Monotonic and Nonmonotonic Preference Revision *

Jan Chomicki **Joyce Song**
University at Buffalo
{chomicki,jsong5}@cse.buffalo.edu

Abstract

We study here preference revision, considering both the *monotonic* case where the original preferences are preserved and the *nonmonotonic* case where the new preferences may override the original ones. We use a relational framework in which preferences are represented using binary relations (not necessarily finite). We identify several classes of revisions that preserve order axioms, for example the axioms of strict partial or weak orders. We consider applications of our results to preference querying in relational databases.

1 Introduction

The notion of *preference* is common in various contexts involving decision or choice. Classical utility theory [Fishburn, 1970] views preferences as *binary relations*. A similar view has recently been espoused in database research [Chomicki, 2003; Kießling, 2002; Kießling & Köstler, 2002], where preference relations are used in formulating *preference queries*. In AI, various approaches to compact specification of preferences have been explored [Boutilier *et al.*, 2004]. The semantics underlying such approaches typically relies on preference relations between worlds.

However, user preferences are rarely static [Pu, Faltings, & Torrens, 2003]. A database user may be disappointed by the result of a preference query and decide to revise the preferences in the query. In fact, a user may start with a partial or vague concept of her preferences, and subsequently refine that concept. An agent may learn more about its task domain and consequently revise its preferences. Thus, it is natural to study *preference revision*, as we do in the present paper.

In our formulation, preference revision shares some of the principles, namely minimal change and primacy of new information, with classical belief revision [Gärdenfors & Rott, 1995]. However, its basic setting is different. In belief revision, propositional theories are revised with propositional formulas, yielding new theories. In preference revision, binary preference relations are revised with other preference relations, yielding new preference relations. Preference relations are single, finitely representable (though possibly infinite) first-order structures, satisfying order axioms. Moreover, belief revision focuses on axiomatic properties of belief revision operators and various notions of revision minimality, while preference revision focuses on axiomatic, order-theoretic properties of revised preferences.

We distinguish between *monotonic* and *nonmonotonic* preference revision. In the former, the original preference relation is fully incorporated into the revised one. In the latter, the original preference relation may conflict with the revising relation, leading to the necessity of retracting some of the original preferences. We focus on two special cases: *refinement* in which both the original and the revising relation are preserved (this is analogous to *expansion* in belief revision [Gärdenfors & Rott, 1995]), and *overriding revision* in which the revising relation may override the original one. We adopt the notion of minimal change based on symmetric difference between sets of tuples.

The challenges are: (1) to guarantee that suitable order properties, for example the axioms of strict partial orders, are preserved by the revisions, and (2) to obtain unique revisions. Strict partial orders (and weak orders), apart from being intuitive, enjoy a number of attractive properties in the context of preference queries, as explained later in the paper. So it is desirable for revisions to preserve such orders. The uniqueness property is also important from the user's point of view, as the user typically desires to obtain a single revised preference relation. The presence of multiple revision candidates necessitates some form of aggregation of or choice among the candidates. Fortunately, in the cases studied in this paper there exist least revisions preserving the appropriate order axioms, and thus uniqueness is obtained automatically.

We adopt the preference query framework of [Chomicki, 2003] (a similar model was described in [Kießling, 2002]), in which preference relations between tuples are defined by logical formulas. [Chomicki, 2003] proposed a new relational algebra operator called *winnow* that selects from its argument relation the *most preferred tuples* according to the given preference relation.

Example 1 *Consider the relation* $Car(Make, Year)$ *and the following preference relation* $\succ_{C_1}$ *between* Car *tuples:*

within each make, prefer a more recent car.

which can be defined as follows:

$$(m, y) \succ_{C_1} (m', y') \equiv m = m' \wedge y > y'.$$

*Research supported by NSF grant IIS-0307434.

The winnow operator ω_{C_1} returns for every make the most recent car available. Consider the instance r_1 of Car in Figure 1. The set of tuples $\omega_{C_1}(r_1)$ is shown in Figure 2.

