Towards a Classification of Preference Handling Approaches in Nonmonotonic Reasoning

James P. Delgrande[1], **Torsten Schaub**[2*], **Hans Tompits**[3], **and Kewen Wang**[2]
[1] School of Computing Science, Simon Fraser University, Burnaby, B.C., Canada V5A 1S6
e-mail: jim@cs.sfu.ca
[2] Institut für Informatik, Universität Potsdam, Postfach 90 03 27, D–14439 Potsdam, Germany
e-mail: [torsten,kewen]@cs.uni-potsdam.de
[3] Institut für Informationssysteme 184/3, Technische Universität Wien,
Favoritenstraße 9–11, A–1040 Vienna, Austria, e-mail: tompits@kr.tuwien.ac.at

Abstract

In recent years there has been large amount of disparate work concerning the representation and reasoning with preferential information in approaches to nonmonotonic reasoning. Given the variety of underlying systems, assumptions, motivations, and intuitions, it is difficult to compare or relate one approach with another. Here we present an overview and classification for approaches to dealing with preference. A set of criteria for classifying approaches is given, followed by a set of desiderata that an approach might be expected to satisfy. A comprehensive set of approaches is subsequently given and classified with respect to these sets of underlying principles.

Introduction

The notion of *preference* is pervasive in commonsense reasoning, in part because preferences constitute a very natural and effective way of resolving indeterminate situations. In decision making, for example, one may have various desiderata, not all of which can be simultaneously satisfied; in such a situation, preferences among desiderata may allow one to come to an appropriate compromise solution. In legal reasoning, laws may conflict. Conflicts may be resolved by principles such as ruling that newer laws will have priority over less recent ones, and laws of a higher authority have priority over laws of a lower authority. For a conflict among these principles one may further decide that the "authority" preference takes priority over the "recency" preference.

Preference has a decidedly nonmonotonic flavour. Or, more accurately, it may be considered as having a *fundamental* nonmonotonic aspect. Roughly, given a preference ordering, however constituted, and some basic or case-specific information, Ψ, one may come up with a set of desired outcomes. However, a strict superset of this case-specific information, $\Psi \cup \Phi$, may lead to a different set of desired outcomes. For example, imagine feeding information into an automated financial advisor: that one is a relatively cautious investor, that one has a long-term horizon, etc. Given these preferences, a set of recommended mutual funds may be suggested by the automated advisor. If the user subsequently states that they also prefer that their funds invest in environmentally and socially responsible companies, then a different set of suggestions may well result.

In AI, a standard approach to handling preferences is to take an existing system and, in one fashion or another, equip it with preferences. For example (Brewka 1994; Delgrande & Schaub 2000a) add preferences to Default Logic while (McCarthy 1986; Lifschitz 1985a) and (Zhang & Foo 1997a; Brewka & Eiter 1999) do the same in circumscription and logic programming, respectively. However, although the notion of "preference" is intuitively straightforward, there is surprising variety in how this notion is realised in various approaches. Thus some approaches take a preference ordering as expressing a "desirability" that a property be adopted while in others the ordering expresses the order in which properties (or whatever) are to be considered. As we later describe, some approaches conflate the notion of *inheritance of properties* with the general notion of preference. The outcome of course is that, depending on how the notion of preference is interpreted, different conclusions may be forthcoming. At the same time, while logical preference handling already constitutes an indispensable means for legal reasoning systems (cf. (Gordon 1993; Prakken 1997b)), it is also being used in other application areas such as intelligent agents and e-commerce (Grosof 1999) and the resolution of grammatical ambiguities (Cui & Swift 2001).

In this paper we survey various approaches to handling preference information that have appeared in the literature. The intent is to consider ways (or *dimensions* or *axes*) in which the general notion of *preference* may be interpreted in a system, and to classify and evaluate approaches based on these axes. We begin, in the next section, by considering a number of ways, or dimensions, in which approaches may be classified. As well we discuss a number of desiderata that an approach or system may be expected to satisfy. In the following section we compare and contrast extant systems with respect to these criteria, concentrating on points of interest illustrated by a particular approach.

Comparing Approaches to Preference

In this section we consider a number of ways in which approaches to representing and reasoning with preferences can be compared. In the first subsection we consider ways to *classify* approaches to preference – that is, relatively neutral

---

[*] Affiliated with the School of Computing Science at Simon Fraser University, Burnaby, Canada.

criteria (or “axes” or “dimensions”) by which approaches may be distinguished or compared. In the second subsection we suggest possible desiderata for approaches, or properties that an approach ideally will satisfy. (Note however that the difference between a criterion and desideratum isn’t necessarily a clear-cut distinction).

Informally, a preference relation will be a binary relation $<$ between objects of a specific type (formulas, rules, sets of objects, etc.). Most often $<$ will be a partial order. The idea is that objects with higher precedence or preference are to be asserted (concluded, applied, …) over lower ranked objects. Thus for $\delta_2 < \delta_1$, if δ_1 and δ_2 are in conflict, one might expect, all other things being equal, that the higher-ranked object δ_1 will be asserted over the lower, δ_2.[1] Different approaches have further interpreted or constrained the relation $<$ in a multitude of ways; it is the purpose of this paper then to try to provide some framework, or perspective, to these various approaches.

