowledge about the world and preferences are described by two different stratified propositional knowledge bases which capture the certainty and the priority of formulas respectively. Then, by assuming a commensurateness hypothesis between the level of certainty and the preference priority, pessimistic and optimistic criteria, justified on the basis of postulates, have been proposed for the selection of an optimal decision [27].

Despite the nice computational features offered by ASP, the use of a logic programming approach for modeling qualitative DMU and for computing an optimal decision has remained almost unexplored. Existing proposals [10], [30] suggest a general methodology for modeling qualitative decision problems by means of LPODs. The common idea is to encode a decision problem by means of an LPOD and to select an optimal decision by means of comparison criteria based on the LPODs semantics. However, these approaches are not satisfactory *w.r.t.* the handling of preferences and uncertainty for several reasons. First, preferences are always considered to be part of the world description, whereas, in qualitative DMU, preferences and knowledge about the world are modeled in two separate knowledge bases. Secondly, these approaches do not deal with uncertainty, or, if they do, they do not treat uncertain information in a formal way. Therefore, it is interesting to look for a methodology able to capture qualitative DMU in logic programming (in a way closer to its possibilistic logic formulation) and to compute the pessimistic and optimistic criteria. To this end, our fifth contribution has been:

C5: An ASP-based methodology for computing decision under uncertainty

Contrary to what existing logic programming-based methodologies for handling qualitative decision proposed, in our approach [19], we associate preferences and knowledge about the world with priority and certainty degrees modeled in terms of necessity values (according to possibilistic logic). Then, the solutions to a given decision problem are computed on the basis of a procedure which finds an optimal decision (if it exists) according to the pessimistic and optimistic criteria proposed in the possibilistic logic setting. Briefly, the procedure works as follows. First, we have encoded a decision problem in terms abduction and LPODs in order to deal with the case of fully certain knowledge and all-or-nothing preferences. Then, we have generalized this result to the case in which knowledge and preferences are matters of degrees using LPPODs (which offers a qualitative framework for handling uncertain information). Although we have not addressed any implementation details in the thesis, the described approach can be implemented on top of the *posPsmodels* solver. The proposed methodology attempt to bridge the gap between logic programming and qualitative DMU since, to the best of our knowledge, any proposal with a satisfactory handling of preferences under uncertainty was made in this respect.

V. Concluding Remarks

Throughout the research presented in this thesis, we have aimed at understanding the role played by preferences in ASP from different perspectives. Indeed, logic programming, by supporting symbolic representation and computation of knowledge, can provide an effective tool to build the reasoning mechanisms needed by intelligent systems which assist users in the fulfillment of complex tasks.

The overall objective of this thesis has turned to be an attempt to contribute on these active research trends in the field of logic programming. Therefore, our research has consisted of a collection of results on the study of the *representation of preferences* focusing on different domains such as *reasoning under incomplete and uncertain information, user preference modeling*, and *qualitative decision making under uncertainty*. From the research performed, we can conclude that preferences have a multi-faceted relationship with several knowledge representation domains and they have been understood in different ways in the literature. We believe that our proposed research analysis, frameworks, and methods might prove to be useful for the logic programming research towards the design of automated reasoning mechanisms for handling incomplete and uncertain knowledge, user preferences, and decision under uncertainty.

REFERENCES

[1] Baldauf, M., Dustdar, S., Rosenberg, F.: A survey on context-aware systems, *International Journal of Ad Hoc and Ubiquitous Computing*, **2**(4), 2007, 263–277.

[2] Balduccini, M., Mellarkod, V. S.: CR-Prolog with Ordered Disjunction, in: *Proceedings of the 2nd International Workshop on Answer Set Programming, Advances in Theory and Implementation* (M. D. Vos, A. Provetti, Eds.), vol. 78 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2003, 98–112.

[3] Baral, C.: *Knowledge Representation, Reasoning and Declarative Problem Solving*, Cambridge University Press, 2003.

[4] Baral, C., Gelfond, M., Rushton, N.: Probabilistic Reasoning with Answer Sets, *Theory and Practice of Logic Programming*, **9**(1), 2009, 57–144.

