
Operators and Laws for Combining Preference Relations

HAJNAL ANDRÉKA, *Mathematical Institute, Hungarian Academy of Science, Budapest Pf. 127 H-1364, Hungary.*

E-mail: andreka@math-inst.hu

MARK RYAN, *School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK.*

E-mail: mdr@cs.bham.ac.uk

PIERRE-YVES SCHOBENS, *Institut d'Informatique, Facultés Universitaires de Namur, Rue Grandgagnage 21, 5000 Namur, Belgium.*

E-mail: pys@info.fundp.ac.be

Abstract

The paper is a theoretical study of a generalization of the lexicographic rule for combining ordering relations. We define the concept of priority operator: a priority operator maps a family of relations to a single relation which represents their lexicographic combination according to a certain priority on the family of relations. We present four kinds of results.

- We show that the lexicographic rule is the only way of combining preference relations which satisfies natural conditions (similar to those proposed by Arrow).
- We show in what circumstances the lexicographic rule propagates various conditions on preference relations, thus extending Grosz's results.
- We give necessary and sufficient conditions on the priority relation to determine various relationships between combinations of preferences.
- We give an algebraic treatment of this form of generalized prioritization. Two operators, called *but* and *on the other hand*, are sufficient to express any prioritization. We present a complete equational axiomatization of these two operators.

These results can be applied in the theory of social choice (a branch of economics), in non-monotonic reasoning (a branch of artificial intelligence), and more generally wherever relations have to be combined.

Keywords: Preference relations, priority relations, default reasoning, lexicographic combination, Arrows theorem, social choice.

1 Introduction

The lexicographic combination of orderings constructs a single ordering from several individual ones. Traditionally, the individual orderings will order words according to their *i*th letter using alphabetical ordering, and the combination will then be the usual ordering of dictionaries. This combination thus says that a letter on the left is more important than any letter on its right, thereby giving a *priority* between letter indices. If the first letter of the first word is strictly before the first letter of the second word, this first word will indeed appear first in the dictionary. In case of ties, the second ordering will be used, and so on.

In this paper we study a generalization of this combination of relations, in which the priority ordering on the indices may be an arbitrary order instead of a finite linear one, and the relations themselves need not be orders.

Applications of this work potentially include any application of the lexicographic rule in computer science and artificial intelligence, and are therefore varied and widespread. We mention some of them here:

Artificial intelligence. Default logics have been used in AI for twenty years [13, 5]. The lexicographic rule was first proposed for prioritized defaults by Lifschitz [19, 20] in the setting of circumscription. Later, Grosz [14] recognized its applicability to any preferential logic, and dubbed it *generalized prioritization*. The lexicographic rule has also been used for preferential logics in Ryan [25] and Schobbens [30]. In this context, a priority operator is a policy for controlling which defaults represent exceptions for which other defaults. In the specific case of circumscription, a priority operator is a circumscription policy. The lexicographic rule has also been used in belief revision [28].

Requirements specification. The requirements that users may specify are often soft, and as such express a preference over a set of possible implementations rather than a hard set of implementations. Inconsistencies easily arise if the requirements are interpreted as hard, whereas resolving a set of soft requirements involves finding a compromise between the preferences each requirement denotes. Priority operators in this setting represent a policy for putting together the requirements.

Concretely, the use of default constraints in specifications has been proposed for modelling requirements [4, 30, 26, 27, 15]. The priority operator used to put together the preferences on models these defaults express may be derived from the structure of the specification [26], the use of a logical connective ‘but’ expressing exceptions [30], or an explicit hierarchy [9].

Economics. Preferences originate from economics, and naturally our work can also be used there. Two subdomains are more particularly concerned:

Social choice. The study of combinations of preferences for social choice was initiated by Condorcet [7]. Here, each input relation represents the preferences of a member of the group, and the output represents the preferences of the group. This domain has yielded mostly negative results, the best known being Arrow’s impossibility of combining linear orders under very natural conditions [1] recalled in section 3. In this paper, we show that, surprisingly, when working in the slightly more general settings of relations, or even pre-orders, we obtain, on the contrary, a possibility theorem, yielding our lexicographic combinations as the only solution. Various extensions of the lexicographic combination were also studied in [11, 12, 3, 17, 18].

Multi-criteria decision. Currently these results are more used in a different branch of economics, multi-criteria decision. Arrow has rewritten his results with this application in mind in [2]. Here, the input relations represent rankings according to the various relevant criteria, and the single output represents their combination, on which the final choice will be based.