	Make	*Year*
t_1	VW	2002
t_2	VW	1997
t_3	Kia	1997

Figure 1: The Car relation

	Make	*Year*
t_1	VW	2002
t_3	Kia	1997

Figure 2: The result of winnow

Example 2 *Example 1 provides a motivation for studying preference revision. Seeing the result of the query $\omega_{C_1}(r_1)$, a user may realize that the preference relation $\succ_{C_1}$ is not quite what she had in mind. The result of the query may contain some unexpected or unwanted tuples, for example t_3. Thus the preference relation needs to be modified, for example by refining it with the following preference relation $\succ_{C_2}$:*

$$(m, y) \succ_{C_2} (m', y') \equiv m = ''\text{VW}'' \wedge m' \neq ''\text{VW}'' \wedge y = y'.$$

The resulting refinement will contain both $\succ_{C_1}$ and $\succ_{C_2}$. The tuple t_3 is now dominated by t_2 and will not be returned to the user.

In the terminology used in research on preference reasoning in AI [Boutilier *et al.*, 2004], a relational database instance corresponds to the set of *feasible outcomes* and the winnow operator picks the undominated (best) outcomes from this set, according to the given preferences. A preference setting can be affected by a change in preferences or a modification of the set of possible outcomes. In this research, we address the former problem; the latter one, database update, has been extensively studied in database research. Moreover, we limit ourselves to preference revisions in which new preference information is combined, perhaps nonmonotonically, with the old one. We assume that the domains of preferences do not change in revisions.

2 Basic notions

We are working in the context of the relational model of data. Relation schemas consist of finite sets of attributes. For concreteness, we consider two infinite, countable domains: $\mathcal{D}$ (uninterpreted constants, for readability shown as strings) and $\mathcal{Q}$ (rational numbers), but our results, except where explicitly indicated, hold also for finite domains. We assume that database instances are finite sets of tuples. Additionally, we have the standard built-in predicates.

2.1 Preference relations

We adopt here the framework of [Chomicki, 2003].

Definition 2.1 *Given a relation schema $R(A_1 \cdots A_k)$ such that U_i, $1 \leq i \leq k$, is the domain (either $\mathcal{D}$ or $\mathcal{Q}$) of the attribute A_i, a relation $\succ$ is a* preference relation over R *if it is a subset of $(U_1 \times \cdots \times U_k) \times (U_1 \times \cdots \times U_k)$.*

Although we assume that database instances are finite, in the presence of infinite domains preference relations can be infinite.

Typical properties of a preference relation $\succ$ include:

- *irreflexivity*: $\forall x.\ x \not\succ x$;
- *transitivity*: $\forall x, y, z.\ (x \succ y \wedge y \succ z) \Rightarrow x \succ z$;
- *negative transitivity*: $\forall x, y, z.\ (x \not\succ y \wedge y \not\succ z) \Rightarrow x \not\succ z$;
- *connectivity*: $\forall x, y.\ x \succ y \vee y \succ x \vee x = y$;
- *strict partial order* (SPO) if $\succ$ is irreflexive and transitive;
- *interval order* (IO) if $\succ$ is an SPO and satisfies the condition $\forall x, y, z, w.\ (x \succ y \wedge z \succ w) \rightarrow (x \succ w \vee z \succ y)$;
- *weak order* (WO) if $\succ$ is a negatively transitive SPO;
- *total order* if $\succ$ is a connected SPO.

Every total order is a WO; every WO is an IO.

Definition 2.2 *A* preference formula (pf) *$C(t_1, t_2)$ is a first-order formula defining a preference relation $\succ_C$ in the standard sense, namely*

$$t_1 \succ_C t_2 \text{ iff } C(t_1, t_2).$$

An intrinsic preference formula (ipf) *is a preference formula that uses only built-in predicates.*

By using the notation $\succ_C$ for a preference relation, we assume that there is an underlying pf C. Occasionally, we will limit our attention to ipfs consisting of the following two kinds of atomic formulas (assuming we have two kinds of variables: $\mathcal{D}$-variables and $\mathcal{Q}$-variables):

- *equality constraints*: $x = y$, $x \neq y$, $x = c$, or $x \neq c$, where x and y are $\mathcal{D}$-variables, and c is an uninterpreted constant;
- *rational-order constraints*: $x\theta y$ or $x\theta c$, where $\theta \in \{=, \neq, <, >, \leq, \geq\}$, x and y are $\mathcal{Q}$-variables, and c is a rational number.

An ipf all of whose atomic formulas are equality (resp. rational-order) constraints will be called an *equality* (resp. *rational-order*) ipf. Clearly, ipfs are a special case of general constraints [Kuper, Libkin, & Paredaens, 2000], and define *fixed*, although possibly infinite, relations.