[5] Bauters, K., Schockaert, S., Cock, M. D., Vermeir, D.: Possibilistic Answer Set Programming Revisited, in: *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence, (UAI'10)*, AUAI Press, 2010.

[6] Benferhat, S., Dubois, D., Prade, H.: Towards a Possibilistic Logic Handling of Preferences, *Applied Intelligence*, **14**(3), 2001, 303–317.

[7] Boutilier, C.: Toward a Logic for Qualitative Decision Theory, in: *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning, (KR'94)* (J. Doyle, E. Sandewall, Eds.), Morgan Kaufmann, 1994, 75–86.

[8] Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H., Poole, D.: CP-nets: a tool for representing and reasoning with conditional ceteris paribus preference statements, *Journal of Artificial Intelligence Research*, **21**(1), 2004, 135–191.

[9] Brafman, R. I., Domshlak, C.: Preference Handling - An Introductory Tutorial, *AI Magazine*, **30**(1), 2009, 58–86.

[10] Brewka, G.: Answer Sets and Qualitative Decision Making, *Synthese*, **146**(1-2), 2005, 171–187.

[11] Brewka, G., Benferhat, S., Le Berre, D.: Qualitative Choice Logic, *Artificial Intelligence*, **157**(1-2), 2004, 203–237.

[12] Brewka, G., Niemelä, I., Syrjänen, T.: Logic Programs with Ordered Disjunction, *Computational Intelligence*, **20**(2), 2004, 333–357.

[13] Brewka, G., Niemelä, I., Truszczyński, M.: Answer Set Optimization, in: *Proceedings of 18th International Joint Conference on Artificial Intelligence, (IJCAI'03)* (G. Gottlob, T. Walsh, Eds.), Morgan Kaufmann, 2003, 867–872.

[14] Brewka, G., Niemelä, I., Truszczyński, M.: Preferences and Nonmonotonic Reasoning, *AI Magazine*, **29**(4), 2008, 69–78.

[15] Ceccaroni, L., Codina, V., Palau, M., Pous, M.: PaTac: Urban, Ubiquitous, Personalized Services for Citizens and Tourists, in: *Proceedings of the Third International Conference on Digital Society*, IEEE Computer Society, Washington, DC, USA, 2009, 7–12.

[16] Confalonieri, R., Iñan, H., Palau, M.: Handling Uncertain User Preferences in a Context-aware System, in: *Proceedings of 14th International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems, (IPMU 2012), to appear*, 2012.

[17] Confalonieri, R., Nieves, J. C.: Nested Preferences in Answer Set Programming, *Fundamenta Informaticae*, **113**(1), 2011, 19–39, ISSN 0169-2968.

[18] Confalonieri, R., Nieves, J. C., Osorio, M., Vázquez-Salceda, J.: Possibilistic Semantics for Logic Programs with Ordered Disjunction, in: *Proceedings of 6th International Symposium on Foundations of Information and Knowledge Systems, (FoIKS 2010)* (S. Link, H. Prade, Eds.), vol. 5956 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, Heidelberg, 2010, 133–152.

[19] Confalonieri, R., Prade, H.: Answer Set Programming for Computing Decisions Under Uncertainty, in: *Proceedings of the 11th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2011)* (W. Liu, Ed.), vol. 6717 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, Berlin, Heidelberg, 2011, 485–496.

[20] Confalonieri, R., Prade, H.: Encoding Preference Queries to an Uncertain Database in Possibilistic Answer Set Programming, in: *Proceedings of 14th International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems, (IPMU 2012), to appear*, 2012.

[21] Confalonieri, R., Prade, H., Nieves, J. C.: Handling Exceptions in Logic Programming without Negation as Failure, in: *Proceedings of the 11th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2011)* (W. Liu, Ed.), vol. 6717 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, Berlin, Heidelberg, 2011, 509–520.

[22] Delgrande, J., Schaub, T., Tompits, H., Wang, K.: A classification and Survey of Preference Handling Approaches in Nonmonotonic Reasoning, *Computational Intelligence*, **20**(2), 2004, 308–334.

[23] Dimopoulos, Y., Kakas, A. C.: Logic Programming without Negation as Failure, in: *Proceedings of the 1995 International Symposium on Logic Programming* (J. W. Lloyd, Ed.), MIT Press, 1995, ISBN 0-262-62099-5, 369–383.