There is one large and important subclass of preference-like relations that we will not discuss here, that associated with *inheritance of properties*. Essentially, in inheritance of properties, the preference ordering is determined by the *specificity* of antecedent information. As well, with inheritance, one only infers properties from the most specific applicable subclass. Consider rules concerning primary means of locomotion: “animals normally walk”, “birds normally fly”, “penguins normally swim”. If we learn that some thing is penguin (and so a bird and animal), then we would want to apply the highest-ranked default and, all other things being equal, conclude that it swims. However, if the penguin in question is hydrophobic, and so doesn’t swim, we wouldn’t want to inherit the next most specific conclusion, that it flys, and so in this case we would conclude nothing about locomotion. However, in a preference ordering one would try to apply the next default and so, again all other things being equal, conclude that the penguin flys. So inheritance of properties leads to different behaviour from preference orderings, as we interpret them here. See (Delgrande & Schaub 2000c) for a further discussion.

Classifying Approaches to Preference

We describe here a number of ways in which approaches to preference may be classified. For ease of exposition and concreteness, we will most often use Default Logic (Reiter 1980) to illustrate various concepts. Thus we may write

$$\frac{:Red}{Red} < \frac{:Blue}{Blue} < \frac{:Green}{Green} \tag{1}$$

to show a preference over colours, implemented as an ordering on default rules. However it should be emphasised that this is for illustration only; we have no particular preference for Default Logic;[2] some other system could be the “host” system; preferences need not be on rules and so on. Similarly a phrase such as “a higher-ranked rule is applied” is simply an abbreviation for the much more cumbersome “a higher-ranked object (be it a rule, term, formula, set, etc.) is applied (concluded, asserted, etc.).”

We have the following set of not-necessarily independent criteria for classifying approaches to preference:

Host system Previously (during the 1990’s) Default Logic (Reiter 1980) was by-and-large the host system of choice, in that the majority of approaches to adding preferences added them to Default Logic. More recently the emphasis has shifted to logic programs, and in particular extended logic programs. Likely this change reflects a general shift in focus in the research community, from Default Logic being the most popular nonmonotonic reasoning formalism, to the emergence of extended logic programs and answer set programming. The main thing that can be said about the “underlying system” is that it is easier to compare approaches that use the same base system. As well, a specific approach to preference may be “ported” from one underlying system to another, as for example is done in (Delgrande & Schaub 2000a; Delgrande, Schaub, & Tompits 2002) and (Brewka & Eiter 1999; Brewka & Eiter 2000).

Meta-level vs. object level preferences Most commonly, given some underlying host system, a preference ordering is imposed “externally” on rules of the system. For example, a default theory (D, W) may be extended to a *preferred default theory* $(D, W, <)$ where $< \subseteq D \times D$ gives a preference ordering on how rules may be applied. Alternatively, preferences may be imposed at the object level. For example in (Delgrande & Schaub 2000a), constants representing names are associated with the default rules. Instead of a relation $\delta_2 < \delta_1$ between default rules one can now assert $n_2 \prec n_1$ between the corresponding names, where $\prec$ is a (new) binary relation in the object language.

External, or meta-level preferences, have the advantage that they are (usually) easier to realise: the underlying inference relation is modified to take into account preferences. On the other hand, the object-level approach allows one to formalise preferences *within* a theory, instead of *about* a theory. As well, the object-level approach is potentially more flexible since one may cancel preferences or apply preferences in a context (e.g. $\alpha \supset \{\neg\}(n_2 \prec n_1)$), or have preferences apply by default (e.g. $\frac{\alpha : n_2 \prec n_1}{n_2 \prec n_1}$).

Static vs. dynamic preferences A closely-related distinction to the preceding concerns whether preferences are static, or fixed at the time the theory is specified, or dynamic, and so can be determined “on the fly”. An approach with external preferences will, of necessity, have static preferences. In the case of Default Logic, an approach with static, object-level preferences, would have preferences appearing only in the world knowledge W, and as ground atomic formulas; otherwise preferences would be (potentially) dynamic. In the case of extended logic programs, an approach with static, object-level preferences, would have preferences appearing only as ground facts (i.e. as rules of the form $(n_2 \prec n_1) \leftarrow$).

Properties of the preference ordering The majority of approaches assume that the relation $<$ is a (irreflexive) partial order, and this seems to be the minimal notion that would

---

[1] Note that some approaches use $<$ in the opposite sense to us.

[2] well, ok, actually we do.

justify the use of the term “preference”. However, one might go on and impose further conditions, such a connectivity or (in the case of infinite orderings) well-foundedness. As well, as we describe subsequently, in determining preferred outcomes, a given partial order may be extended to a total order.

What is the preference ordering an ordering on? A preference ordering $<$ is a binary relation on objects of some given type. This distinction then concerns the *things* that $<$ is a binary relation on. Although seemingly clear-cut, there are some subtleties here.

First, in Default Logic or extended logic programs, preferences would most naturally (in fact, seemingly unavoidably) be on the rules in a theory. However we have already noted one distinction: in an external preference relation, the preferences are indeed on the rules themselves. In an object-level preference relation, the preferences are expressed on constants naming the rules; it is then up to the implementer of such an approach to ensure that these constants do indeed denote the rules in question.

