# Preference Relations by Approximation

**Mario Alviano**
University of Calabria, Italy
alviano@mat.unical.it

**Javier Romero**
University of Potsdam, Germany
javier@cs.uni-potsdam.de

**Torsten Schaub**
University of Potsdam, Germany
torsten@cs.uni-potsdam.de

**Abstract**

Declarative languages for knowledge representation and reasoning provide constructs to define preference relations over the set of possible interpretations, so that preferred models represent optimal solutions of the encoded problem. We introduce the notion of approximation for replacing preference relations with stronger preference relations, that is, relations comparing more pairs of interpretations. Our aim is to accelerate the computation of a non-empty subset of the optimal solutions by means of highly specialized algorithms. We implement our approach in Answer Set Programming (ASP), where problems involving quantitative and qualitative preference relations can be addressed by ASPRIN, implementing a generic optimization algorithm. Unlike this, chains of approximations allow us to reduce several preference relations to the preference relations associated with ASP's native weak constraints and heuristic directives. In this way, ASPRIN can now take advantage of several highly optimized algorithms implemented by ASP solvers for computing optimal solutions.

## Introduction

Languages for knowledge representation and reasoning provide syntactic constructs for encoding common sense knowledge, and are equipped with formal semantics so that each model of the instance in input describes a plausible scenario for the encoded knowledge (van Harmelen, Lifschitz, and Porter 2008). Scenarios differ from each other, and several *preference relations* may be applied to them depending on properties that are more desirable. Preference relations can be *quantitative*, *qualitative*, or a combination of them.

Examples of quantitative preferences are weighted and unweighted *weak constraints* (Buccafurri, Leone, and Rullo 2000; Simons, Niemelä, and Soininen 2002), a construct widely used in Answer Set Programming (ASP) (Gelfond and Lifschitz 1991); weak constraints may also specify levels, so to obtain a lexicographic combination of cardinality and weight preferences. Examples of qualitative preferences are subset and superset minimality (Di Rosa, Giunchiglia, and Maratea 2010), the former preference relation underlying circumscription (McCarthy 1980; Lifschitz 1986).

These preference relations have been widely studied, and very efficient algorithms have been developed for them. This is not the case for other preference relations, among them *answer set optimization* (aso) (Brewka, Niemelä, and Truszczynski 2003), *partially ordered set* (poset) (Rosa and Giunchiglia 2013), and several composite preferences. Actually, efficient algorithms are either specific of a formalism and a preference relation, as for example for weak constraints in ASP, or obtained by mapping the preference relation of interest to another, as done for example by CIRCUMSCRIPTINO to enumerate circumscribed models by computing cardinality optimal models (Alviano 2017).

The aim of this paper is to generalize the mapping approach to other common preference relations. For this purpose, knowledge bases and preference relations are first considered at a semantic level: knowledge bases are sets of models; preference relations are partial orders over the power set of a domain of interest, and can be associated with specific knowledge bases in order to define iterative algorithms for computing preferred models of the knowledge base in input.

Working at a semantic level makes clear the similarities between different preference relations. For example, less-cardinality preferences are *supersets* of subset preferences. Such a property provides an alternative proof for the correctness of the algorithm implemented by CIRCUMSCRIPTINO for computing subset optimal models. Actually, this property can be further generalized by adding an *upstream expansion function* for introducing auxiliary symbols. Specifically, some mappings between different preference relations require the introduction of auxiliary atoms, and expansion functions impose that truth values of auxiliary atoms are uniquely determined by truth values of original atoms. We show how the resulting new notion, referred to as *approximation*, can be applied to several preferences, namely cardinality, weight, aso, poset, and many composite preferences. Interestingly, different approximations can be applied sequentially, so that all considered preferences can be mapped to a single kind of preferences, namely lexicographic composition of less-weight preferences, viz., the preference associated with weak constraints.

Since weak constraints are the native ASP construct to encode preferences, all algorithms developed and implemented in efficient ASP solvers are hence available for computing preferred models of all preference relations considered in this paper. In fact, the notion of approximation is implemented in ASPRIN (Brewka et al. 2015b), a general purpose logic-based system where knowledge bases and preference relations are encoded by ASP programs. The implementa-

$$\begin{tikzcd}
\text{and} \arrow[r] & \text{lexico} & \text{pareto} \arrow[l] \\
\text{less-weight} \arrow[ur, dashed] & \text{more-weight} \arrow[l] & \text{poset} \arrow[ul] \\
\text{subset} \arrow[u, dotted] & \text{superset} \arrow[u, dotted] \arrow[l, dashed] & \text{neg(less-weight)} \arrow[ul, dotted] \\
 & \text{aso} \arrow[u] & \text{neg(subset)} \arrow[ul, dashed]
\end{tikzcd}$$

Figure 1: Approximations to normalize common preference relations in lexicographic composition of less-weight (dotted and solid arrows) or subset preferences (dashed and solid arrows). An arrow from $\alpha$ to $\beta$ denotes that preference $\alpha$ is approximated by preference $\beta$.

tion is assessed empirically, comparing different algorithms of CLASP (Gebser et al. 2015). Lexicographic composition of subset preferences is also natively supported by CLASP via `#heuristic` directives (Gebser et al. 2013c). Hence, yet another solving strategy is obtained by introducing approximations targeting lexicographic composition of subset preferences. These mapping are interesting from two points of view. Theoretically, they provide mapping from qualitative to quantitative preferences, and vice versa. Pragmatically, they provide two normal forms associated with several computational strategies.

A summary of the approximations to obtain the two normal forms mentioned above is given in Figure 1. Intuitively, the normalization procedures work bottom-up on the tree structure of the preference relation in input, and apply approximations to all nodes that are not already in normal form. For example, aso is approximated by superset, which in turn is approximated by either more-weight or subset, depending on the desired normal form.

## Knowledge Bases and Preferences

A *knowledge base* $\Gamma$ is a pair $(\mathcal{A}, \mathcal{M})$ such that $\mathcal{A}$ is a (finite) set of (propositional) atoms, and $\mathcal{M}$ is a (finite) set of subsets of $\mathcal{A}$. Set $\mathcal{A}$ is the *domain* of $\Gamma$, and each set in $\mathcal{M}$ is a *model* of $\Gamma$. The knowledge base $\Gamma$ is *consistent* if $\mathcal{M} \neq \emptyset$, otherwise $\Gamma$ is *inconsistent*. The *projection* of a knowledge base $(\mathcal{A}, \mathcal{M})$ on a set $\mathcal{A}' \subseteq \mathcal{A}$ is the knowledge base $(\mathcal{A}', \{I \cap \mathcal{A}' \mid I \in \mathcal{M}\})$.

For a set $\mathcal{A}$ of atoms, let $\mathcal{A}^?$ denote the set $\{p^? \mid p \in \mathcal{A}\}$, and for all $n \geq 0$ let $\mathcal{A}^n$ denote the set $\{p^n \mid p \in \mathcal{A}\}$, where $p^?$ and $p^n$ are fresh atoms. For a knowledge base $\Gamma$, and $n \geq 0$, let $\Gamma^n$ denote the knowledge base obtained from $\Gamma$ by replacing each atom $p^?$ with $p^n$. (Intuitively, $p^?$ is a placeholder to be replaced with some $p^n$.)

The *join* $\Gamma \bowtie \Gamma'$ of two knowledge bases $\Gamma = (\mathcal{A}, \mathcal{M})$, $\Gamma' = (\mathcal{A}', \mathcal{M}')$ is defined as follows:

$$\Gamma \bowtie \Gamma' := (\mathcal{A} \cup \mathcal{A}', \{I \cup J \mid I \in \mathcal{M},\ J \in \mathcal{M}', \\ I \cap \mathcal{A}' = J \cap \mathcal{A}\}).$$

Intuitively, models of $\Gamma$ and $\Gamma'$ are combined if they agree on the shared domain $\mathcal{A} \cap \mathcal{A}'$. (In fact, note that $I \cap \mathcal{A} \cap \mathcal{A}' = I \cap \mathcal{A}'$ since $I \subseteq \mathcal{A}$, and $J \cap \mathcal{A} \cap \mathcal{A}' = J \cap \mathcal{A}$ since $J \subseteq \mathcal{A}'$.)

**Algorithm 1:** $solveOpt(\Gamma$ : knowledge base $(\mathcal{A}, \mathcal{M})$, $\succeq$ : preference relation over $\mathcal{A})$

```
1 $I := \bot$;
2 $J := solve(\Gamma)$;
3 while $J \neq \bot$ do
4     $I := J \cap \mathcal{A}$;
5     $J := solve(\Gamma \bowtie [\succ] \bowtie (\mathcal{A}^?, \{I^?\}))$;
6 return $I$;
```

**Example 1.** The join of $(\{a,b\}, \{\{a\},\{b\}\})$ and $(\{b,c\}, \{\emptyset, \{b,c\}, \{c\}\})$ is $(\{a,b,c\}, \{\{a\}, \{a,c\}, \{b,c\}\})$. Indeed, $\{a\}$ is obtained by combining $\{a\}$ and $\emptyset$, $\{a,c\}$ is obtained by combining $\{a\}$ and $\{c\}$, and finally $\{b,c\}$ is obtained by combining $\{b\}$ and $\{b,c\}$. ■

A *preference relation* $\succeq$ over a domain $\mathcal{A}$ is a partial order over $2^{\mathcal{A}}$, that is, $\succeq$ is a subset of $2^{\mathcal{A}} \times 2^{\mathcal{A}}$, and $\succeq$ is reflexive, antisymmetric and transitive. (Note that $\succeq$ also defines a preference relation over any superset of $\mathcal{A}$ in the obvious way, that is, $I \succeq J$ if and only if $I \cap \mathcal{A} \succeq J \cap \mathcal{A}$.) Let $\succ$ be the strict partial order obtained from $\succeq$, that is, $I \succ J$ if and only if $I \succeq J$ and $J \not\succeq I$. (Recall that a strict partial order is an irreflexive and transitive relation.)