Every preference relation $\succ$ generates an indifference relation $\sim$: two tuples t_1 and t_2 are *indifferent* ($t_1 \sim t_2$) if neither is preferred to the other one, i.e., $t_1 \not\succ t_2$ and $t_2 \not\succ t_1$. We will denote by $\sim_C$ the indifference relation generated by $\succ_C$.

Composite preference relations are defined from simpler ones using logical connectives. We focus on two basic ways of composing preference relations:

- *union*:

$$t_1\ (\succ_1 \cup \succ_2)\ t_2 \text{ iff } t_1 \succ_1 t_2 \vee t_1 \succ_2 t_2;$$

- *prioritized composition* (where $\sim_1$ is the indifference relation generated by $\succ_1$):

$$t_1 \ (\succ_1 \rhd \succ_2) \ t_2 \text{ iff } t_1 \succ_1 t_2 \vee (t_1 \sim_1 t_2 \wedge t_1 \succ_2 t_2).$$

We also consider transitive closure:

Definition 2.3 *The* transitive closure *of a preference relation $\succ$ over a relation schema R is a preference relation $TC(\succ)$ over R defined as:*

$$(t_1, t_2) \in TC(\succ) \text{ iff } t_1 \succ^n t_2 \text{ for some } n > 0,$$

where:

$$t_1 \succ^1 t_2 \equiv t_1 \succ t_2$$
$$t_1 \succ^{n+1} t_2 \equiv \exists t_3.\, t_1 \succ t_3 \wedge t_3 \succ^n t_2.$$

Clearly, in general Definition 2.3 leads to infinite formulas. However, as shown in [Chomicki, 2003], in the cases that we consider in this paper the preference relation $\succ_{C*}$ will in fact be defined by a finite formula (this is because transitive closure can be expressed as a terminating Datalog program).

2.2 Winnow

We define now an algebraic operator that picks from a given relation the set of the *most preferred tuples*, according to a given preference relation.

Definition 2.4 [Chomicki, 2003] *If R is a relation schema and $\succ$ a preference relation over R, then the* winnow operator *is written as $\omega_\succ(R)$, and for every instance r of R:*

$$\omega_\succ(r) = \{t \in r \mid \neg\exists t' \in r.\, t' \succ t\}.$$

If a preference relation is defined using a pf C, we write simply ω_C instead of $\omega_{\succ_C}$. A *preference query* is a relational algebra query containing at least one occurrence of the winnow operator.

2.3 Preference revision

The basic setting is as follows: We have a preference relation $\succ$ and revise it with a *revising* preference relation $\succ_0$ to obtain a *revised* preference relation $\succ'$. We also call $\succ'$ a *revision* of $\succ$. We limit ourselves to preference relations over the same schema.

The revisions are characterized by a number of different parameters:

- *axiom preservation*: what order axioms are preserved in $\succ'$;
- *content preservation*: what preference relations are preserved in $\succ'$;
- *ordering* (of revisions).

Definition 2.5 *A revision $\succ'$ of $\succ$ with $\succ_0$ is:*

- *a* transitive (resp. SPO, WO) *revision if $\succ'$ is transitive (resp. an SPO, a WO);*
- *a* monotonic *revision if $\succ \subseteq \succ'$;*
- *a* refinement *revision (*refinement *for short) if $\succ \cup \succ_0 \subseteq \succ'$;*
- *an* overriding *revision if $\succ_0 \rhd \succ \subseteq \succ'$.*

A refinement is monotonic. An overriding revision does not have to be monotonic because it may fail to preserve $\succ$.

We order revisions using the symmetric difference ($\ominus$).

Definition 2.6 *Assume $\succ_1$ and $\succ_2$ are two revisions of a preference relation $\succ$ with a preference relation $\succ_0$. We say that $\succ_1$ is* closer *than $\succ_2$ to $\succ$ if $\succ_1 \ominus \succ \subset \succ_2 \ominus \succ$.*

Definition 2.7 *A* minimal (resp. least) *revision of $\succ$ with $\succ_0$ is a revision that is minimal (resp. least) in the closeness order among all revisions of $\succ$ with $\succ_0$.*

Similarly, we talk about least transitive refinements (or overriding revisions), least SPO (or WO) refinements or overriding revisions etc. If we consider only refinements or overriding revisions of a fixed preference relation, closeness reduces to set containment. Also, for finite domains the least element in a class of revisions minimizes the partial-order distance [Bogart, 1973] to the original relation $\succ$.