Second, there is a distinction between what a user would regard as a preference, and how the preference would be implemented. Thus, informally, it makes sense to think of preferences as being on formulas: for example, one might wish to express that green things are preferred to blue things, which are preferred to red. This could be expressed within a first-order language by predicates such as $\mathit{Pref}(\mathit{Green}(x), \mathit{Blue}(x))$ and $\mathit{Pref}(\mathit{Blue}(x), \mathit{Red}(x))$. Thus preferences would be expressed on (reified) formulas such as $\mathit{Green}(x)$. However, for implementation such a preference relation might be translated into a suitably-quantified version of something like (1). That is, the underlying reasoning machinery might make use of (here) Default Logic. Such a scheme has a number of advantages, including adherence to a knowledge engineering principle that says a user should only be given the power that they need for expressing a problem. As well, here the preference relation $\mathit{Pref}(\cdot,\cdot)$ would be translated into preferences on *normal* defaults which might then come with improved complexity characteristics over preferences on general rules. However, the specification of such a “knowledge engineered” language remains largely for future research.

At present, for Default Logic and extended logic programs, preference one way or another, is generally expressed on the rules. Exceptions to this include (Sakama & Inoue 2000), wherein preferences are given directly on *atoms* of the language, along with others such as (Pradhan & Minker 1996; Lifschitz 1985b). As well, we note that for a general approach, an account of preference on sets of objects will need to be given. For example in purchasing a car, one might wish to express that a car that is safe and economical is preferred to one that is just safe, which in turn is preferred to one that is safe and powerful. Thus perhaps:

$$\left\{\tfrac{:S}{S}, \tfrac{:P}{P}\right\} < \tfrac{:S}{S} < \left\{\tfrac{:S}{S}, \tfrac{:E}{S}\right\}.$$

Prescriptive vs. descriptive preferences The intuition behind a preference ordering is that higher-ranked rules are to be applied before lower-ranked ones. A major distinction as to how this can be done concerns whether $<$ specifies the order in which defaults are to be applied, or provides a notion of “desirability” that a rule be applied. In a *prescriptive* interpretation, the idea is that an order on defaults specifies the order in which the defaults are to be considered for application. Thus one applies (if possible) the most preferred default(s), the next most preferred, and so on. In a *descriptive* interpretation, the preference order represents a ranking on desired outcomes: the desirable (or: preferred) situation is one where the most preferred default(s) are applied.[3] The distinction between these interpretations is illustrated in the following example (Brewka & Eiter 2000):

$$\tfrac{:A}{A} < \tfrac{:\neg B}{\neg B} < \tfrac{A:B}{B}. \qquad (2)$$

Assume that there is no initial world knowledge. In a prescriptive interpretation, one would fail to apply the most preferred default (viz. $\frac{A:B}{B}$) since the antecedent isn’t provable. However, one might expect to apply the two lesser-preferred defaults, giving an extension containing $\{A, \neg B\}$.[4] In a descriptive interpretation one might observe that by applying the least-preferred default, the most preferred default can be applied; this yields an extension containing $\{A, B\}$.

A full discussion of this distinction is given in (Delgrande & Schaub 2000a). We briefly recapitulate two salient points here. First, a descriptive interpretation seems to rely on a meta-level specification of preference (more accurately: we are not aware of any object-level specifications, nor do we know how such might be carried out). In contrast, with a prescriptive object-level approach, we can potentially axiomatise within a theory how different preference orders interact.

Second, a prescriptive interpretation arguably comes with more representational force and allows a tighter characterisation of a domain. That is, a prescriptive interpretation forces a knowledge base designer to be explicit about what things should be applied in what order. A descriptive interpretation on the other hand gives a wish list of preferences which may or may not be meaningful. This is illustrated by the example (2), where the default $\frac{A:B}{B}$ has highest priority, but this default can only be applied if the prerequisite is proved; one way that this can come about is by applying the lower-ranked default $\frac{:A}{A}$. But this implies that $\frac{:A}{A}$ should be considered first and so have higher priority than $\frac{A:B}{B}$. As well, there is no situation in which $\frac{A:B}{B}$ can be applied and $\frac{:A}{A}$ cannot. Thus, the inference structure of default logic would seem to dictate that $\frac{:A}{A}$ not be ranked below $\frac{A:B}{B}$. Yet this is what the order $<$ in (2) stipulates.

Going from Preferences to Preferred Results Given a theory and a set of (object- or meta-level) preferences, the

---

[3] This isn’t necessarily a cut-and-dried distinction; for example, (Brewka & Eiter 2000) contains elements of both.

[4] This is for instance obtained in (Baader & Hollunder 1993a; Brewka 1994; Marek & Truszczyński 1993); the approach presented in (Delgrande & Schaub 2000a) yields no “preferred” extension.

standard computational problem is to generate a set of preferred outcomes. In Default Logic or extended logic programs, a preferreed set of outcomes would be part of an extension or answer set. The set of all extensions or answer sets would represent the possible sets of preferred outcomes when there is ambiguity in the underlying theory.