Let $\Gamma = (\mathcal{A}, \mathcal{M})$ be a knowledge base, and $\succeq$ be a preference relation over $\mathcal{A}$. $I \in \mathcal{M}$ is a $\succeq$*-optimal* model of $\Gamma$ if there is no $J \in \mathcal{M}$ such that $J \succ I$.

**Example 2.** Let $\Gamma$ be $(\{a,b,c,d\}, \mathcal{M})$, where $\mathcal{M}$ comprises $\{a,b\}$, $\{b,c\}$, $\{d\}$, and their supersets. Let $\subseteq_{\{a,b,c,d\}}$ be such that $I \succeq J$ if and only if $I \cap \{a,b,c,d\} \subseteq J \cap \{a,b,c,d\}$. The $\subseteq_{\{a,b,c,d\}}$-optimal models of $\Gamma$ are $\{a,b\}$, $\{b,c\}$, and $\{d\}$. ■

## Iterative Algorithms

Let $\succeq$ be a preference relation on domain $\mathcal{A}$. Let $[\succ]$ denote $(\mathcal{A} \cup \mathcal{A}^?, \{I \cup J^? \mid I \succ J\})$, the *preference knowledge base* for $\succeq$. Similarly, let $[\not\succ]$ denote $(\mathcal{A} \cup \mathcal{A}^?, \{I^? \cup J \mid I \not\succ J\})$, the *complementary preference knowledge base* for $\succeq$. Note that, for all $I, J \subseteq \mathcal{A}$, $I \succ J$ if and only if $[\succ] \bowtie (\mathcal{A}, \{I\}) \bowtie (\mathcal{A}^?, \{J^?\})$ is consistent, and $I \not\succ J$ if and only if $[\not\succ] \bowtie (\mathcal{A}^?, \{I^?\}) \bowtie (\mathcal{A}, \{J\})$ is consistent.

**Example 3** (Continuing Example 2)**.** Let $[\subset_{\{a,b,c,d\}}]$ be the preference knowledge base for $\subseteq_{\{a,b,c,d\}}$, subset on domain $\{a,b,c,d\}$. Hence, $\{a,b\} \cup \{a,b,c\}^?$ is a model of $[\subset_{\{a,b,c,d\}}]$, while $\{a,b\} \cup \{b,c\}^?$ is not. Let $[\not\subset_{\{a,b,c,d\}}]$ be the complementary preference knowledge base for $\subseteq_{\{a,b,c,d\}}$. Then, $\{a,b\}^? \cup \{b,c\}$ is a model of $[\not\subset_{\{a,b,c,d\}}]$, while $\{a,b\}^? \cup \{a,b,c\}$ is not. ■

Preference knowledge bases offer an intuitive procedure to verify the optimality of models via a consistency check (with certificate).

**Proposition 1.** *Let* $\Gamma = (\mathcal{A}, \mathcal{M})$ *be a knowledge base, and* $\succeq$ *be a preference relation over* $\mathcal{A}$*.* $I \in \mathcal{M}$ *is a* $\succeq$*-optimal model of* $\Gamma$ *iff* $\Gamma \bowtie [\succ] \bowtie (\mathcal{A}^?, \{I^?\})$ *is inconsistent.*

Additionally, when the consistency check actually returns a model $J$, it is guaranteed that $J \succ I$, which provides

**Algorithm 2:** *enumOpt*($\Gamma$ : knowledge base $(\mathcal{A}, \mathcal{M})$, $\succeq$ : preference relation over $\mathcal{A}$)

```
1 for $n \in \mathbb{N}^+$ do
2     $I := solveOpt(\Gamma, \succeq)$;
3     if $I = \bot$ then break;
4     print $I$;
5     $\Gamma := \Gamma \bowtie (\mathcal{A}, 2^{\mathcal{A}} \setminus \{I\}) \bowtie [\not\succ]^n \bowtie (\mathcal{A}^n, \{I^n\})$;
```

an iterative sat-unsat algorithm shown as Algorithm 1. The algorithm searches for a first model by means of procedure *solve*, which returns a model of the knowledge base provided in input if it exists, and $\bot$ otherwise. The initial model $I$, if any, is possibly improved by calling *solve* on $\Gamma \bowtie [\succ] \bowtie (\mathcal{A}^?, \{I^?\})$, and this improvement is repeated until $\bot$ is returned. The last computed model is then the output of the algorithm, a $\succeq$-optimal model of $\Gamma$.

**Example 4** (Continuing Example 3)**.** Consider $solveOpt(\Gamma, \subseteq_{\{a,b,c,d\}})$. Suppose that $solve(\Gamma)$ returns $\{a, b, c\}$. Algorithm 1 then calls *solve* for the knowledge base $\Gamma \bowtie [\subset_{\{a,b,c,d\}}] \bowtie (\{a, b, c, d\}^?, \{\{a, b, c\}^?\})$, where $[\subset_{\{a,b,c,d\}}] \bowtie (\{a, b, c, d\}^?, \{\{a, b, c\}^?\})$ has models of the form $I \cup \{a, b, c\}^?$, for all $I \subset \{a, b, c\}$. A model, say $\{a, b\} \cup \{a, b, c\}^?$, is returned, and the knowledge base $\Gamma \bowtie [\subset_{\{a,b,c,d\}}] \bowtie (\{a, b, c, d\}^?, \{\{a, b\}^?\})$ is processed. Function *solve* returns $\bot$, and the algorithm terminates giving in output $\{a, b\}$, a $\subseteq_{\{a,b,c,d\}}$-optimal model of $\Gamma$. ■

In order to compute a second $\succeq$-optimal model, and therefore obtain an enumeration algorithm, a few interpretations have to be blocked. First of all, previously computed $\succeq$-optimal models have to be blocked; to this aim, for an interpretation $I \subseteq \mathcal{A}$, the following knowledge base can be used: $(\mathcal{A}, 2^{\mathcal{A}} \setminus \{I\})$. Moreover, interpretations being less preferred than a previously computed $\succeq$-optimal model $I$ have to be discarded as well; this is achieved by the knowledge base $([\not\succ] \bowtie (\mathcal{A}^?, \{I^?\}))^n = [\not\succ]^n \bowtie (\mathcal{A}^n, \{I^n\})$. The resulting enumeration procedure is reported as Algorithm 2.

**Example 5** (Continuing Example 4)**.** After model $\{a, b\}$ is computed, $\Gamma$ is joined with $(\{a, b, c, d\}, 2^{\{a,b,c,d\}} \setminus \{\{a, b\}\})$ to discard $\{a, b\}$, and with $[\not\subset_{\{a,b,c,d\}}]^1 \bowtie (\{a, b, c, d\}^1, \{\{a, b\}^1\})$ to discard supersets of $\{a, b\}$. The next call to *solveOpt* returns a second $\subseteq_{\{a,b,c,d\}}$-optimal model, say $\{a, c\}$, and $\Gamma$ is joined with $(\{a, b, c, d\}, 2^{\{a,b,c,d\}} \setminus \{\{a, c\}\})$ to discard $\{a, c\}$, and with $[\not\subset_{\{a,b,c,d\}}]^2 \bowtie (\{a, b, c, d\}^2, \{\{a, c\}^2\})$ to discard supersets of $\{a, c\}$. At this point, the only models of $\Gamma$ are $\{a, d\}$, $\{b, d\}$, $\{c, d\}$, and $\{d\}$. The latter is actually returned by the next call to *solveOpt*, so $\Gamma$ is joined with $(\{a, b, c, d\}, 2^{\{a,b,c,d\}} \setminus \{\{d\}\})$ to discard $\{d\}$, and with $[\not\subset_{\{a,b,c,d\}}]^3 \bowtie (\{a, b, c, d\}^3, \{\{d\}^3\})$ to discard supersets of $\{d\}$. Hence, $\Gamma$ is now inconsistent, and the algorithm terminates. ■

## Preference Approximation

In Algorithm 1, the preference knowledge base $[\succ]$ forces procedure *solve* to search for a model being more preferred than the last computed model. A stronger preference relation can be also used for this purpose, and is the underlying idea of preference approximations.

Let $\mathcal{A}, \mathcal{A}'$ be sets of atoms such that $\mathcal{A} \subseteq \mathcal{A}'$. An *expansion function* from $\mathcal{A}$ to $\mathcal{A}'$ is a function $e : 2^{\mathcal{A}} \to 2^{\mathcal{A}'}$ such that $e(I) \cap \mathcal{A} = I$, for all $I \subseteq \mathcal{A}$. Abusing of notation, for a knowledge base $\Gamma = (\mathcal{A}, \mathcal{M})$, let $e(\Gamma)$ be the knowledge base $(\mathcal{A}', \{e(I) \mid I \in \mathcal{M}\})$. Let $id : 2^{\mathcal{A}} \to 2^{\mathcal{A}}$ be the identity expansion function, that is, $id(I) = I$ for any interpretation $I$.

Let $\succeq$ be a preference relation for $\mathcal{A}$, and $e$ be an expansion function from $\mathcal{A}$ to $\mathcal{A}'$. A preference relation $\succeq'$ is an approximation of $\succeq$ with respect to $e$, or *e-approximation*, if $I \succ J$ implies $e(I) \succ' e(J)$, for all $I, J \subseteq \mathcal{A}$. If also $e(I) \succ' e(J)$ implies $I \succ J$, then $\succeq'$ is a reduction of $\succeq$ with respect to $e$, or *e-reduction*. First of all, note that expansion functions, approximations and reductions are closed under composition.

**Theorem 1.** *Let $e : 2^{\mathcal{A}} \to 2^{\mathcal{A}'}$, $e' : 2^{\mathcal{A}'} \to 2^{\mathcal{A}''}$ be expansion functions. Their composition $e' \circ e : 2^{\mathcal{A}} \to 2^{\mathcal{A}''}$ is an expansion function. Moreover, if $\succeq'$ is an e-approximation of $\succeq$, and $\succeq''$ is an $e'$-approximation of $\succeq'$, then $\succeq''$ is an $(e' \circ e)$-approximation of $\succeq$.*

**Corollary 1.** *If $\succeq'$ is an e-reduction of $\succeq$, and $\succeq''$ is an $e'$-reduction of $\succeq'$, then $\succeq''$ is an $(e' \circ e)$-reduction of $\succeq$.*

Intuitively, an expansion function $e$ and an $e$-approximation $\succeq'$ can be used to compute some $\succeq$-optimal models of a knowledge base $\Gamma$ by focusing on $\succeq'$-optimal models of $e(\Gamma)$, as formalized by the following theorem.