Example 3 *Consider the preference relation $\succ = \{(a,b),(b,c),(a,c)\}$ representing the preference order $a \succ b \succ c$, and the following revision of $\succ$, $\succ_1 = \{(b,a),(b,c),(a,c)\}$. The revision $\succ_1$ is the least SPO overriding revision of $\succ$ with $\succ_0 = \{(b,a)\}$. It achieves the effect of swapping a and b in the preference order. The partial-order distance of $\succ$ and $\succ_1$ is 2.*

To further describe the behavior of revisions, we define *preference conflicts*.

Definition 2.8 *A* conflict *between a preference relation $\succ$ and a preference relation $\succ_0$ is a pair (t_1, t_2) such that $t_1 \succ_0 t_2$ and $t_2 \succ t_1$. A* hidden conflict *between $\succ$ and $\succ_0$ is a pair (t_1, t_2) such that $t_1 \succ_0 t_2$ and there exist $s_1, \ldots s_k$, $k \geq 1$, such that $t_2 \succ s_1 \succ \cdots \succ s_k \succ t_1$ and $t_1 \not\succ_0 s_k \not\succ_0 \cdots \not\succ_0 s_1 \not\succ_0 t_2$.*

A hidden conflict is a conflict (if $\succ$ is an SPO) but not necessarily vice versa.

Example 4 *If $\succ_0 = \{(a,b)\}$ and $\succ = \{(b,a)\}$, then (a,b) is a conflict which is not hidden. If we add (b,c) and (c,a) to $\succ$, then the conflict is also a hidden conflict ($s_1 = c$). If we further add (c,b) or (a,c) to $\succ_0$, then the conflict is not hidden anymore.*

In this paper, we focus on refinement and overriding revisions because in our opinion they capture two basic ways of revising preferences. A refinement does not retract any preferences or resolve conflicts: it only adds new preferences necessitated by order properties. So for a refinement to satisfy SPO properties, all conflicts need to be avoided. An overriding revision, on the other hand, can override some of the original preferences if they conflict with the new ones. Overriding can deal with conflicts which are not hidden and solves all of them in the same fashion: it gives higher priority to new preference information (i.e., $\succ_0$). Both refinement and overriding revisions preserve the revising relation $\succ_0$.

We now characterize those combinations of $\succ$ and $\succ_0$ that avoid all (or only hidden) conflicts.

Definition 2.9 *A preference relation $\succ$ is* compatible *(resp.* semi-compatible*) with a preference relation $\succ_0$ if there are no conflicts (resp. no hidden conflicts) between $\succ$ and $\succ_0$.*

Compatibility is symmetric and implies semi-compatibility for SPOs. Semi-compatibility is not necessarily symmetric. Examples 1 and 2 show a pair of compatible relations. The compatibility of $\succ$ and $\succ_0$ *does not require* the acyclicity of $\succ \cup \succ_0$ or that one of the following hold: $\succ \subseteq \succ_0$, $\succ_0 \subseteq \succ$, or $\succ \cap \succ_0 = \emptyset$. For the former, consider $\succ = \{(a,b),(c,d)\}$ and $\succ_0 = \{(b,c),(d,a)\}$. For the latter, consider $\succ = \{(a,b),(b,c),(a,c)\}$ and $\succ_0 = \{(a,b),(a,d)\}$.

A semi-compatible relation $\succ_0$ may conflict with a given preference relation $\succ$. However, in each such case, i.e., when $t_1 \succ_0 t_2$ and $t_2 \succ t_1$, all the ways of deriving $t_2 \succ t_1$ by transitivity have at least one pair of tuples in conflict with some pair of tuples in $\succ_0$, and are therefore blocked.

All the properties listed above, including both variants of compatibility, are decidable for equality or rational order ipfs. For example, semi-compatibility is expressed by the condition $\succ_0^{-1} \cap TC(\succ - \succ_0^{-1}) = \emptyset$ where $\succ^{-1}$ is the inverse of the preference relation $\succ$.

The compatibility (resp. semi-compatibility) of $\succ$ and $\succ_0$ is a *necessary* condition for the refinements (resp. overriding revisions) of $\succ$ with $\succ_0$ to be SPOs. In the next section, we will see that those are not *sufficient* conditions: further restrictions on the preference relations will be introduced.

3 Preservation of order axioms

We prove now a number of results that characterize refinement and overriding revisions of of preference relations. The results are of the form:

Given that the original preference relation $\succ$ and the revising relation $\succ_0$ satisfy certain order axioms, what kind of order axioms does the revision $\succ'$ satisfy?