This section intuitively introduces the problems and the solutions considered in this paper. We use an example from Economics, since such examples are readily explained from common sense.

EXAMPLE 1.1

Claire and Bob have to replace their old car. As often, they have different criteria for selecting

the new car, although some of them are common, but ranked differently.

The preference of Claire is guided by the following criteria (in increasing order of importance):

- the maximum speed (M);
- the elegance of the design (D);
- the ease with which it can be driven in town (E);
- the price (P).

The criteria for Bob are ranked differently:

- the ease with which the car can be driven in town (E);
- the maximum speed (M);
- the price (P).

Some of these criteria are simple, and can be directly computed from the technical data of the car. Others can be decomposed, say: the ease with which the car can be driven in town (E) is an aggregation of:

- the length of the car (L);
- its weight (W);
- its turning circle diameter (C);
- the presence of automatic transmission (A).

Let us say the last one is the most important, the other ones are equally important, but are clearly expressed in incomparable units, so that, for instance, adding them makes no sense. The final choice should at least be Pareto-optimal: no other car will be better for both Claire and Bob than the one selected.

Now, these criteria must be applicable to any specific market. In this paper, we do not work directly with numerical criteria like the ones above. We consider the market M containing economic alternatives, in this case the various cars that are available; say $M = \{t, h, r, m, n\}$. The numerical criteria are converted into a preference ordering. For instance, if the actual characteristics of the cars are as in Table 1, we forget the numeric values to remember only

TABLE 1. Car characteristics

		t	h	r	m	n
length	L	3.5	3.5	7.3	5.0	3.7
weight	W	0.7	0.9	3.5	1.5	0.7
turning circle diameter	C	3.2	3.4	6.4	3.4	3.2
automatic transmission	A	N	Y	Y	N	N
maximal speed	M	110	130	180	250	120
price	P	10	10	100	20	11

their ordering. For example, for the turning circle diameter (C), we remember only that t is equivalent to n (in the notation of the main part of this paper, $t R_C^= n$), while n is strictly preferred to h (written $n R_C^< h$), and so on: in summary, $t R_C^= n R_C^< h R_C^= m R_C^< r$. In some cases, no meaningful comparison can be established, so that both incomparable alternatives

should be kept in the final choice. For instance a shoe cannot be compared to a car, say. We write this $s R^{\#} c$.

In the example, all preferences are transitive, and this is usually considered as a condition for them to be rational. However, many empirical studies have shown that intransitive preferences are the norm rather than the exception for human decision makers. Therefore, this study does not assume transitivity, but intends to preserve it when it exists. That is to say, when the underlying preferences are transitive, so should be their combination. We shall use (T) to refer to preservation of transitivity. We assume several other properties of the combination. It should not advantage any alternative except from the selected criteria (B), and should respect the criteria when they are unanimous (U). Finally, alternatives that are not involved in a comparison should not influence the result (I): for instance, if m is preferred to n , this should not depend on whether h is present in the market M or not, but only on the performance of m, n for the selected criteria.

If we accept these natural rationality postulates (IBUT), we demonstrate below that the problem can be expressed by priority graphs, or by algebraic expressions. For instance, the algebraic expressions for the example above are:

$$\begin{aligned} \text{Claire} &= M/D/E/P \\ \text{Bob} &= E/M/P \\ \text{where } E &= (L\|W\|C)/A \\ \text{Result} &= \text{Bob}\|\text{Claire} \end{aligned}$$

where $/$ expresses priority of the second term, while $\|$ puts both sides on equal priority. In this example, our theory shows how to simplify the computations: it is useless to repeat the computation of E for Bob, of M for Claire, since anyway these criteria will be better taken into account by the other person. So $\text{Result} = (M\|(D/E))/P$ gives the same result more efficiently. It is also clear from this expression that h is to be chosen in the example, without even looking at criteria L, W, C, D .

Our principal definition is that of *priority operator*. A priority operator specifies a way of putting together a family of relations to make a single relation. We call these relations *preference relations*: the idea is that they relate elements of M (interpretations, economic alternatives, etc.) according to some preference criterion.

We present results of four kinds.