The preceding prescriptive/descriptive distinction represents a broad characterisation of computational strategies that may be employed. With respect to the preference ordering $<$, there are also two different specific computational strategies. In most of the existing approaches, the notion of a preferred extension is defined directly from the ordering $<$, unmediated by implied total orderings. There are also a few approaches, for example (Brewka & Eiter 1999), that are explicitly based on total ordering. That is, one has to first generate all possible total extensions of the given partial order $<$. Each total order then is used to generate a preferred extension. That is, we have two specific computational strategies with respect to the preference ordering $<$:

1. One might generate the set of all orderings from the partial order given by $<$ (e.g. (Brewka 1994)). Each total order then is used to generate a preferred extension.
2. One might generate preferred extensions directly from the ordering $<$, unmediated by implied total orderings.

   This second case has two realizations:

   (a) Generate all extensions of the underlying (preference-free) theory, and use $<$ to filter non-preferred extensions (e.g. (Sakama & Inoue 2000)).

   (b) Generate only preferred extensions directly from $<$ (e.g. (Delgrande & Schaub 2000b)).

Clearly the last possibility appears on the surface to be the most appealing, since it generates neither extraneous extensions nor specialisations of the preference ordering. On the other hand, there has been no work (that we are aware of) comparing the adequacy of these broad characterizations either from a formal or a pragmatic viewpoint.[5]

Evaluating Approaches to Preference

This subsection discusses a number of possible desiderata that an approach may be expected to satisfy. To begin with, (Brewka & Eiter 1999) propose two "principles" argued to constitute a minimal requirement for preference handling in a rule-based system. While the principles are formulated with respect to static preferences, the second need not be (Delgrande, Schaub, & Tompits 2002). The principles are expressed with respect to rule-based systems. Thus approaches such as Default Logic and logic programming are most naturally covered by these principles, although they are also applicable, for example, to a circumscriptive abnormality theory with preferences.

Principle I: *Let B_1 and B_2 be two extensions[6] of a prioritised theory $(T, <)$ generated by ground rules $R \cup \{d_1\}$ and $R \cup \{d_2\}$, where rules $d_1, d_2 \notin R$. If d_1 is preferred over d_2 then B_2 is not a preferred extension of T.*

The term "generated" is crucial in Principle I: For extension B a rule r is a generating rule just if its prerequisites are in B and it is not defeated by B.

Principle II: *Let B be a preferred extension of a prioritised theory $(T, <)$ and r a ground rule such that at least one prerequisite of r is not in B. Then B is a preferred extension of $(T \cup \{r\}, <')$ whenever $<'$ agrees with $<$ on priorities among rules in T.*

Thus adding a nonapplicable rule in a preferred extension does not make the extension nonpreferred, so long as prior preferences are not changed.

Complexity: For the major approaches to nonmonotonic reasoning, the complexity of general decision problems of interest (along with corresponding search problems) is known. Arguably, adding preferences to a given approach should not change the complexity of a given problem. Thus, consider a decision problem such as:

Is γ a member of all extensions of theory T?

Arguably, the overall complexity should not change if *all extensions* is replaced by *all preferred extensions*. The intuition is that if the complexity does change, then substantial machinery has been added to the underlying formalism in order to implement preferences.

In adding preferences to an approach, the original approach should be changed "minimally", in that by and large, properties of the approach (at least those unrelated to notions of preference) should remain unaffected. This leads to two further criteria.

Is a preferred extension an extension of the theory without preferences? Thus in a default theory with static preferences $(D, W, <)$, one might expect that an extension of this theory also be an extension of the theory without preferences (D, W). For a circumscriptive abnormality theory with preferences, one might expect that its circumscription implies the circumscription without preferences. Similarly, in general a preferred answer set should be also an answer set without preference. However, there are some application domains which require modifications of standard extensions, for example, updating logic programs (Eiter *et al.* 2000) and resolving conflicts caused by classical negation (Buccafurri, Faber, & Leone 1999). In addition, if the preference relation $<$ is empty, the reference theory should have the same extensions with the theory without preference.

---

[5] This isn't totally accurate, since the complexity of various decision problems is known for the major approaches. However, even if we make the eminently reasonable assumption that complexity reflects expressibility, this still says nothing about practical issues.

[6] We prefer the term "extension" to (Brewka & Eiter 1999)'s "belief set". In using "extension" we do not presuppose anything about the underlying system.

Do the properties of the original system remain? This criterion can actually be seen as a collection of criteria: An approach comes with certain formal properties; arguably, the approach with preferences should maintain the same formal properties (unless there is a good reason not to). For example, normal default theories guarantee the existence of extensions. It would seem reasonable that a normal default theory with preferences also guarantee the existence of extensions.

As a second example, in logic programing there are two major semantics for logic programs: answer sets semantics and well-founded semantics. For logic programming without preference, an important property is that the well-founded model provides an approximation to the answer sets semantics. This property should also be preserved in the setting of logic programming with preference.

Approaches

In this section, we cover the salient features of various approaches with respect to how they handle preferences. Approaches are considered in four broad categories: preference in default logic, in logic programming, in updating logic programs, and in other nonmonotonic formalisms.