To capture minimal change of preferences, we typically study *least* revisions. The revision setting helps to overcome the limitations of *preference composition* [Chomicki, 2003] where it is shown that common classes of orders (SPOs, WOs) are often not closed w.r.t. basic preference composition operators like union or prioritized composition. In the results that follow, we obtain closure under least revisions thanks to (1) restricting $\succ$ and $\succ_0$, and (2) guaranteeing transitivity by explicitly applying transitive closure where necessary.

3.1 General properties

Lemma 3.1 *For compatible $\succ$ and $\succ_0$, $\succ_0 \cup \succ = \succ_0 \rhd \succ$.*

Lemma 3.2 *The preference relation $\succ \cup \succ_0$ (resp. $\succ_0 \rhd \succ$) is contained in every refinement (resp. overriding revision) of $\succ$ with $\succ_0$ and is, therefore, the least refinement (resp. least overriding revision) of $\succ$ with $\succ_0$.*

Lemma 3.3 *The preference relation $TC(\succ \cup \succ_0)$ (resp. $TC(\succ_0 \rhd \succ)$) is contained in every transitive refinement (resp. every overriding revision) of $\succ$ with $\succ_0$ and is, therefore, the least transitive refinement (resp. least transitive overriding revision) of $\succ$ with $\succ_0$.*

3.2 Strict partial order revisions

SPOs have several important properties from the user's point of view, and thus their preservation is desirable. For instance, all the preference relations defined in [Kießling, 2002] and in the language Preference SQL [Kießling & Köstler, 2002] are SPOs. Moreover, if $\succ$ is an SPO, then the winnow $\omega_\succ(r)$ is nonempty if (a finite) r is nonempty. The fundamental algorithms for computing winnow require that the preference relation be an SPO [Chomicki, 2003]. Also, in that case incremental computation of revised preference queries becomes possible (Proposition 5.1).

In order to obtain the least SPO revisions, we have to make sure that $TC(\succ \cup \succ_0)$ and $TC(\succ \rhd \succ_0)$ are irreflexive (they are transitive by definition).

Theorem 3.1 *For every compatible preference relations $\succ$ and $\succ_0$ such that one is an interval order (IO) and the other an SPO, the preference relation $TC(\succ \cup \succ_0)$ is the least SPO refinement of $\succ$ with $\succ_0$. Additionally, if the IO is a WO, then $TC(\succ \cup \succ_0) = \succ \cup \succ_0$.*

It seems that the IO requirement in Theorem 3.1 cannot be weakened without needing to strengthen the remaining assumptions. If neither of $\succ$ and $\succ_0$ is an IO, then we can find such elements $x_1, y_1, z_1, w_1, x_2, y_2, z_2, w_2$ that

$$x_1 \succ y_1, z_1 \succ w_1, x_1 \not\succ w_1, z_1 \not\succ y_1$$

and

$$x_2 \succ_0 y_2, z_2 \succ_0 w_2, x_2 \not\succ_0 w_2, z_2 \not\succ_0 y_2.$$

If we can choose $y_1 = x_2$, $z_1 = y_2$, $w_1 = z_2$, and $x_1 = w_2$, then we get a cycle in $\succ \cup \succ_0$. Note that in this case: (1) $\succ$ and $\succ_0$ are still compatible, and (2) there is no SPO refinement of $\succ$ with $\succ_0$.

Example 5 *Consider again the preference relation $\succ_{C_1}$:*

$$(m,y) \succ_{C_1} (m',y') \equiv m = m' \wedge y > y'.$$

Suppose that the new preference information is captured as $\succ_{C_3}$ which is an IO but not a WO:

$$\begin{array}{rcl} (m,y) \succ_{C_3} (m',y') & \equiv & m = \text{''VW''} \wedge y = 1999 \\ & \wedge & m' = \text{''Kia''} \wedge y' = 1999. \end{array}$$

Then $TC(\succ_{C_1} \cup \succ_{C_3})$, which properly contains $\succ_{C_1} \cup \succ_{C_3}$, is defined as the SPO $\succ_{C_4}$:

$$\begin{array}{l} (m,y) \succ_{C_4} (m',y') \equiv m = m' \wedge y > y' \\ \vee\ m = \text{''VW''} \wedge y \geq 1999 \wedge m' = \text{''Kia''} \wedge y' \leq 1999. \end{array}$$

For dealing with overriding revisions compatibility can be replaced by a less restrictive condition, *semi-compatibility*, because prioritized composition already provides a way of resolving some conflicts.