**Theorem 2.** *Let $\Gamma = (\mathcal{A}, \mathcal{M})$ be a knowledge base, $\succeq$ be a preference relation over $\mathcal{A}$, $e : 2^{\mathcal{A}} \to 2^{\mathcal{A}'}$ be an expansion function, and $\succeq'$ be an e-approximation of $\succeq$. For any $I \subseteq \mathcal{A}$, if $e(I) \subseteq \mathcal{A}'$ is a $\succeq'$-optimal model of $e(\Gamma)$, then $I$ is a $\succeq$-optimal model of $\Gamma$.*

According to the theorem above, only some optimal models are preserved in general by an approximation. On the other hand, reductions preserves all optimal models, as formalized by the following corollary of Theorem 2.

**Corollary 2.** *If $\succeq'$ is an e-reduction of $\succeq$, then $I$ is a $\succeq$-optimal model of a knowledge base $\Gamma$ if and only if $e(I) \subseteq \mathcal{A}'$ is a $\succeq'$-optimal model of $e(\Gamma)$.*

It turns out that $\succeq$-optimal models of $\Gamma = (\mathcal{A}, \mathcal{M})$ can be enumerated by a procedure similar to Algorithm 2, where $\Gamma$ is replaced by $e(\Gamma)$, and $\succeq'$ is used to improve models instead of $\succeq$, where $e$ is any expansion function from $\mathcal{A}$, and $\succeq'$ is any $e$-approximation of $\succeq$. Such a procedure is reported as Algorithm 3. Note that $\Gamma$ is in fact replaced by $e(\Gamma)$ (line 1), and $\succeq'$ is used to improve models (line 3). Finally, optimal models are blocked by $(\mathcal{A}, 2^{\mathcal{A}} \setminus \{I \cap \mathcal{A}\})$, and less preferred models are discarded by $([\not\succ] \bowtie (\mathcal{A}^?, \{I^? \cap \mathcal{A}^?\}))^n = [\not\succ]^n \bowtie (\mathcal{A}^n, \{I^n \cap \mathcal{A}^n\})$.

**Example 6** (Continuing Example 2)**.** Let $\leq^{card}_{\{a,b,c,d\}}$ be the preference relation such that, for all $I, J \subseteq \{a, b, c, d\}$,

**Algorithm 3:** $enumOptApprox(\Gamma$ : knowledge base $(\mathcal{A}, \mathcal{M})$, $\succeq$ : preference relation over $\mathcal{A}$, $e : 2^{\mathcal{A}} \to 2^{\mathcal{A}'}$, $\succeq'$ : $e$-approximation of $\succeq)$

```
1 $\Gamma := e(\Gamma)$;
2 for $n \in \mathbb{N}^+$ do
3     $I := solveOpt(\Gamma, \succeq')$;
4     if $I = \bot$ then break;
5     print $I \cap \mathcal{A}$;
6     $\Gamma := \Gamma \bowtie (\mathcal{A}, 2^{\mathcal{A}} \setminus \{I \cap \mathcal{A}\}) \bowtie [\not\succ]^n \bowtie (\mathcal{A}^n, \{I^n \cap \mathcal{A}^n\})$;
```

**Algorithm 4:** $enumOptApprox2(\Gamma$ : knowledge base $(\mathcal{A}, \mathcal{M})$, $\succeq$ : preference relation over $\mathcal{A}$, $e : 2^{\mathcal{A}} \to 2^{\mathcal{A}'}$, $\succeq'$ : $e$-approximation of $\succeq)$

```
1 $\Gamma := e(\Gamma)$;
2 $n := 0$;
3 repeat
4     $X := \{I \mid I$ is printed by $enumOpt(\Gamma, \succeq')\}$;
5     for $I \in X$ do
6         print $I \cap \mathcal{A}$;
7         $n := n + 1$;
8         $\Gamma := \Gamma \bowtie (\mathcal{A}, 2^{\mathcal{A}} \setminus \{I \cap \mathcal{A}\}) \bowtie [\not\succ]^n \bowtie (\mathcal{A}^n, \{I^n \cap \mathcal{A}^n\})$;
9 until $X = \emptyset$;
```

$I \leq^{card}_{\{a,b,c,d\}} J$ if and only if the cardinality of $I \cap \{a, b, c, d\}$ is smaller than or equal to the cardinality of $J \cap \{a, b, c, d\}$. Hence, $\leq^{card}_{\{a,b,c,d\}}$ is an $id$-approximation of $\subseteq_{\{a,b,c,d\}}$, and Algorithm 3 can be executed. The first call to $solveOpt$ must return $\{d\}$, a $\leq^{card}_{\{a,b,c,d\}}$-optimal model of $\Gamma$, and therefore a $\subseteq_{\{a,b,c,d\}}$-optimal model of $\Gamma$. In order to compute a second $\subseteq_{\{a,b,c,d\}}$-optimal model, the knowledge base is joined with $(\{a, b, c, d\}, 2^{\{a,b,c,d\}} \setminus \{\{d\}\})$ to discard $\{d\}$, and with $[\not<^{card}_{\{a,b,c,d\}}]^1 \bowtie (\{a, b, c, d\}^1, \{\{d\}^1\})$ to discard supersets of $\{d\}$. The next call to $solveOpt$ must then return either $\{a, b\}$ or $\{b, c\}$. ■

An alternative enumeration algorithm can be obtained by first computing all $\succeq'$-optimal models of $e(\Gamma)$, for example by invoking $enumOpt(e(\Gamma), \succeq')$; models are then collected and processed in block with $\succeq$ to discard less preferred models. Algorithm 4 implements such a strategy. Note that the algorithm repeatedly calls $enumOpt$ on $e(\Gamma)$ and $\succeq'$, but any enumeration procedure for $\succeq'$-optimal models of $e(\Gamma)$ can be equivalently employed. Note also that a single call to $enumOpt$ is sufficient in the special case of $e$-reductions.

The remainder of this section introduces approximations for common preference relations, and a bottom-up algorithm to normalize them in lexicographic composition of less-weight or subset preferences.

## Simple Preferences

For $w : \mathcal{A} \to \mathbb{Z}$ being a function mapping atoms into integers, let $\leq^w_{\mathcal{A}}$ denote the *less-weight* preference over $\mathcal{A}$, that is, $I \leq^w_{\mathcal{A}} J$ if and only if $\sum_{p \in I \cap \mathcal{A}} w(p) \leq \sum_{p \in J \cap \mathcal{A}} w(p)$. Similarly, let $\geq^w_{\mathcal{A}}$ denote the *more-weight* preference over $\mathcal{A}$, i.e., $I \geq^w_{\mathcal{A}} J$ if and only if $\sum_{p \in I \cap \mathcal{A}} w(p) \geq \sum_{p \in J \cap \mathcal{A}} w(p)$. Finally, let $-w$ be the function mapping each $p \in \mathcal{A}$ to $-1 \cdot w(p)$.

**Proposition 2.** *Let $\mathcal{A}$ be a set of atoms, and $w : \mathcal{A} \to \mathbb{Z}$ be a function. Preference $\leq^{-w}_{\mathcal{A}}$ is an $id$-reduction of $\geq^w_{\mathcal{A}}$, and preference $\geq^{-w}_{\mathcal{A}}$ is an $id$-reduction of $\leq^w_{\mathcal{A}}$.*

For a set $\mathcal{A}$ of atoms, let $\subseteq_{\mathcal{A}}$ denote the *subset* preference over $\mathcal{A}$, that is, $I \subseteq_{\mathcal{A}} J$ if and only if $I \cap \mathcal{A} \subseteq J \cap \mathcal{A}$. Similarly, let $\supseteq_{\mathcal{A}}$ denote the *superset* preference over $\mathcal{A}$, that is, $I \supseteq_{\mathcal{A}} J$ if and only if $I \cap \mathcal{A} \supseteq J \cap \mathcal{A}$. Let $card : \mathcal{A} \to \{1\}$ map atoms to the constant 1, and let $\leq^{card}_{\mathcal{A}}$ and $\geq^{card}_{\mathcal{A}}$ be also called *less-cardinality* and *more-cardinality* preference over $\mathcal{A}$, respectively.

**Proposition 3.** *Let $\mathcal{A}$ be a set of atoms. Preference $\leq^{card}_{\mathcal{A}}$ is an $id$-approximation of $\subseteq_{\mathcal{A}}$, and $\geq^{card}_{\mathcal{A}}$ is an $id$-approximation of $\supseteq_{\mathcal{A}}$.*

Subset and superset preference relations are also linked by the following $e$-reductions.

**Proposition 4.** *Let $\mathcal{A}$ be a set of atoms, $\mathcal{A}^{neg}$ be $\{neg_p \mid p \in \mathcal{A}\}$, where each $neg_p$ is a fresh atom. Preference $\supseteq_{\mathcal{A}^{neg}}$ is an $e^{\mathcal{A}^{neg}}$-reduction of $\subseteq_{\mathcal{A}}$, and $\subseteq_{\mathcal{A}^{neg}}$ is an $e^{\mathcal{A}^{neg}}$-reduction of $\supseteq_{\mathcal{A}}$, where $e^{\mathcal{A}^{neg}} : I \mapsto I \cup \{neg_p \mid p \in \mathcal{A} \setminus I\}$.*

## Answer Set Optimization (aso)

Let $\pi$ be a statement of the form $x_1 > \cdots > x_n : x$, where $x, x_1, \ldots, x_n$ ($n \geq 1$) are atoms in a set $\mathcal{A}$. For $I \subseteq \mathcal{A}$, let $v_\pi(I)$ be 1 if $x \notin I$ or $x_i \notin I$ for all $i \in \{1, \ldots, n\}$, and otherwise let $v_\pi(I)$ be $i$, where $i \in \{1, \ldots, n\}$ is the smallest integer such that $x_i \in I$. For $\boldsymbol{\pi}$ being a set of statements of the form $x_1 > \cdots > x_n : x$, let $aso^{\boldsymbol{\pi}}$ denote the following *aso preference relation*: $\{(I, J) \mid I, J \subseteq \mathcal{A}, v_\pi(I) \geq v_\pi(J)$ for all $\pi \in \boldsymbol{\pi}\}$, where $\mathcal{A}$ is the set of atoms occurring in $\boldsymbol{\pi}$.