Preference in default logic

(Baader & Hollunder 1992; Baader & Hollunder 1993b):

preference preference on rules; static preference; strict partial order

strategy selection function on extensions; prescriptive

approach meta-level; integrating preference information into the quasi-induction definition of default extension

complexity same level as host system

distinguished properties (1) preferred extension is also an extension without preference; (2) Brewka and Eiter's Principle I is not satisfied

related work extension of (Brewka & Eiter 1998; Brewka & Eiter 1999)

(Brewka & Eiter 2000):

preference preference on rules; static preference (plus extension to dynamic case); strict partial order

strategy selection function on extensions; semi-prescriptive

approach meta-level; generate all total orderings, each of which is "applied"

complexity same level as host system

distinguished properties (1) preferred extension is also an extension without preference; (2) Brewka and Eiter's Principle I and II are satisfied

related work extension of (Brewka & Eiter 1998; Brewka & Eiter 1999)

(Delgrande & Schaub 2000b):

preference preference on rules; dynamic preference; strict partial order

strategy selection function on extensions; prescriptive

approach meta-level (compiling an ordered default theory into an ordinary one); apply the preference ordering "directly"

complexity same level as host system

distinguished properties (1) preferred extension is also an extension without preference; (2) Brewka and Eiter's Principle I and II are satisfied

related work (Delgrande, Schaub, & Tompits 2000b; Delgrande, Schaub, & Tompits 2000c; Delgrande, Schaub, & Tompits 2002)

(Rintanen 1998):

preference preference on rules; static preference; total order

strategy selection function on extensions; descriptive

approach meta-level; generate all extensions and filter via preference ordering; lexicographic comparison (derive a lexicographic ordering from the total order on defaults); apply the preference ordering "directly"

complexity higher level than host system

distinguished properties Brewka and Eiter's Principle I and II are not satisfied

Preference in logic programming

(Brewka & Eiter 1998; Brewka & Eiter 1999):

host system extended logic programs under answer sets

strategy selection function on answer sets; semi-prescriptive

preference preference on rules; static preference; strict partial order

approach meta-level; generate all total orderings, each of which is "applied"

complexity same level as host system

distinguished properties (1) preferred answer set is also a standard answer set; (2) Brewka and Eiter's Principle I and II are satisfied

related work (Delgrande, Schaub, & Tompits 2000a; Eiter *et al.* 2001) give translation and implementation

(Delgrande, Schaub, & Tompits 2000b; Delgrande, Schaub, & Tompits 2000c; Delgrande, Schaub, & Tompits 2002):

host system extended logic programs under answer sets

strategy selection function on answer sets; prescriptive

preference preference on rules; dynamic preference; strict partial order

approach object level (compiling an ordered logic program into an ordinary one); apply the preference ordering "directly"

complexity same level as host system

distinguished properties (1) preferred answer set is also a standard answer set; (2) Brewka and Eiter's Principle I and II are satisfied

related work (Delgrande & Schaub 2000b)

implementation `www.cs.uni-potsdam.de/~torsten/plp`

(Grosof 1997):

host system extended logic programs under answer sets (no recursion is allowed)

strategy prescriptive

preference preference on rules; dynamic preference; strict partial order

approach meta-level; apply the preference ordering “directly”

complexity same level as host system

distinguished properties each ordered logic program without recursion has a unique preferred answer sets

related work IBM CommonRules project

implementation `ebusiness.mit.edu/bgrosof/`

(Schaub & Wang 2001; Wang, Zhou, & Lin 2000):

host system extended logic programs under answer sets, regular sets, and well-founded model

strategy selection function on answer sets; prescriptive

preference preference on rules; static preference; strict partial order

approach meta-level; apply the preference ordering “directly”; modify the immediate consequence; each semantics is defined as a special class of the alternating fixpoints

complexity same level as host system

distinguished properties (1) preferred answer set is also a standard answer set; (2) the well-founded model is correct wrt the preferred answer sets; (3) Brewka and Eiter’s Principle I and II are satisfied

related work (Baader & Hollunder 1992; Baader & Hollunder 1993b; Schaub & Wang 2002)

implementation `www.cs.uni-potsdam.de/∼torsten/plp`

(Zhang & Foo 1997a):

host system extended logic programs under answer sets

strategy modified answer sets

preference preference on rules; dynamic preference; strict partial order

approach meta-level; program transformation

complexity higher level than host system

distinguished properties Brewka and Eiter’s Principle I and II are satisfied

implementation `www.cit.uws.edu.au/∼yan/plps.html`

(Gelfond & Son 1997):

host system logic programs under answer sets

strategy modified answer sets; prescriptive

preference preference on rules; dynamic preference, arbitrary order

approach object level, meta-interpretation; apply the preference ordering “directly”

complexity same level as host system

(Sakama & Inoue 2000):

host system extended logic programs (with disjunction) under answer sets

strategy selection function on answer sets; descriptive

preference preference on literals; static preference; strict partial order

approach meta-level; generate all extensions and filter via preference ordering

complexity higher level than host system

distinguished properties Brewka and Eiter’s Principle II is violated

(Buccafurri, Leone, & Rullo 1996; Buccafurri, Faber, & Leone 1999; Buccafurri, Leone, & Rullo 1999; Laenens & Vermeir 1990; Leone & Rossi 1993):

host system ordered logic

strategy modified answer sets; prescriptive

preference preference on rules (called inheritance hierarchy); static preference; strict partial order

approach meta-level; apply the preference ordering “directly”

complexity same level as host system

(Kakas, Mancarella, & Dung 1994):

host system Logic programming without negation as failure (LPwNF) – limited form of classical negation

strategy modified answer sets

preference preference on rules; strict preference; strict partial order

approach meta-level; prioritized argumentation; apply the preference ordering “directly”

complexity higher level than host system?