Theorem 3.2 *For every preference relations $\succ$ and $\succ_0$ such that $\succ_0$ is an IO, $\succ$ is an SPO and $\succ$ is semi-compatible with $\succ_0$, the preference relation $TC(\succ_0 \rhd \succ)$ is the least SPO overriding revision of $\succ$ with $\succ_0$.*

Again, violating any of the conditions of Theorem 3.2 may lead to a situation in which no SPO overriding revision exists.

If $\succ_0$ is a WO, the requirement of semi-compatibility and the computation of transitive closure are unnecessary.

Theorem 3.3 *For every preference relations $\succ_0$ and $\succ$ such that $\succ_0$ is a WO and $\succ$ an SPO, the preference relation $\succ_0 \rhd \succ$ is the least SPO overriding revision of $\succ$ with $\succ_0$.*

Proposition 3.1 *For the preference relations defined using equality or rational order ipfs, the computation of $TC(\succ \cup \succ_0)$ and $TC(\succ \rhd \succ_0)$ terminates.*

The computation of transitive closure is done in a completely database-independent way using Constraint Datalog techniques [Kuper, Libkin, & Paredaens, 2000].

Example 6 *Consider Examples 1 and 5. We can infer that*

$$t_1 = (''\mathrm{VW}'', 2002) \succ_{C_4} (''\mathrm{Kia}'', 1997) = t_3,$$

because

$$(''\mathrm{VW}'', 2002) \succ_{C_1} (''\mathrm{VW}'', 1999),$$
$$(''\mathrm{VW}'', 1999) \succ_{C_3} (''\mathrm{Kia}'', 1999),$$

and

$$(''\mathrm{Kia}'', 1999) \succ_{C_1} (''\mathrm{Kia}'', 1997).$$

The tuples $(''\mathrm{VW}'', 1999)$ *and* $(''\mathrm{Kia}'', 1999)$ *are* not *in the database.*

3.3 Weak order revisions

Weak orders are practically important because they capture the situation where the domain can be decomposed into layers such that the layers are totally ordered and all the elements in one layer are mutually indifferent. This is the case, for example, if the preference relation can be represented using a numeric utility function. If the preference relation is a WO, a particularly efficient (essentially single pass) algorithm for computing winnow is applicable [Chomicki, 2004].

Theorem 3.4 *For every compatible WO preference relations $\succ$ and $\succ_0$, the preference relation $\succ \cup \succ_0$ is the least weak order refinement of $\succ$ with $\succ_0$.*

Again, for overriding revisions, we can relax the compatibility assumption. This immediately follows from the fact that WOs are closed with respect to prioritized composition [Chomicki, 2003].

Proposition 3.2 *For every WO preference relations $\succ$ and $\succ_0$, the preference relation $\succ_0 \rhd \succ$ is the least weak order overriding revision of $\succ$ with $\succ_0$.*

A basic notion in utility theory is that of *representability* of preference relations using numeric utility functions:

Definition 3.1 *A real-valued function u over a schema R* represents *a preference relation $\succ$ over R iff*

$$\forall t_1, t_2 \; [t_1 \succ t_2 \text{ iff } u(t_1) > u(t_2)].$$

Being a WO is a necessary condition for the existence of a numeric representation for a preference relation. However, it is not sufficient for uncountable orders [Fishburn, 1970]. It is natural to ask whether the existence of numeric representations for the preference relations $\succ$ and $\succ_0$ implies the existence of such a representation for the least refinement $\succ' = (\succ \cup \succ_0)$. This is indeed the case.

Theorem 3.5 *Assume that $\succ$ and $\succ_0$ are WO preference relations such that*

1. *$\succ$ and $\succ_0$ are compatible,*
2. *$\succ$ can be represented using a real-valued function u,*
3. *$\succ_0$ can be represented using a real-valued function u_0.*

Then $\succ' = \succ \cup \succ_0$ is a weak order preference relation that can be represented using any real-valued function u' such that for all x, $u'(x) = a \cdot u(x) + b \cdot u_0(x) + c$ where $a, b > 0$.

Surprisingly, the compatibility requirement cannot in general be replaced by semi-compatibility if we replace $\cup$ by $\rhd$ in Theorem 3.5. This follows from the fact that the lexicographic composition of one-dimensional standard orders over $\mathcal{R}$ is not representable using a utility function [Fishburn, 1970]. Thus, preservation of *representability* is possible only under compatibility, in which case $\succ_0 \rhd \succ = \succ_0 \cup \succ$ (Lemma 3.1) and the revision is monotonic.