For $\pi$ of the form $x_1 > \cdots > x_n : x$, let $\mathcal{A}^\pi$ be the set $\{x^\pi_i \mid i \in \{1, \ldots, n\}\}$, where each $x^\pi_i$ is a fresh atom, and let $e^\pi$ be the expansion $I \mapsto I \cup \{x^\pi_i \mid i \in \{1, \ldots, n\},\ v_\pi(I) \leq i\}$. For a set $\boldsymbol{\pi}$ of statements, let $\mathcal{A}^{\boldsymbol{\pi}}$ be the union of all $\mathcal{A}^\pi$, for $\pi \in \boldsymbol{\pi}$, and $e^{\boldsymbol{\pi}}$ be the expansion function obtained by composing all $e^\pi$, in any order.

**Theorem 3.** *Let $aso^{\boldsymbol{\pi}}$ be an aso preference relation over $\mathcal{A}$. Hence, $\supseteq_{\mathcal{A}^{\boldsymbol{\pi}}}$ is an $e^{\boldsymbol{\pi}}$-reduction of $aso^{\boldsymbol{\pi}}$.*

## Partially Ordered Set (poset)

Let $\succ$ be a strict partial order over a set $\mathcal{A}$ of atoms, that is, $\succ$ is a subset of $\mathcal{A} \times \mathcal{A}$, $\succ$ is irreflexive ($p \not\succ p$ for all $p \in \mathcal{A}$), and $\succ$ is transitive ($p_1 \succ p_2$ and $p_2 \succ p_3$ implies $p_1 \succ p_3$, for all $p_1, p_2, p_3 \in \mathcal{A}$). The strict partial order $\succ$ defines a preference relation over $\mathcal{A}$; specifically, $poset^{\succ}$ is the following preference relation:

$$\{(I, I) \mid I \subseteq \mathcal{A}\} \cup \{(I, J) \mid I, J \subseteq \mathcal{A}, I \setminus J \neq \emptyset,$$
$$\text{for all } p \in J \setminus I \text{ there is } p' \in I \setminus J \text{ such that } p' \succ p\}.$$

An $id$-reduction of poset to lexicographic composition of superset preferences will be given in the next section.

## Composite Preferences

Let $\succeq$ be a preference relation over $\mathcal{A}$. Let $neg(\succeq)$ be the following preference relation over $\mathcal{A}$: $\{(I, J) \mid I, J \subseteq \mathcal{A}, J \succeq I\}$. The following statement links the negation of simple preferences by means of $id$-approximations.

**Proposition 5.** *Preference relations $\subseteq_{\mathcal{A}}$ and $\supseteq_{\mathcal{A}}$ are id-reductions of $neg(\supseteq_{\mathcal{A}})$ and $neg(\subseteq_{\mathcal{A}})$, respectively. Similarly, for any $w : \mathcal{A} \to \mathbb{Z}$, $\leq^w_{\mathcal{A}}$ and $\geq^w_{\mathcal{A}}$ are id-reductions of $neg(\geq^w_{\mathcal{A}})$ and $neg(\leq^w_{\mathcal{A}})$, respectively.*

Let $\succeq_1, \dots, \succeq_n$ be preference relations over $\mathcal{A}_1, \dots, \mathcal{A}_n$, respectively, and $\mathcal{A}$ be $\mathcal{A}_1 \cup \dots \cup \mathcal{A}_n$. Let $and(\succeq_1, \dots, \succeq_n)$ be the following preference relation over $\mathcal{A}$:

$$\{(I, J) \mid I, J \subseteq \mathcal{A},\ I \succeq_i J \text{ and } J \succeq_i I\ \forall i \in \{1, \dots, n\}\}$$
$$\cup\, \{(I, J) \mid I, J \subseteq \mathcal{A},\ I \succ_i J\ \forall i \in \{1, \dots, n\}\}.$$

Let $lexico(\succeq_1)$ be $\succeq_1$, and $lexico(\succeq_1, \dots, \succeq_n)$ be

$$\{(I, J) \mid I, J \subseteq \mathcal{A}, I \succeq_1 J, \text{ and either } I \succ_1 J \text{ or } (I, J) \in lexico(\succeq_2, \dots, \succeq_n)\},$$

for $n \geq 2$. Let $pareto(\succeq_1, \dots, \succeq_n)$ be

$$\{(I, J) \mid I, J \subseteq \mathcal{A},\ I \succeq_i J\ \forall i \in \{1, \dots, n\}\}.$$

As a direct consequence of their definitions, these composite preferences can be flattened.

**Proposition 6.** *Let $\succeq_1, \dots, \succeq_n$ be preference relations. For all $0 \leq k < m \leq n$, and for all $\rho \in \{and, lexico, pareto\}$, $\rho(\succeq_1, \dots, \succeq_k, \rho(\succeq_{k+1}, \dots, \succeq_m), \succeq_{m+1}, \dots, \succeq_n)$ is equal to $\rho(\succeq_1, \dots, \succeq_n)$.*

Moreover, lexico approximates the other two composite preferences.

**Theorem 4.** *Let $\succeq_1, \dots, \succeq_n$ be preference relations. Hence, $lexico(\succeq_1, \dots, \succeq_n)$ is an id-approximation of $and(\succeq_1, \dots, \succeq_n)$ and of $pareto(\succeq_1, \dots, \succeq_n)$.*

As anticipated in the previous section, lexico can also encode poset preferences.

**Theorem 5.** *Let $\succ$ be a strict partial order over a set $\mathcal{A}$ of atoms. Let $\mathcal{A}_i$ be $\{p \in \mathcal{A} \mid \exists_{=i}\, p' \succ p\}$. Hence, $lexico(\supseteq_{\mathcal{A}_0}, \dots, \supseteq_{\mathcal{A}_{|\mathcal{A}|-1}})$ is an id-reduction of $poset^{\succ}$.*

Finally, lexico can also encode less-weight preferences.

**Theorem 6.** *Given $\leq^w_{\mathcal{A}}$, let $n$ be $\left\lceil \log_2(1 + \sum_{p \in \mathcal{A}} w(p)) \right\rceil$, and $e^w_{\mathcal{A}}$ be $I \mapsto I \cup \left\{x_{i-1} \mid i \in \{1, \dots, n\},\ \sum_{p \in I \cap \mathcal{A}} w(p) \equiv 0 \mod 2^{i-1}\right\}$, where each $x_i$ is a fresh atom. Thus, $lexico(\subseteq_{\{x_{n-1}\}}, \dots, \subseteq_{\{x_0\}})$ is an $e^w_{\mathcal{A}}$-reduction of $\leq^w_{\mathcal{A}}$.*

## Normal Forms

This section introduces two normal forms for the preference relations introduced in the previous sections, and shows how to obtain them by applying approximations.

The *lexico-weight normal form* (LWNF) and the *lexico-subset normal form* (LSNF) are the following:

$$lexico(\leq^{w_1}_{\mathcal{A}_1}, \dots, \leq^{w_n}_{\mathcal{A}_n})$$
$$lexico(\subseteq_{\mathcal{A}_1}, \dots, \subseteq_{\mathcal{A}_n})$$

where $n \geq 1$, $\mathcal{A}_i$ is a set of atoms, and $w_i : \mathcal{A}_i \to \mathbb{Z}$, for all $i \in \{1, \dots, n\}$.

The LWNF normalization starts with $f := id$, and performs a depth-first search on the tree structure of the preference relation in input. If a node $N$ is not already in LWNF, then $N$ and possibly $f$ are replaced as follows:

1. if $N$ is $\geq^w_{\mathcal{A}}$, then $N := \leq^{-w}_{\mathcal{A}}$ (Proposition 2);
2. if $N$ is $\subseteq_{\mathcal{A}}$, then $N := \leq^{card}_{\mathcal{A}}$ (Proposition 3);
3. if $N$ is $\supseteq_{\mathcal{A}}$, then $N := \geq^{card}_{\mathcal{A}}$ (Proposition 3);
4. if $N$ is $aso^{\boldsymbol{\pi}}$, then $N := \supseteq_{\mathcal{A}^{\boldsymbol{\pi}}}$, and $f := e^{\boldsymbol{\pi}} \circ f$ (Theorem 3);
5. if $N$ is $poset^{\succ}$, then $N := lexico(\supseteq_{\mathcal{A}_0}, \dots, \supseteq_{\mathcal{A}_{|\mathcal{A}|-1}})$, where $\mathcal{A}_i$ is $\{p \in \mathcal{A} \mid \exists_{=i}\, p' \succ p\}$ (Theorem 5);
6. if $N$ is $neg(\leq^w_{\mathcal{A}})$, then $N := \geq^w_{\mathcal{A}}$ (Proposition 5);
7. if $N$ is $\rho(\leq^{w_1}_{\mathcal{A}_1}, \dots, \leq^{w_n}_{\mathcal{A}_n})$, with $\rho \in \{and, pareto\}$, then $N := lexico(\leq^{w_1}_{\mathcal{A}_1}, \dots, \leq^{w_n}_{\mathcal{A}_n})$ (Theorem 4);
8. if $N$ is the preference $lexico(\leq^{w_1}_{\mathcal{A}_1}, \dots, \leq^{w_k}_{\mathcal{A}_k}, lexico(\leq^{w_{k+1}}_{\mathcal{A}_{k+1}}, \dots, \leq^{w_m}_{\mathcal{A}_m}), \leq^{w_{m+1}}_{\mathcal{A}_{m+1}}, \dots, \leq^{w_n}_{\mathcal{A}_n})$, then $N := lexico(\leq^{w_1}_{\mathcal{A}_1}, \dots, \leq^{w_k}_{\mathcal{A}_n})$ (Proposition 6).