distinguished properties LPwNF can characterize default negation

(Dimopoulos & Kakas 1995):

host system extension of LPwNF

strategy modified answer sets

preference preference on rules; static preference; strict partial order

approach meta-level; prioritized argumentation; apply the preference ordering “directly”

(Pradhan & Minker 1996):

host system definite logic programs

strategy modified answer sets

preference preference on atoms; static preference; strict partial order

approach meta-level; employ preference to resolve conflicts between different logic programs; apply the preference ordering “directly”

(Cui & Swift 2001):

host system logic grammars under well-founded model

strategy prescriptive

preference preference on rules; dynamic preference; strict partial order

approach meta-level; apply the preference ordering “directly”;

complexity same level as host system

implementation
`www.cs.sunysb.edu/∼tswift/interpreters.html`

(Brewka 1996):

host system logic programs under well-founded semantics

strategy prescriptive

preference preference on rules; dynamic preference; strict partial order

approach meta-level; apply the preference ordering “directly”

complexity same level as host system

(Prakken 1997a):

host system logic programs

strategy prescriptive

preference preference on rules; strict partial order

approach meta-level; argumentation-based; apply the preference ordering “directly”

Preference and updating logic programs

(Zhang & Foo 1997b; Zhang & Foo 1998):

host system extended logic programs

strategy modified answer sets

preference preference on rules

approach meta-level; program transformation

distinguished properties describes the update of a logic program using the preference approach of (Zhang & Foo 1997a)

(Alferes & Pereira 2000):

host system dynamic logic programs

strategy semi-prescriptive

preference preference on rules; static preferences; strict partial order

approach meta-level

distinguished properties extends update mechanism of (Alferes *et al.* 1998) by allowing preferences between rules, based on the preference approach of (Brewka & Eiter 1999)

Preference in other nonmonotonic formalisms

(Lifschitz 1985b):

host system circumscription

strategy meta-level; preorder (preferences induce strata)

preference static preference, preference on special-purpose predicates, namely *ab* predicates

approach meta-level; generate all extensions

related work (Chen 1997; Gelfond & Lifschitz 1988; Wakaki & Satoh 1997) provide compilations from preferred circumscription into logic programs

(Inoue & Sakama 1999):

host system abduction

strategy selection function on minimal explanations; descriptive

preference static preference, preference on abducibles (literals)

approach meta-level; generate all extensions; apply the preference ordering “directly”

complexity higher level than host system

related work this semantics is equivalent to the preferred answer sets in (Sakama & Inoue 2000)

(Nute 1987; Nute 1994; Billington 1993; Antoniou *et al.* 2000):

host system defeasible logic

strategy prescriptive

preference preference on rules; static preference; arbitrary order

approach meta-level; integrating preference into resolution procedure

(You, Wang, & Yuan 2001):

host system priority logic (prioritized argumentation)

strategy deriving preference on arguments from rule preference

preference preference on rules; static preference; arbitrary order

approach meta-level; generate all acceptable arguments and select

complexity higher level than host system

related work prioritized argumentation is also investigated in (Dimopoulos & Kakas 1995; Prakken 1997a)

(Ryan 1992):

host system classical logic (ordered theory presentations)

strategy descriptive

preference preference on formulas; static preference; strict partial order

approach meta-level; generate all models and filter via preference ordering; apply the preference ordering “directly”

complexity higher level than host system

Conclusion

We have presented an overview and classification of approaches to dealing with preference in nonmonotonic reasoning. A set of criteria for classifying approaches is first given, followed by a set of desiderata that an approach might be expected to satisfy. A comprehensive set of approaches is subsequently given and classified with respect to these sets of principles. The intent is to provide some structure on the area, so that seemingly unrelated systems may be compared or related with each other. The full version of this paper will also give a higher-level survey and distillation of these approaches. As well, a suggested list of open problems and research topics will be given in the full version.

References

Alferes, J., and Pereira, L. 2000. Updates plus Preferences. In Aciego, M.; de Guzmán, I.; Brewka, G.; and Pereira, L., eds., *Proc. 7th European Workshop on Logics in Artificial Intelligence* (*JELIA 2000*), volume 1919 of *Lecture Notes in Computer Science*, 345–360. Springer.

Alferes, J.; Leite, J.; Pereira, L.; Przymusinska, H.; and Przymusinski, T. 1998. Dynamic Logic Programming. In Cohn, A.; Schubert, L.; and Shapiro, S., eds., *Proc. 6th Int. Conf. on Principles of Knowledge Representation and Reasoning* (*KR'98*), 98–111. Morgan Kaufmann.

Antoniou, G.; Billington, D.; Governatori, G.; and Maher, M. J. 2000. Defeasible logic versus logic programming without negation as failure. *Journal of Logic Programming* 42(1):47–57.

Baader, F., and Hollunder, B. 1992. Embedding defaults into terminological knowledge representation formalisms. In Nebel, B.; Rich, C.; and Swartout, W., eds., *Proceedings of the Third International Conference on the Principles of Knowledge Representation and Reasoning*, 306–317.