We conclude this section by showing a general scenario in which the refinement of WOs occurs in a natural way. Assume that we have a numeric utility function u representing a (WO) preference relation $\succ$. The indifference relation $\sim$ generated by $\succ$ is defined as:

$$x \sim y \;\equiv\; u(x) = u(y).$$

Suppose that the user discovers that $\sim$ is too coarse and needs to be further refined. This may occur, for example, when x and y are tuples and the function u takes into account only some of their components. Another function u_0 may be defined to take into account other components of x and y (such components are called *hidden attributes* [Pu, Faltings, & Torrens, 2003]). The revising preference relation $\succ_0$ is now:

$$x \succ_0 y \;\equiv\; u(x) = u(y) \wedge u_0(x) > u_0(y).$$

It is easy to see that $\succ_0$ is an SPO compatible with $\succ$ but not necessarily a WO. Therefore, by Theorem 3.1 the preference relation $\succ \cup \succ_0$ is the least SPO refinement of $\succ$ with $\succ_0$.

4 Checking axiom satisfaction

If none of the results described so far implies that the least transitive refinement of $\succ$ with $\succ_0$ is an SPO, then this condition can often be explicitly checked. Specifically, one has to: (1) compute the transitive closure $TC(\succ \cup \succ_0)$, and (2) check whether the obtained relation is irreflexive.

From Proposition 3.1, it follows that for equality and rational order ipfs the computation of $TC(\succ \cup \succ_0)$ yields some finite ipf $C(t_1, t_2)$. Then the second step reduces to checking whether $C(t, t)$ is unsatisfiable, which is a decidable problem for equality and rational order ipfs.

Example 7 *Consider Examples 1 and 2. Neither of the preference relations $\succ_{C_1}$ and $\succ_{C_2}$ is a weak or interval order. Therefore, the results established earlier in this paper do not apply. The preference relation $\succ_{C*} = TC(\succ_{C_1} \cup \succ_{C_2})$ is defined as follows:*

$$(m, y) \succ_{C*} (m', y') \;\equiv\; m = m' \wedge y > y' \\ \vee\; m = ''\mathrm{VW}'' \wedge m' \neq ''\mathrm{VW}'' \wedge y \geq y'$$

The preference relation $\succ_{C*}$ is irreflexive (this can be effectively checked). It also properly contains $\succ_{C_1} \cup \succ_{C_2}$, because $t_1 \succ_{C*} t_3$ but $t_1 \not\succ_{C_1} t_3$ and $t_1 \not\succ_{C_2} t_3$. The query $\omega_{C*}(Car)$ evaluated in the instance r_1 (Figure 1) returns only the tuple t_1.

Similar considerations apply to overriding revisions and WOs.

5 Iterating monotonic preference revision

Consider the scenario in which we iterate monotonic preference revision to obtain a sequence of preference relations $\succ_1, \ldots, \succ_n$ such that each is an SPO and $\succ_1 \subseteq \cdots \subseteq \succ_n$. (Recall that refinement is monotonic but overriding revision not necessarily so.) Assume that those relations are used to extract the best tuples from a fixed relation instance r. Such evaluation provides feedback to the user about the quality of the given preference relation and may be helpful in constructing its subsequent refinements.

In this scenario, the sequence of query results is:

$$r_0 = r, r_1 = \omega_{\succ_1}(r), r_2 = \omega_{\succ_2}(r), \ldots, r_n = \omega_{\succ_n}(r).$$

Proposition 5.1 below implies that the sequence $r_0, r_1, \ldots, r_n$ is decreasing:

$$r_0 \supseteq r_1 \supseteq \cdots \supseteq r_n$$

and that it can be computed incrementally:

$$r_1 = \omega_{\succ_1}(r_0), r_2 = \omega_{\succ_2}(r_1), \ldots, r_n = \omega_{\succ_n}(r_{n-1}).$$

To compute r_i, there is no need to look at the tuples in $r - r_{i-1}$, nor to recompute winnow from scratch. The sets of tuples $r_1, \ldots, r_n$ are likely to have much smaller cardinality than $r_0 = r$.