The LSNF normalization is similar, with $N$ and $f$ replaced as follows:

1. if $N$ is $\leq^w_{\mathcal{A}}$, then $N := lexico(\subseteq_{\{x_{n-1}\}}, \dots, \subseteq_{\{x_0\}})$, and $f := e^w_{\mathcal{A}} \circ f$, where $n = \left\lceil \log_2(1 + \sum_{p \in \mathcal{A}} w(p)) \right\rceil$ and each $x_i$ is a fresh atom (Theorem 6);
2. if $N$ is $\geq^w_{\mathcal{A}}$, then $N := \leq^{-w}_{\mathcal{A}}$ (Proposition 2);
3. if $N$ is $\supseteq_{\mathcal{A}}$, then $N := \subseteq_{\mathcal{A}^{neg}}$, and $f := e^{\mathcal{A}^{neg}}$ (Proposition 4);
4. if $N$ is $aso^{\boldsymbol{\pi}}$, then $N := \supseteq_{\mathcal{A}^{\boldsymbol{\pi}}}$, and $f := e^{\boldsymbol{\pi}} \circ f$ (Theorem 3);
5. if $N$ is $poset^{\succ}$, then $N := lexico(\supseteq_{\mathcal{A}_0}, \dots, \supseteq_{\mathcal{A}_{|\mathcal{A}|-1}})$, where $\mathcal{A}_i$ is $\{p \in \mathcal{A} \mid \exists_{=i}\, p' \succ p\}$ (Theorem 5);
6. if $N$ is $neg(\subseteq_{\mathcal{A}})$, then $N := \supseteq_{\mathcal{A}}$ (Proposition 5);
7. if $N$ is $\rho(\subseteq_{\mathcal{A}_1}, \dots, \subseteq_{\mathcal{A}_n})$, with $\rho \in \{and, pareto\}$, then $N := lexico(\subseteq_{\mathcal{A}_1}, \dots, \subseteq_{\mathcal{A}_n})$ (Theorem 4);
8. if $N$ is the preference $lexico(\subseteq_{\mathcal{A}_1}, \dots, \subseteq_{\mathcal{A}_k}, lexico(\subseteq_{\mathcal{A}_{k+1}}, \dots, \subseteq_{\mathcal{A}_m}), \subseteq_{\mathcal{A}_{m+1}}, \dots, \subseteq_{\mathcal{A}_n})$, then $N := lexico(\subseteq_{\mathcal{A}_1}, \dots, \subseteq_{\mathcal{A}_n})$ (Proposition 6).

**Theorem 7.** *Let $\succeq$ be a preference relation obtained by composing less-weight, more-weight, subset, superset, aso, poset, and, lexico, pareto. If $N$ and $f$ are the output of the LWNF (or LSNF) normalization of $\succeq$, then $N$ is an $f$-approximation of $\succeq$.*

## Answer Set Programming

In this section, we concretize the previous approach in terms of ASP. To this end, we start with some ASP background, and then show how knowledge bases can be represented by

ASP programs, and how the join of knowledge bases can be reduced to the union of ASP programs. As a result, ASP solvers can be used to implement the previous algorithms.

A *literal* is an atom or the failure symbol $\bot$, possibly preceded by (default) *negation* $\sim$. An ASP program $\Pi$ is a set of rules of the form $H \leftarrow B$, where $H$ and $B$ are respectively a disjunction and a conjunction of literals. Let $\mathit{atoms}(\Pi)$ denote the set of atoms occurring in $\Pi$. Let $\{p\}$ be a compact representation of $p \vee {\sim}p \leftarrow$, also called *choice rule*. For a set $\mathcal{A}$ of atoms, let $G^{\mathcal{A}}$ be the ASP program comprising a choice rule $\{p\}$ for each $p \in \mathcal{A}$.

An ASP program $\Pi$ is *normal* if each rule head in $\Pi$ is an atom, and is *nondisjunctive* if each rule head in $\Pi$ is an atom or empty. The *dependency graph* $\mathcal{G}_\Pi$ of $\Pi$ has nodes $\mathit{atoms}(\Pi)$, an arc $p \rightarrow^{+} q$ if there is a rule with $p$ in the head and $q$ in the body, and an arc $p \rightarrow^{-} q$ if there is a rule with $p$ in the head and literal ${\sim}q$ in the head or in the body. $\Pi$ is *stratified* if there is no cycle in $\mathcal{G}_\Pi$ involving a negative arc ($\rightarrow^{-}$).

An interpretation $I$ is a set of atoms. Relation $\models$ is defined as follows: $I \not\models \bot$; for an atom $p$, $I \models p$ if $p \in I$; $I \models {\sim}p$ if $I \not\models p$; $I \models H \leftarrow B$ if $I \not\models \ell$ for some $\ell \in B$, or $I \models \ell$ for some $\ell \in H$; $I \models \Pi$ if $I \models r$ for all $r \in \Pi$. Let $\Pi^I$ be the *program reduct* obtained from $\Pi$ by replacing each ${\sim}p$ with $\bot$ if $p \in I$, and with ${\sim}\bot$ if $p \notin I$. $I$ is a *stable model* of $\Pi$ if $I \models \Pi$, and there is no $J \subset I$ such that $J \models \Pi^I$. Let $\mathit{SM}(\Pi)$ denote the set of stable models of $\Pi$.

A *weak constraint* is of the form $[w@l] \leftarrow\!\!\sim p$, where $p$ is an atom, and $w$ and $l$ are integers called *weight* and *level*. The preference relation $\succeq_W$ associated with a set $W$ of weak constraints whose levels are $l_1 > \cdots > l_n$ (for some $n \geq 0$) is $\mathit{lexico}(\leq_{\mathcal{A}_1}^{w_1}, \ldots, \leq_{\mathcal{A}_n}^{w_n})$, where $\mathcal{A}_i$ is $\{p \mid [w@l_i] \leftarrow\!\!\sim p \in W\}$, and $w_i : p \mapsto \sum_{[w@l_i] \leftarrow\!\sim p \in W} w$.

In order to apply the algorithms presented in the previous sections, some associations between ASP programs and knowledge bases are required. First of all, an ASP program $\Pi$ defines the knowledge base $(\mathit{atoms}(\Pi), \mathit{SM}(\Pi))$. If clear from the context, $\Pi$ will be used to denote the associated knowledge base.

Preference knowledge bases are defined by *preference programs*. Let $\succeq$ be a preference relation over $\mathcal{A}$, and $\Pi_\succ$ be a program over $\mathcal{A}'$ such that $\mathcal{A} \cup \mathcal{A}^? \subseteq \mathcal{A}'$ and each $p \in \mathcal{A} \cup \mathcal{A}^?$ only occurs in rule bodies of $\Pi_\succ$. Then, $\Pi_\succ$ is a preference program for $\succeq$ if the projection of $G^{\mathcal{A} \cup \mathcal{A}^?} \cup \Pi_\succ$ on $\mathcal{A} \cup \mathcal{A}^?$ is $[\succ]$. *Complementary preference programs* $\Pi_{\not\succ}$ are defined analogously, replacing $\Pi_\succ$ and $[\succ]$ by $\Pi_{\not\succ}$ and $[\not\succ]$, respectively. If $\succeq$ is a preference relation over $\mathcal{A}$ such that deciding whether $I \succ J$ for $I, J \subseteq \mathcal{A}$ can be done in polynomial time, then there is always some nondisjunctive and stratified preference program $\Pi_\succ$ for $\succeq$, and $\Pi_{\not\succ}$ can be automatically constructed from $\Pi_\succ$; see (Brewka et al. 2015b) for details.

**Example 7.** Below is a preference program for $[\subset_{\{a,b,c,d\}}]$:

$$\begin{array}{lll}
\bot \leftarrow a, {\sim}a^? & \mathit{strict} \leftarrow {\sim}a, a^? & \bot \leftarrow {\sim}\mathit{strict}\\
\bot \leftarrow b, {\sim}b^? & \mathit{strict} \leftarrow {\sim}b, b^? & \\
\bot \leftarrow c, {\sim}c^? & \mathit{strict} \leftarrow {\sim}c, c^? & \\
\bot \leftarrow d, {\sim}d^? & \mathit{strict} \leftarrow {\sim}d, d^? &
\end{array}$$

For example, $\{a, b\} \cup \{a, b, c\}^? \cup \{\mathit{strict}\}$ is a model of the program above (extended with $G^{\{a,b,c,d\} \cup \{a,b,c,d\}^?}$), while neither $\{a, b\} \cup \{b, c\}^?$ nor $\{a, b\} \cup \{b, c\}^? \cup \{\mathit{strict}\}$ are. The complementary preference program is obtained by replacing each $x \in \mathcal{A}$ with $x^? \in \mathcal{A}^?$ and vice versa, and each $\bot$ in rule heads with a fixed atom $\mathit{ok}$ that is forced to be true:

$$\begin{array}{lll}
\mathit{ok} \leftarrow a^?, {\sim}a & \mathit{strict} \leftarrow {\sim}a^?, a & \mathit{ok} \leftarrow {\sim}\mathit{strict}\\
\mathit{ok} \leftarrow b^?, {\sim}b & \mathit{strict} \leftarrow {\sim}b^?, b & \bot \leftarrow {\sim}\mathit{ok}\\
\mathit{ok} \leftarrow c^?, {\sim}c & \mathit{strict} \leftarrow {\sim}c^?, c & \\
\mathit{ok} \leftarrow d^?, {\sim}d & \mathit{strict} \leftarrow {\sim}d^?, d &
\end{array}$$

For example, $\{a, b\}^? \cup \{b, c\} \cup \{\mathit{strict}, \mathit{ok}\}$ is a model of the program above (extended with $G^{\{a,b,c,d\} \cup \{a,b,c,d\}^?}$), while $\{a, b\}^? \cup \{a, b, c\} \cup X$ (for all $X \subseteq \{\mathit{strict}, \mathit{ok}\}$) is not. ■

The knowledge bases $(\mathcal{A}, \{I\})$ and $(\mathcal{A}, 2^{\mathcal{A}} \setminus \{I\})$, for a set of atoms $\mathcal{A}$ and $I \subseteq \mathcal{A}$, can be respectively represented by the ASP programs $\Pi_{\mathcal{A},I}$ and $G^{\mathcal{A}} \cup \Pi_{\mathcal{A},\overline{I}}$, where $\Pi_{\mathcal{A},I}$ is $\{p \leftarrow\mid p \in I\} \cup \{\bot \leftarrow p \mid p \in \mathcal{A} \setminus I\}$ and $\Pi_{\mathcal{A},\overline{I}}$ is $\{\bot \leftarrow \bigwedge_{p \in I} p \wedge \bigwedge_{p \in \mathcal{A} \setminus I} {\sim}p\}$. We drop $\mathcal{A}$ from $\Pi_{\mathcal{A},I}$ and $\Pi_{\mathcal{A},\overline{I}}$ whenever clear from context. Finally, for $n \geq 1$, let $\Pi^n$ denote the ASP program obtained from $\Pi$ by replacing all occurrences of atoms of the form $p^?$ with $p^n$.