Baader, F., and Hollunder, B. 1993a. How to prefer more specific defaults in terminological default logic. In Bajcsy, R., ed., *Proceedings of the International Joint Conference on Artificial Intelligence*, 669–674. Morgan Kaufmann Publishers.

Baader, F., and Hollunder, B. 1993b. How to prefer more specific defaults in terminological default logic. Technical Report RR-92-58, DFKI.

Billington, D. 1993. Defeasible logic is stable. *Journal of Logic and Computation* 3(4):379–400.

Brewka, G., and Eiter, T. 1998. Preferred answer sets for extended logic programs. In Cohn, A.; Schubert, L.; and Shapiro, S., eds., *Proceedings of the Sixth International Conference on the Principles of Knowledge Representation and Reasoning*, 86–97. Morgan Kaufmann Publishers.

Brewka, G., and Eiter, T. 1999. Preferred answer sets for extended logic programs. *Artificial Intelligence* 109(1-2):297–356.

Brewka, G., and Eiter, T. 2000. Prioritizing default logic. In Hölldobler, S., ed., *Intellectics and Computational Logic — Papers in Honour of Wolfgang Bibel*. Kluwer Academic Publishers. 27–45.

Brewka, G. 1994. Adding priorities and specificity to default logic. In Pereira, L., and Pearce, D., eds., *European Workshop on Logics in Artificial Intelligence (JELIA'94)*, Lecture Notes in Artificial Intelligence, 247–260. Springer Verlag.

Brewka, G. 1996. Well-founded semantics for extended logic programs with dynamic preferences. *Journal of Artificial Intelligence Research* 4:19–36.

Buccafurri, F.; Faber, W.; and Leone, N. 1999. Disjunctive logic programs with inheritance. In *Proceedings of the International Conference on Logic Programming*, 79–93. The MIT Press.

Buccafurri, F.; Leone, N.; and Rullo, P. 1996. Stable models and their computation for logic programming with inheritance and true negation. *Journal of Logic Programming* 27:5–43.

Buccafurri, F.; Leone, N.; and Rullo, P. 1999. Semantics and expressiveness of disjunctive ordered logic. *Annals of Mathematics and Artificial Intelligence* 25:311–337.

Chen, J. 1997. Embedding prioritized circumscription into logic programs. In *Proc. of the 10th International symposium on Foundations of Intelligent Systems (ISMIS'97), LNAI 1325*, 50–59. Springer-Verlag.

Cui, B., and Swift, T. 2001. Preference logic grammars: Fixed-point semantics and application to data standardization. *Artificial Intelligence*. To appear.

Delgrande, J., and Schaub, T. 2000a. Expressing preferences in default logic. *Artificial Intelligence* 123(1-2):41–87.

Delgrande, J., and Schaub, T. 2000b. Expressing preferences in default logic. *Artificial Intelligence* 123(1-2):41–87.

Delgrande, J., and Schaub, T. 2000c. The role of default logic in knowledge representation. In Minker, J., ed., *Logic-Based Artificial Intelligence*. Kluwer Academic Publishers. 107–126.

Delgrande, J.; Schaub, T.; and Tompits, H. 2000a. A compilation of Brewka and Eiter's approach to prioritization. In Ojeda-Aciego, M.; Guzmán, I.; Brewka, G.; and Pereira, L., eds., *Proceedings of the European Workshop on Logics in Artificial Intelligence (JELIA 2000)*, volume 1919 of *Lecture Notes in Artificial Intelligence*, 376–390. Springer-Verlag.

Delgrande, J.; Schaub, T.; and Tompits, H. 2000b. Logic programs with compiled preferences. In Baral, C., and Truszczyński, M., eds., *Proceedings of the Eighth International Workshop on Non-Monotonic Reasoning*. `arXiv.org` e-Print archive.

Delgrande, J.; Schaub, T.; and Tompits, H. 2000c. Logic programs with compiled preferences. In Horn, W., ed., *Proceedings of the European Conference on Artificial Intelligence*, 392–398. IOS Press.

Delgrande, J.; Schaub, T.; and Tompits, H. 2002. A framework for compiling preferences in logic programs. *Theory and Practice of Logic Programming*. To appear.

Dimopoulos, Y., and Kakas, C. 1995. Logic programming without negation as failure. In Lloyd, J., ed., *Proceedings of the International Symposium of Logic Programming*, 369–383. The MIT Press.

Eiter, T.; Fink, M.; Sabbatini, G.; and Tompits, H. 2000. Considerations on updates of logic programs. In *Proceedings of the Seventh European Workshop on Logics in Artificial Intelligence (JELIA'2000)*, volume 1919 of *Lecture Notes in Artificial Intelligence*, 2–20. Springer-Verlag.

Eiter, T.; Faber, W.; Leone, N.; and Pfeifer, G. 2001. Computing preferred and weakly preferred answer sets by meta-interpretation in answer set programming. In Provetti, A., and Cao, S., eds., *Proceedings AAAI 2001 Spring Symposium on Answer Set Programming: Towards Efficient and*

Scalable Knowledge Representation and Reasoning, 45–52. AAAI Press.