[24] Domshlak, C., Hüllermeier, E., Kaci, S., Prade, H.: Preferences in AI: An overview, *Artifical Intelligence*, **175**(7-8), May 2011, 1037–1052.

[25] Dubois, D., Lang, J., Prade, H.: Possibilistic logic, in: *Handbook of Logic in Artificial Intelligence and Logic Programming - Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning* (D. M. Gabbay, C. J. Hogger, J. A. Robinson, J. H. Siekmann, Eds.), Oxford University Press, Inc., New York, NY, USA, 1994, 439–513.

[26] Dubois, D., Prade, H.: Possibility theory as a basis for qualitative decision theory, in: *Proceedings of the 14th International Joint Conference on Artificial Intelligence, (IJCAI'95)*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995, 1924–1930.

[27] Dubois, D., Prade, H., Sabbadin, R.: Decision-theoretic foundations of qualitative possibility theory, *European Journal of Operational Research*, **128**(3), 2001, 459–478.

[28] Eiter, T., Gottlob, G.: On the computational cost of disjunctive logic programming: Propositional case, *Annals of Mathematics and Artificial Intelligence*, **15**(3), 1995, 289–323.

[29] Gebser, M., Kaminski, R., König, A., Schaub, T.: Advances in *gringo* series 3, in: *Proceedings of the 11th International Conference on Logic Programming and Nonmonotonic Reasoning, (LPNMR'11)* (J. P. Delgrande, W. Faber, Eds.), vol. 6645 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, Heidelberg, 2011, ISBN 978-3-642-20894-2, 345–351.

[30] Grabos, R.: Qualitative Model of Decision Making, in: *Artificial Intelligence: Methodology, Systems, and Applications* (C. Bussler, D. Fensel, Eds.), vol. 3192 of *Lecture Notes in Artificial Intelligence*, Springer Berlin / Heidelberg, 2004, 480–489.

[31] Kaci, S.: *Working with Preferences: Less Is More*, Springer-Verlag, 2011.

[32] Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., Scarcello, F.: The DLV system for knowledge representation and reasoning, *ACM Transactions on Computational Logic*, **7**(3), 2006, 499–562.

[33] Lifschitz, V., Tang, L. R., Turner, H.: Nested expressions in logic programs, *Annals of Mathematics and Artificial Intelligence*, **25**(3-4), 1999, 369–389.

[34] Lukasiewicz, T.: Weak nonmonotonic probabilistic logics, *Artificial Intelligence*, **168**(1-2), 2005, 119–161.

[35] Nicolas, P., Garcia, L., Stéphan, I., Lefèvre, C.: Possibilistic uncertainty handling for answer set programming, *Annals of Mathematics and Artificial Intelligence*, **47**(1-2), 2006, 139–181.

[36] Niemelä, I., Simons, P.: Smodels - An Implementation of the Stable Model and Well-Founded Semantics for Normal LP, in: *Proceeding of 4th International Conference on Logic Programming and Nonmonotonic Reasoning, (LPNMR'97)* (J. Dix, U. Furbach, A. Nerode, Eds.), vol. 1265 of *Lecture Notes in Computer Science*, Springer-Verlag, London, UK, 1997, 421–430.

[37] Nieves, J. C., Osorio, M., Cortés, U.: Semantics for Possibilistic Disjunctive Programs, *Theory and Practice of Logic Programming*, (doi: 10.1017/S1471068411000408), 2011.

[38] Pearl, J.: System Z: a natural ordering of defaults with tractable applications to nonmonotonic reasoning, in: *Proceedings of the 3rd conference on Theoretical Aspects of Reasoning about Knowledge, (TARK '90)*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990, 121–135.

[39] Dupin de Saint-Cyr, F., Prade, H.: Handling uncertainty and defeasibility in a possibilistic logic setting, *International Journal of Approximate Reasoning*, **49**(1), 2008, 67–82.

[40] Tan, S.-W., Pearl, J.: Qualitative decision theory, in: *Proceedings of the 12th National Conference on Artificial Intelligence (vol. 2), (AAAI'94)*, American Association for Artificial Intelligence, Menlo Park, CA, USA, 1994, 928–933.