Under some conditions, joins of knowledge bases can be obtained by unions of ASP programs, as formalized below.

**Theorem 8.** *Let $\Pi, \Pi'$ be ASP programs such that each $p \in \mathit{atoms}(\Pi)$ may only occur in rule bodies of $\Pi'$. The knowledge base $\Pi \bowtie (\Pi' \cup G^{\mathit{atoms}(\Pi) \cap \mathit{atoms}(\Pi')})$ is equivalent to the knowledge base $\Pi \cup \Pi'$.*

Thanks to Theorem 8 we can use unions of ASP programs to represent the joins of knowledge bases of the algorithms from the previous sections. Indeed, the following claim is analogous to Proposition 1.

**Proposition 7.** *Let $\Gamma = (\mathcal{A}, \mathcal{M})$ be a knowledge base defined by program $\Pi$, $\succeq$ be a preference relation over $\mathcal{A}$, and $\Pi_\succ$ be a preference program for $\succeq$. $I \in \mathcal{M}$ is a $\succeq$-optimal model of $\Gamma$ if and only if $\Pi \cup \Pi_\succ \cup \Pi_I^?$ is inconsistent.*

Hence, the join of Algorithm 1 is represented by $\Pi \cup \Pi_\succ \cup \Pi_I^?$. Similarly, the join of Algorithm 2 is represented by $\Pi \cup \Pi_{\overline{I}} \cup \Pi_{\not\succ}^n \cup \Pi_I^n$, and the one of Algorithms 3 and 4 by $\Pi \cup \Pi_{\overline{I \cap \mathcal{A}}} \cup \Pi_{\not\succ}^n \cup \Pi_{I \cap \mathcal{A}}^n$. Therefore, Algorithms 1 and 2 can be implemented using an ASP system for the procedure *solve*. Essentially, this corresponds to the solving algorithm of ASPRIN presented in (Brewka et al. 2015b).

To implement Algorithms 3 and 4, instead, we also need to represent expansion functions by *expansion programs*. Let $e : 2^{\mathcal{A}} \rightarrow 2^{\mathcal{A}'}$ be an expansion function, and $\Pi_e$ be a program over $\mathcal{A}''$ such that $\mathcal{A}' \subseteq \mathcal{A}''$ and each $p \in \mathcal{A}$ only occurs in rule bodies of $\Pi_e$. Then, $\Pi_e$ is an expansion program for $e$ if for all $I \subseteq \mathcal{A}$, the program $\Pi_e \cup \{p \leftarrow\mid p \in I\}$ has a unique stable model $J$ such that $J \cap \mathcal{A}' = e(I)$. If $e$ is computable in polynomial time, then there is always some normal and stratified expansion program $\Pi_e$ for $e$.

**Example 8.** Let $\pi$ be the aso statement $x_1 > x_2 > x_3 : x$.

Below is an expansion program $\Pi_{e^{\{\pi\}}}$ for $e^{\{\pi\}}$:

$$\begin{array}{lll} x_1^\pi \leftarrow {\sim}x & x_1^\pi \leftarrow x_1 & x_1^\pi \leftarrow {\sim}x_2, {\sim}x_3 \\ x_2^\pi \leftarrow x_1^\pi & x_2^\pi \leftarrow x_2 & \\ x_3^\pi \leftarrow x_2^\pi & x_3^\pi \leftarrow x_3 & \end{array}$$

For $I \subseteq \{x_1, x_2, x_3\}$, or $I \subseteq \{x_1, x\}$, $\Pi_{e^{\{\pi\}}} \cup \{p \leftarrow \mid p \in I\}$ has unique stable model $I \cup \{x_1^\pi, x_2^\pi, x_3^\pi\} = e(I)$. Similarly, for $I$ being $\{x, x_2\}$ or $\{x, x_2, x_3\}$, $\Pi_{e^{\{\pi\}}} \cup \{p \leftarrow \mid p \in I\}$ has unique stable model $I \cup \{x_2^\pi, x_3^\pi\} = e(I)$. Finally, for $I$ being $\{x, x_3\}$, $\Pi_{e^{\{\pi\}}} \cup \{p \leftarrow \mid p \in I\}$ has unique stable model $I \cup \{x_3^\pi\} = e(I)$. ■

Expansion programs can be used to represent knowledge bases of the form $e(\Gamma)$, as formalized next.

**Proposition 8.** *Let $\Gamma = (\mathcal{A}, \mathcal{M})$ be a knowledge base defined by program $\Pi$, $e : 2^{\mathcal{A}} \to 2^{\mathcal{A}'}$ be an expansion function, and $\Pi_e$ be an expansion program for $e$. Then, the projection of $\Pi \cup \Pi_e$ on $\mathcal{A}'$ is $e(\Gamma)$.*

The following claim is analogous to Theorem 2.

**Theorem 9.** *Let $\Gamma = (\mathcal{A}, \mathcal{M})$ be a knowledge base defined by program $\Pi$, $\succeq$ be a preference relation over $\mathcal{A}$, $e : 2^{\mathcal{A}} \to 2^{\mathcal{A}'}$ be an expansion function represented by program $\Pi_e$, and $\succeq'$ be an $e$-approximation of $\succeq$. For any $I \subseteq \mathcal{A}$, if $I \subseteq \mathcal{A}'$ is a $\succeq'$-optimal model of the projection of $\Pi \cup \Pi_e$ on $\mathcal{A}'$, then $I \cap \mathcal{A}$ is a $\succeq$-optimal model of $\Gamma$.*

Therefore, Algorithms 3 and 4 can be implemented in ASP replacing $e(\Gamma)$ by $\Pi \cup \Pi_e$, and using Algorithms 1 and 2 for procedures *solveOpt* and *enumOpt*, provided that programs $\Pi_{\not\succ}$, $\Pi_{\succ'}$ and $\Pi_{\not\succ'}$ are available.

Alternatively, procedures *solveOpt* and *enumOpt* can be implemented calling directly an ASP solver, whenever a native method for computing $\succeq'$-optimal stable models is available. In CLASP, this is the case for preference relations represented in the normal forms LWNF and LSNF. In the first case, if $\succeq'$ has the form $\mathit{lexico}(\leq^{w_1}_{\mathcal{A}_1}, \ldots, \leq^{w_n}_{\mathcal{A}_n})$, $\succeq'$-optimal models correspond to $\succeq_W$-optimal models, where $W$ is the set of weak constraints $\{[w_i(p)@n - i + 1] \leftsquigarrow p \mid i \in \{1, \ldots, n\}, p \in \mathcal{A}_i\}$. In the second case, if $\succeq'$ has the form $\mathit{lexico}(\subseteq_{\mathcal{A}_1}, \ldots, \subseteq_{\mathcal{A}_n})$, $\succeq'$-optimal models can be computed by CLASP's *heuristic* directives: $\{$`#heuristic p. [n-i+1,false]`$\mid$ $i \in \{1, \ldots, n\}$, $p \in \mathcal{A}_i\}$; see (Gebser et al. 2015) for details. In fact, we can apply this approach to any preference relation obtained by composing relations from above, given that we have shown how to approximate them by preference relations in normal form (Theorem 7).

## Implementation

The third version of ASPRIN (Brewka et al. 2015b; 2015a) (`https://potassco.org/asprin`) implements approximations by calling an ASP solver for procedures *solveOpt* and *enumOpt*. Option `--approximation=weak` activates the method using weak constraints, while `--approximation=heuristic` activates the method using heuristic directives. Here we illustrate the `weak` mode by means of an example. (The description of the `heuristic` mode is analogous; weights are unary encoded in the current implementation of the `heuristic` mode, and a one-to-one implementation of Theorem 6 is left as future work.)

The following lines are part of an ASPRIN program that defines predicates `a/1`, `b/1` and `dom/1`:

```
#preference(p1,subset) { a(X) : dom(X) }.
#preference(p2,subset) { b(X) : dom(X) }.
#preference(p3,pareto) {  **p1;   **p2 }.
#optimize(p3).
```

The first line defines preference statement `p1` of type `subset`, declaring the subset preference relation over atoms of predicate `a`. Preference statement `p2` declares a subset preference relation over `b`. Statement `p3` combines `p1` and `p2` by `pareto`. Finally, the optimize statement of the last line instructs ASPRIN to compute `p3`-optimal models.