Gelfond, M., and Lifschitz, V. 1988. Compiling circumscription theories into logic programs. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, 455–459. The MIT Press.

Gelfond, M., and Son, T. 1997. Reasoning with prioritized defaults. In Dix, J.; Pereira, L.; and Przymusinski, T., eds., *Third International Workshop on Logic Programming and Knowledge Representation*, volume 1471 of *Lecture Notes in Computer Science*, 164–223. Springer-Verlag.

Gordon, T. 1993. *The pleading game: An Artificial Intelligence Model of Procedural Justice*. Dissertation, Technische Hochschule Darmstadt, Alexanderstraße 10, D-64283 Darmstadt, Germany.

Grosof, B. 1997. Prioritized conflict handling for logic programs. In Maluszynsk, J., ed., *Logic Programming: Proceedings of the 1997 International Symposium*, 197–211. The MIT Press.

Grosof, B. 1999. Business rules for electronic commerce. `http://www.research.ibm.com/rules/papers.html`. IBM Research.

Inoue, K., and Sakama, C. 1999. Abducing priorities to derive intended conclusions. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 44–49. Morgan Kaufmann Publishers.

Kakas, A.; Mancarella, P.; and Dung, P. M. 1994. The acceptability semantics for logic programs. In Hentenryck, P. V., ed., *Proceedings of the International Conference on Logic Programming*, 504–519. The MIT Press.

Laenens, E., and Vermeir, D. 1990. A fixpoint semantics for ordered logic. *Journal of Logic and Computation* 1(2):159–185.

Leone, N., and Rossi, G. 1993. Well-founded semantics and stratification for ordered logic programs. *New Generation Computing* 12(1):91–121.

Lifschitz, V. 1985a. Closed-world databases and circumscription. *Artificial Intelligence* 27:229–235.

Lifschitz, V. 1985b. Closed-world databases and circumscription. *Artificial Intelligence* 27:229–235.

Marek, V., and Truszczyński, M. 1993. *Nonmonotonic logic: context-dependent reasoning*. Artifical Intelligence. Springer-Verlag.

McCarthy, J. 1986. Applications of circumscription to formalizing common-sense knowledge. *Artificial Intelligence* 28:89–116.

Nute, D. 1987. Defeasible reasoning. In *Proceedings of the 20th Hawaii International Conference on Systems Science*, 470–477. IEEE Press.

Nute, D. 1994. Defeasible logic. In *Handbook of Logics in Artificial Intelligence and Logic Programming, Vol. 3*. Oxford University Press. 353–395.

Pradhan, S., and Minker, J. 1996. Using priorities to combine knowledge bases. *International Journal of Cooperative Information Systems* 5(2-3):333–364.

Prakken, H. 1997a. Argument-based logic programming with defeasible priorities. *Journal of Applied Non-classical Logics* 7:25–75.

Prakken, H. 1997b. *Logical Tools for Modelling Legal Argument*. Kluwer Academic Publishers.

Reiter, R. 1980. A logic for default reasoning. *Artificial Intelligence* 13:81–132.

Rintanen, J. 1998. Lexicographic priorities in default logic. *Artificial Intelligence* 106:221–265.

Ryan, M. 1992. Representing defaults as sentences with reduced priority. In Nebel, B.; Rich, C.; and Swartout, W., eds., *Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning*, 649–660. Morgan Kaufmann Publishers.

Sakama, C., and Inoue, K. 2000. Prioritized logic programming and its application to commonsense reasoning. *Artificial Intelligence* 123(1-2):185–222.

Schaub, T., and Wang, K. 2001. A comparative study of logic programs with preference. In Nebel, B., ed., *Proceedings of the International Joint Conference on Artificial Intelligence*, 597–602. Morgan Kaufmann Publishers.

Schaub, T., and Wang, K. 2002. Preferred well-founded semantics for logic programming by alternating fixpoints: preliminary report. In *Proc. of the Workshop on NMR'2002*, to appear.

Wakaki, T., and Satoh, K. 1997. Compiling circumscription into extended logic programs. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 182–187. Morgan Kaufmann Publishers.

Wang, K.; Zhou, L.; and Lin, F. 2000. Alternating fixpoint theory for logic programs with priority. In *Proceedings of the First International Conference on Computational Logic*, volume 1861 of *Lecture Notes in Computer Science*, 164–178. Springer-Verlag.

You, J.; Wang, X.; and Yuan, L. 2001. Nonmonotonic reasoning as prioritized argumentation. *IEEE Transactions on Knowledge and Data Engineering* 13(6):968–979.

Zhang, Y., and Foo, N. 1997a. Answer sets for prioritized logic programs. In Maluszynski, J., ed., *Proceedings of the International Symposium on Logic Programming (ILPS-97)*, 69–84. MIT Press.

Zhang, Y., and Foo, N. 1997b. Towards Generalized Rule-based Updates. In *Proc. 15th Int. Joint Conf. on Artificial Intelligence* (*IJCAI'97*), volume 1, 82–88. Morgan Kaufmann.

Zhang, Y., and Foo, N. 1998. Updating Logic Programs. In Prade, H., ed., *Proc. 13th Europ. Conf. on Artificial Intelligence* (*ECAI'98*), 403–407. Wiley.