Proposition 5.1 [Chomicki, 2003] *If $\succ_1$ and $\succ_2$ are preference relations over a relation schema R and $\succ_1 \subseteq \succ_2$, then for all instances r of R:*

- $\omega_{\succ_2}(r) \subseteq \omega_{\succ_1}(r)$;
- $\omega_{\succ_2}(\omega_{\succ_1}(r)) = \omega_{\succ_2}(r)$ *if $\succ_1$ and $\succ_2$ are SPOs.*

6 Related work

[Hansson, 1995] presents a general framework for modeling change in preferences. Preferences are represented syntactically using sets of ground preference formulas, and their semantics is captured using sets of preference relations. Thanks to the syntactic representation preference revision is treated similarly, though not identically, to belief revision, and some axiomatic properties of preference revisions are identified. The result of a revision is supposed to be minimally different from the original prefence relation (using a notion of minimality based on symmetric difference) and satisfy some additional background postulates, for example specific order axioms. [Hansson, 1995] does not address the issue of constructing revised relations, does not characterize cases when the desired revised preference relation is unique, nor does it study the properties of specific classes of preference relations. On the other hand, [Hansson, 1995] discusses also preference contraction, and domain expansion and shrinking. [Williams, 1997] considers revising a ranking (a WO) of a finite set of product profiles with new information, and shows that a new ranking, satisfying the AGM revision postulates [Gärdenfors & Rott, 1995], can be computed in a simple way. [Pu, Faltings, & Torrens, 2003] formulates different scenarios of preference revision and does not contain any formal framework. [Freund, 2004] describes minimal change revision of *rational* preference relations between propositional formulas.

7 Conclusions and future work

We have presented a general framework for revising preference relations and established a number of order axiom preservation results for specific classes of revisions. In the future, we plan to consider more general classes of revisions and databases with restricted domains, e.g., Boolean. Another direction is the design of a *revision language* in which different parameters of preference revision can be explicitly specified by the user. Connections to *iterated belief revision* [Darwiche & Pearl, 1997] should also be explored.

References

[Bogart, 1973] Bogart, K. P. 1973. Preference Structures I: Distances Between Transitive Preference Relations. *Journal of Mathematical Sociology* 3:49–67.

[Boutilier *et al.*, 2004] Boutilier, C.; Brafman, R. I.; Domshlak, C.; Hoos, H. H.; and Poole, D. 2004. CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements. *Journal of Artificial Intelligence Research* 21:135–191.

[Chomicki, 2003] Chomicki, J. 2003. Preference Formulas in Relational Queries. *ACM Transactions on Database Systems* 28(4):427–466.

[Chomicki, 2004] Chomicki, J. 2004. Semantic Optimization of Preference Queries. In *International Symposium on Constraint Databases*, 133–148. Paris, France: Springer-Verlag, LNCS 3074.

[Darwiche & Pearl, 1997] Darwiche, A., and Pearl, J. 1997. On the Logic of Iterated Belief Revision. *Artificial Intelligence* 89(1–2):1–29.

[Fishburn, 1970] Fishburn, P. C. 1970. *Utility Theory for Decision Making*. Wiley & Sons.

[Freund, 2004] Freund, M. 2004. On the Revision of Preferences and Rational Inference Processes. *Artificial Intelligence* 152:105–137.

[Gärdenfors & Rott, 1995] Gärdenfors, P., and Rott, H. 1995. Belief Revision. In Gabbay, D. M.; Hogger, C, J.; and Robinson, J. A., eds., *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 4. Oxford University Press. 35–132.

[Hansson, 1995] Hansson, S. O. 1995. Changes in Preference. *Theory and Decision* 38:1–28.

[Kießling & Köstler, 2002] Kießling, W., and Köstler, G. 2002. Preference SQL - Design, Implementation, Experience. In *International Conference on Very Large Data Bases (VLDB)*, 990–1001.

[Kießling, 2002] Kießling, W. 2002. Foundations of Preferences in Database Systems. In *International Conference on Very Large Data Bases (VLDB)*, 311–322.

[Kuper, Libkin, & Paredaens, 2000] Kuper, G.; Libkin, L.; and Paredaens, J., eds. 2000. *Constraint Databases*. Springer-Verlag.

[Pu, Faltings, & Torrens, 2003] Pu, P.; Faltings, B.; and Torrens, M. 2003. User-Involved Preference Elicitation. In *IJCAI Workshop on Configuration*.

[Williams, 1997] Williams, M.-A. 1997. Belief Revision via Database Update. In *International Intelligent Information Systems Conference*.