Preference types (`subset`, `pareto`, etc.) are encoded in ASPRIN by ASP programs. Many preference types (including all ones studied here) are already encoded in ASPRIN's library, which can be extended by simply adding other ASP programs. These encodings rely on the reification of the preference statements generated by the system. For our example, the reification produces the following rules (where for simplicity we replaced some terms by symbol _):

```
p(p1,subset).  p(p2,subset).  p(p3,pareto).
p(p1,_,_,for(a(X)),_) :- dom(X).
p(p2,_,_,for(b(X)),_) :- dom(X).
p(p3,_,_,name(p1),_). p(p3,_,_,name(p2),_).
optimize(p3).
```

Moreover, the atoms appearing in the preference statements are made accessible via predicate `holds/1`:

```
holds(a(X)) :- a(X), dom(X).
holds(b(X)) :- b(X), dom(X).
```

For the `weak` mode, ASPRIN requires the implementation of so-called weak approximation programs. They contain three elements: (1) the expansion program of the approximation (empty in our example); (2) the definition of the corresponding preference relation in LWNF using predicate `wc(P,W,L,X)` (which means that for preference `P`, atom `X` has weight `W` at level `L`); and (3) a mapping of `wc/4` into weak constraints. For `subset` and `pareto` we have the following rules:

```
wc(P,1,1,X):- p(P,subset),p(P,_,_,for(X),_).
wc(P,W,L2,X):-p(P,pareto),
              p(P,_,_,name(P1),_),
              wc(P1,W,L1,X),map(P,P1,L1,L2).
:~ wc(P,W,L,X),optimize(P),holds(X). [W@L,X]
```

The first rule gives weight `1` at level `1` to atoms `X` appearing in a subset preference; this corresponds to item 2 of the LWNF normalization, using Proposition 3. The second rule maps `pareto` to `lexico`, which is flattened at the same time; this corresponds to items 7–8 of the LWNF normalization, using Theorem 4 and Proposition 6. Note that the rule uses atom `map(P,P1,L1,L2)` to represent that level `L1` of preference `P1` is mapped to level `L2` in `P`; the definition of `map/4` is omitted for simplicity. Finally, the last rule activates the weak constraints associated with the preference to be optimized, using predicate `holds/1` to refer to the original atoms. The encoding can be directly used in ASPRIN to compute one optimal model, and together with a preference program can be used to enumerate many models.

## Experiments

We now evaluate our approach by contrasting its implementation with the one of ASPRIN. To this end, we consider the following variants: basic ASPRIN (B), ASPRIN in heuristic approximation mode (H), ASPRIN in weak approximation mode with CLASP's iterative algorithm (I), and ASPRIN in weak approximation mode with CLASP's unsat core algorithm (U). These implementations are evaluated in view of computing a single and all optimal models.

The dedicated solving techniques used in H, I, and U have already proved their merits in various settings. This is mainly due to their highly optimized implementation in existing ASP solvers. Hence, we expect that their use leads to an improvement when computing a single model. As for computing several models, heuristic mode must use Algorithm 3 for enumeration (because CLASP does not support Algorithm 4 in this mode), while for weak mode we opt for using Algorithm 4. For the latter we expect a general improvement from using CLASP's inner enumeration implementation. However, we are solving different problems when approximating, and the approximated problems may behave differently than the original ones. For instance, it is usually easier to compute a subset minimal model than a cardinality minimal one. On the other hand, it is not clear whether this difference persists when we enumerate all optimal models, since in this case we have to go through the whole search space. For instance, a larger part of the search space can be pruned after computing the first cardinality optimal model than after computing the first subset minimal one. This could lead to reversed roles in enumeration.

All experiments ran on an Intel Xeon 2.20GHz processor under Linux. Each run was limited to 1 hour runtime and 8GB of memory. Each instance was ran twice, once to compute one optimal model, and another time to compute all optimal models. We report the aggregated results per benchmark class: number of timeouts (in parentheses, if applicable), average runtime in seconds for computing one model, and velocity for computing all models (i.e., number of computed optimal models divided by runtime in hours).

In our first two experimental series, we use the benchmark set from (Brewka et al. 2015b) comprising 193 benchmark instances from eight different classes: **15-Puzzle**, **Crossing**, and **Valves** stem from the ASP competitions of 2009 and 2013; **Ricochet** Robots from (Gebser et al. 2013a), for which subset and cardinality minimal models coincide, Circuit **Diagnosis** from (Siddiqi 2011) and adapted to ASP in (Gebser et al. 2013b), Metabolic Network **Expansion** from (Schaub and Thiele 2009), Transcription Network **Repair** from (Gebser et al. 2010), and **Timetabling** from (Banbara et al. 2013). All classes involve a single optimization statement; **Valves** and **Timetabling** deal with weight summation, all others with cardinality. In what follows, we only use the more general term 'weight' (and drop 'cardinality').

We report in Table 1 results obtained on weight optimizing benchmarks (marked by $\Sigma$) and their $\subseteq$-minimizing counterparts. The latter are obtained from the above benchmarks by replacing weight by subset optimization. Note that $\subseteq$-minimal models form a superset of cardinality minimal ones. Considering the first series ($\Sigma$), we observe that unsat-core-based approximation (U) performs best by far when computing a single optimum — except for **Valves**. Here, iterative approximation works better, slightly outperforming plain ASPRIN. The two latter exhibit a similar behavior on the other classes. In this setting, the heuristic variant of ASPRIN (H) performs worst, although it even gets close to U on **Ricochet** and **Crossing**. The overall trend changes when computing all optimal models and the roles of I and U are reversed. This is insofar surprising since in both cases the ASP solver is used to enumerate all models sharing the objective value of the initially found optimal model. We have no clear explanation but conjecture that this is due to learning effects. Generally, I and U have a higher velocity and thus compute more models than basic ASPRIN, and in turn its heuristic extension H. In addition, we also ran CLASP using its builtin iterative and unsat-core-based optimization, and its performance was similar to ASPRIN using I and U approximation, respectively. This shows that the latter generate no computational overhead. Also, our approximate algorithms allow us to overcome a previous performance gap between CLASP and ASPRIN when dealing with weight optimization.

When switching to $\subseteq$-optimization, U keeps its pole position but H is on a par. As above, **Valves** and now also **Crossing** are exceptions where ASPRIN is better. The iterative approximation performs well but is generally slower. When enumerating $\subseteq$-optimal models, unsat-core-based approximation U has a clear edge over all others, especially on classes with many optimal models. For instance, all methods time out on **Expansion**, **Repair** and **Diagnosis** due to overwhelmingly many optimal models. However, many more models are computed by both weak approximations U and I since they take advantage of CLASP's inner functionality.

Table 2 summarizes our results with aggregated preference. The first series ($\ell$) deals with lexicographically ordered weight preferences, and the second ($\wp$) with the combination of subset preferences according to the Pareto principle. In both cases, the corresponding benchmark instances are obtained from the aforementioned sets by dividing their optimization statements in 16 parts, as detailed in (Brewka et al. 2015b). Again, unsat-core-based approximation U has a clear edge in the first series $\ell$. When computing one optimal model, exceptions occur on **Ricochet** and **Valves** where H and I perform best. The basic ASPRIN setting performs worst overall. As above, this is to be expected since U and I exploit CLASP's implementation, which includes lexicographic aggregation. These observations carry over to the enumeration of optimal models, where U is best, followed by I, H, and B. Mixed results are obtained when computing one Pareto-optimal model ($\wp$). Here, the heuristic approach H performs quite well — except on **Valves**. When computing all Pareto-optimal models, we observe the familiar pattern: the field is dominated by the approximation techniques U and I, followed by H and B.

Next, we evaluate our approach on *poset* benchmarks. For this, we draw upon a corresponding benchmark set in (Brewka et al. 2015a), which is itself a selection of random and structured SAT instances provided by (Di Rosa, Giunchiglia, and Maratea 2010) and translated to ASP in the straightforward way (cf. (Niemelä 1999)). Interestingly,

| | B,$\Sigma$ | H,$\Sigma$ | I,$\Sigma$ | U,$\Sigma$ | B,$\subseteq$ | H,$\subseteq$ | I,$\subseteq$ | U,$\subseteq$ |
|---|---|---|---|---|---|---|---|---|
| **Ricochet** | 207, 34 | 35, 267 | 217, 24 | 32, 279 | 156, 38 | 32, 262 | 232, 24 | 32, 274 |
| **Timetabling** | (3) 1226, 164 | (11) 3300, 63 | (1) 318, 2128 | 2, 3158 | 92, 207 | 2, 2226 | 32, 1251 | 3, 2189 |
| **15-puzzle** | 35, 144 | 395, 16 | 36, 130 | 16, 143 | 10, 4365 | 27, 2480 | 37, 11158 | 16, 13018 |
| **Crossing** | 63, 931 | 11, 352 | 41, 1963 | 5, 698 | 0, 729 | 2, 13349 | 43, 16553 | 5, 40776 |
| **Valves** | 39, 549 | (10) 1900, 79 | 37, 585 | (20) 2725, 117 | 17, 895 | (18) 2312, 246 | 60, 1051 | 240, 451 |
| **Expansion** | 94, 1893 | (29) 3485, 3 | 40, 6492 | 3, 14273 | 24, 105 | 3, 7582 | 41, 4697 | 3, 8070 |
| **Repair** | 74, 1083 | (30) 3600, 0 | 25, 30239 | 1, 11810 | 5, 670 | 1, 4183 | 26, 17764 | 1, 47002 |
| **Diagnosis** | 192, 410 | 37, 485 | 46, 48789 | 0, 30370 | 16, 105 | 0, 14232 | 46, 79800 | 1, 429557 |
| **Total** | (3) 241, 651 | (80) 1595, 158 | (1) 95, 11294 | (20) 348, 7606 | 40, 889 | (18) 297, 5570 | 65, 16537 | 38, 67667 |

Table 1: Experimental results on weight ($\Sigma$) and subset ($\subseteq$) optimization

| | B,$\ell$ | H,$\ell$ | I,$\ell$ | U,$\ell$ | B,$\wp$ | H,$\wp$ | I,$\wp$ | U,$\wp$ |
|---|---|---|---|---|---|---|---|---|
| **Ricochet** | 188, 41 | 37, 234 | 223, 24 | 175, 72 | 248, 30 | 36, 256 | 229, 24 | 200, 69 |
| **Timetabling** | (10) 3007, 31 | (11) 3300, 209 | 29, 454 | 8, 899 | 65, 133 | 7, 1086 | 24, 664 | 8, 1170 |
| **15-puzzle** | 24, 176 | (7) 3600, 0 | 23, 210 | 47, 241 | 10, 3472 | 319, 263 | 23, 1305 | 50, 383 |
| **Crossing** | 5, 801 | 13, 979 | 1, 2886 | 0, 4763 | 1, 752 | 4, 4211 | 1, 1714 | 0, 2929 |
| **Valves** | 22, 615 | (15) 1975, 224 | 35, 570 | 137, 655 | 21, 984 | (16) 2089, 241 | 25, 968 | 98, 784 |
| **Expansion** | (6) 1702, 0 | 24, 171 | 81, 129 | 3, 1303 | 18, 184 | 3, 796 | 82, 2167 | 3, 3577 |
| **Repair** | (21) 2529, 0 | 20, 208 | 2, 2563 | 2, 3419 | 3, 610 | 1, 1854 | 2, 2271 | 2, 2733 |
| **Diagnosis** | (30) 3600, 0 | 1, 6408 | 42, 9269 | 0, 50068 | 2, 846 | 0, 6464 | 42, 10673 | 0, 10673 |
| **Total** | (67) 1385, 208 | (33) 1121, 1054 | 54, 2013 | 46, 7678 | 46, 876 | (16) 308, 1896 | 54, 2473 | 45, 2790 |

Table 2: Experimental results on lexicographic ($\ell$) and Pareto ($\wp$) optimization

| | B | H | I | U |
|---|---|---|---|---|
| **d0** | 1, 753 | 0, 3410 | (100) 3600, 0 | (100) 3600, 0 |
| **d0.00621** | 1, 620 | 1, 1004 | (100) 3600, 0 | (100) 3600, 0 |
| **d0.04972** | 19, 56 | (11) 782, 289 | (100) 3600, 0 | (100) 3600, 0 |
| **d0.02486** | 7, 156 | (8) 509, 787 | (100) 3600, 0 | (100) 3600, 0 |
| **d0.01243** | 2, 424 | 29, 378 | (100) 3600, 0 | (100) 3600, 0 |
| **d1** | 79, 40 | (13) 784, 26 | (100) 3600, 0 | (100) 3600, 0 |
| **Total** | 18, 342 | (32) 351, 982 | (600) 3600, 0 | (600) 3600, 0 |
| **maxsat** | 143, 138 | 111, 2319 | (3) 384, 18846 | (1) 155, 14542 |
| **pbo-mqc_nencdr** | 9, 2444 | (8) 481, 790 | 80, 811 | (8) 512, 113 |
| **pbo-mqc_nlogencdr** | 5, 3337 | 96, 1079 | 23, 1903 | 74, 346 |
| **primes** | (16) 407, 5429 | (17) 415, 6875 | (52) 1274, 28398 | (39) 957, 49685 |
| **routing** | 3, 9630 | 12, 9389 | 306, 32549 | 1, 32501 |
| **minone** | 53, 2637 | 56, 2693 | (1) 267, 11169 | 60, 20034 |
| **Total** | (16) 103, 3936 | (25) 195, 3857 | (56) 389, 15613 | (48) 293, 19537 |

Table 3: Experimental results on *poset* optimization

this benchmark set was used in (Brewka et al. 2015a) to demonstrate that ASPRIN is at eye level with the dedicated implementation of *poset* described in (Di Rosa, Giunchiglia, and Maratea 2010). Surprisingly, basic ASPRIN is indeed the most effective approach for solving the random instances, listed in the first half of Table 2, followed by its heuristic-driven variant H. Our formerly impressing approximation techniques I and U fail to solve a single instance in time; tuning CLASP parameters gave no improvement either. This illustrates nicely how our approximation may result in much harder problems than the original one. The picture changes slightly when dealing with structured problems, reflected by the second half of Table 2. Although the basic approach is still superior, and still followed by its heuristic variant, our weak approximation techniques are not completely lost, and they manage to enumerate more models.

We refrain from giving detailed results on our evaluation with *aso* preferences, since the available benchmarks are randomly generated (Zhu and Truszczyński 2013) and we thus obtain similar results as with *poset* on random instances: Our approximations I and U fail to solve a single instance in time. The basic approach is best for computing one optimal model, and H is better when computing many.

## Conclusion

Several qualitative and quantitative preference relations are linked by the notion of approximation given in this paper. Such links are not only interesting from a theoretical point of view, but actually put in practice by our implementation in ASPRIN. Interestingly, preference relations are encoded by ASP facts, and each approximation is a module comprising ASP rules for encoding the expansion function and the target preference relation. The system is therefore open to the addition of new approximations.

The two normal forms considered in this paper are motivated by the fact that weak constraints and heuristic directives, native constructs of CLASP, actually define lexicographic compositions of less-weight and subset preference relations. Hence, thanks to our normalization procedures, ASPRIN can now take advantage of several highly optimized algorithms implemented by CLASP for computing optimal solutions. Specifically, we tested the following strategies of CLASP: domain heuristic, iterative, and unsatisfiable core analysis. Our experiments register a performance improvement of ASPRIN, more evident in the enumeration of optimal models.

## Acknowledgments

Mario Alviano was partially supported by the POR CALABRIA FESR 2014-2020 project "DLV Large Scale" (CUP J28C17000220006), by the EU H2020 PON I&C 2014-2020 project "S2BDW" (CUP B28I17000250008), and by GNCS-INdAM. Javier Romero and Torsten Schaub were partially supported by DFG SCHA 550/9.

## References

Alviano, M. 2017. Model enumeration in propositional circumscription via unsatisfiable core analysis. *Theory and Practice of Logic Programming* 17(5-6):708–725.

Alviano, M. 2018. Query answering in propositional circumscription. In Lang, J., ed., *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, 1669–1675. ijcai.org.

Banbara, M.; Soh, T.; Tamura, N.; Inoue, K.; and Schaub, T. 2013. Answer set programming as a modeling language for course timetabling. *Theory and Practice of Logic Programming* 13(4-5):783–798.

Brewka, G.; Delgrande, J. P.; Romero, J.; and Schaub, T. 2015a. Implementing preferences with ASPRIN. In Calimeri, F.; Ianni, G.; and Truszczynski, M., eds., *Logic Programming and Nonmonotonic Reasoning - 13th International Conference, LPNMR 2015, Lexington, KY, USA, September 27-30, 2015. Proceedings*, volume 9345 of *Lecture Notes in Computer Science*, 158–172. Springer.

Brewka, G.; Delgrande, J. P.; Romero, J.; and Schaub, T. 2015b. ASPRIN: Customizing answer set preferences without a headache. In Bonet, B., and Koenig, S., eds., *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, 1467–1474. AAAI Press.

Brewka, G.; Niemelä, I.; and Truszczynski, M. 2003. Answer set optimization. In Gottlob, G., and Walsh, T., eds., *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*, 867–872. Morgan Kaufmann.

Buccafurri, F.; Leone, N.; and Rullo, P. 2000. Enhancing Disjunctive Datalog by Constraints. *IEEE Trans. Knowl. Data Eng.* 12(5):845–860.

Cabalar, P., and Son, T., eds. 2013. *Proceedings of the Twelfth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'13)*, volume 8148 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag.

Di Rosa, E.; Giunchiglia, E.; and Maratea, M. 2010. Solving satisfiability problems with preferences. *Constraints* 15(4):485–515.

Gebser, M.; Guziolowski, C.; Ivanchev, M.; Schaub, T.; Siegel, A.; Thiele, S.; and Veber, P. 2010. Repair and prediction (under inconsistency) in large biological networks with answer set programming. In Lin, F., and Sattler, U., eds., *Proceedings of the Twelfth International Conference on Principles of Knowledge Representation and Reasoning (KR'10)*, 497–507. AAAI Press.

Gebser, M.; Jost, H.; Kaminski, R.; Obermeier, P.; Sabuncu, O.; Schaub, T.; and Schneider, M. 2013a. Ricochet robots: A transverse ASP benchmark. In Cabalar and Son (2013), 348–360.

Gebser, M.; Kaufmann, B.; Otero, R.; Romero, J.; Schaub, T.; and Wanko, P. 2013b. Domain-specific heuristics in answer set programming. In desJardins, M., and Littman, M., eds., *Proceedings of the Twenty-Seventh National Conference on Artificial Intelligence (AAAI'13)*, 350–356. AAAI Press.

Gebser, M.; Kaufmann, B.; Romero, J.; Otero, R.; Schaub, T.; and Wanko, P. 2013c. Domain-specific heuristics in answer set programming. In desJardins, M., and Littman, M. L., eds., *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*. AAAI Press.

Gebser, M.; Kaminski, R.; Kaufmann, B.; Romero, J.; and Schaub, T. 2015. Progress in clasp Series 3. In Calimeri, F.; Ianni, G.; and Truszczynski, M., eds., *LPNMR 2015*, volume 9345 of *Lecture Notes in Computer Science*, 368–383. Springer.

Gelfond, M., and Lifschitz, V. 1991. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Comput.* 9(3/4):365–386.

Lifschitz, V. 1986. On the satisfiability of circumscription. *Artif. Intell.* 28(1):17–27.

McCarthy, J. 1980. Circumscription - A form of nonmonotonic reasoning. *Artif. Intell.* 13(1-2):27–39.

Niemelä, I. 1999. Logic Programs with Stable Model Semantics as a Constraint Programming Paradigm. *Ann. Math. Artif. Intell.* 25(3-4):241–273.

Rosa, E. D., and Giunchiglia, E. 2013. Combining approaches for solving satisfiability problems with qualitative preferences. *AI Commun.* 26(4):395–408.

Schaub, T., and Thiele, S. 2009. Metabolic network expansion with ASP. In Hill, P., and Warren, D., eds., *Proceedings of the Twenty-fifth International Conference on Logic Programming (ICLP'09)*, volume 5649 of *Lecture Notes in Computer Science*, 312–326. Springer-Verlag.

Siddiqi, S. 2011. Computing minimum-cardinality diagnoses by model relaxation. In Walsh, T., ed., *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI'11)*, 1087–1092. IJCAI/AAAI Press.

Simons, P.; Niemelä, I.; and Soininen, T. 2002. Extending and implementing the stable model semantics. *Artif. Intell.* 138(1-2):181–234.

van Harmelen, F.; Lifschitz, V.; and Porter, B. W., eds. 2008. *Handbook of Knowledge Representation*, volume 3 of *Foundations of Artificial Intelligence*. Elsevier.

Zhu, Y., and Truszczyński, M. 2013. On optimal solutions of answer set optimization problems. In Cabalar and Son (2013), 556–568.