1. We show that priority operators are canonical: they are the only way of combining preference relations with different priorities which satisfies the very natural conditions above, inspired by Arrow [1, 2].
2. Next, we define several natural properties of preference relations: transitivity, reflexivity, irreflexivity, and well-foundedness. We show in what circumstances these properties are *propagated* by priority operators. This generalizes a result by Grosf [14].
3. We give necessary and sufficient conditions on the priority relation to determine whether the result of a priority operator is always *included* in the result of another combination. This also extends a result of Grosf [14]. We also give necessary and sufficient conditions for other relationships between the results of priority operators, such as *equality* and *preferential entailment*.
4. We give an algebraic treatment of generalized prioritization. We formally define two binary priority operators, called *but* and *on the other hand*, and show them to be sufficient

to express any priority operator. We present a complete equational axiomatization of these two operators.

The structure of the paper is as follows. The next section presents basic definitions. Section 3 presents the results which show that the lexicographic rule is the only way of combining preference relations that satisfies the natural generalization of Arrow's conditions. Propagation of properties of preference relations by the rule is summarized in Section 4, Table 3. Section 5 develops proof rules for priority graphs, and 6 explores composition of priority operators. Section 7 summarizes our algebraic treatment of priority operators, and conclusions are drawn in Section 8.

There is a long appendix to this paper, which covers the mathematical details and proofs which have been omitted from the text in order not to interrupt the flow. The structure of the appendix mirrors that of the paper.

2 Priority operators

Let M be a set containing at least two elements. The elements of M are the subject of the preferences: in the example above, it was the set of cars which were available on the market. From the point of view of our application to default reasoning, M is the set of interpretation structures of the logic. Default rules or formulas express preferences on M . The results presented in the paper work for any applications of prioritized preference, such as default reasoning, social choice or multi-criteria decision. M is simply the set of objects which are ordered by preference, which in economics are called economic alternatives. (Of course there must be at least two of them, otherwise there is nothing to choose.)

DEFINITION 2.1

A *preference relation* (sometimes just called a *preference*) is any binary relation on M . Preference relations will be written R, R_1, R_2, \dots , or $R', R'' \dots$.

For intuition, the reader will be helped by reading R as meaning 'better than, or indifferent' or 'as preferred as'. We do not assume that R is transitive and reflexive, since our mathematical results do not depend on these properties.

In the non-monotonic application, each default formula denotes a preference relation on M which orders interpretations according to how nearly they satisfy the default information. As usual in the literature, interpretations 'lower' in the relation are those which are closer to satisfying the default. For $m, n \in M$, the expression $m R n$ means that m is as preferred as n .

DEFINITION 2.2

Given a preference relation R , we define the derived relations

$m \bar{R} n$	iff	not mRn .	'not better (nor indifferent)'
$m R^< n$	iff	mRn and not nRm .	'strictly better'
$m R^\equiv n$	iff	mRn and nRm .	'indifferent'
$m R^\# n$	iff	neither mRn nor nRm .	'incomparable'

We also use F to denote the full relation $M \times M$, and \emptyset to denote the empty relation. Thus, $\bar{F} = F^< = F^\# = \emptyset^< = \emptyset^\equiv = \emptyset$ and $F^\equiv = \emptyset^\# = \emptyset = F$.

Now suppose we have a family of preference relations $(R_x)_{x \in V}$, all on the same set M . This can come about because we have several defaults, each of them denoting a preference

relation among interpretations of a non-monotonic logic, or because we have several deciders, each having its own preference among the economic alternatives. Also, the preferences can originate from different criteria that we wish to combine according to their importance. We want to combine these relations into a single relation on the same set M . The next step is usually to pick the minimal (or preferred) interpretations (or alternatives) according to it.

DEFINITION 2.3

A V -ary operator is any map taking some preference relations $(R_x)_{x \in V}$ and returning a single preference relation. (V may be infinite.)

Of particular interest are operators which combine preference relations according to some priority, which is a strict partial order on V .

The lexicographic combination of $(R_x)_{x \in V}$ ($V \neq \emptyset$) according to priority $<$ on V is the relation R given by

$$mRn \iff \forall x \in V. (mR_x n \vee \exists y \in V. (y < x \wedge mR_y^< n)). \quad (*)$$

This generalizes the familiar rule used for the alphabetic ordering of words in a dictionary, by allowing the priority $<$ (position of letter in word) to be an arbitrary partial order, and by allowing the preference relations (ordering of letters in alphabet) to be an arbitrary relation. Intuitively, the lexicographic rule says that m is preferred to n overall if it is preferred at each index, except possibly those for which there is an index of greater priority at which m is strictly preferred to n . To understand how this reduces to the familiar alphabetic ordering when $<$ is a finite total order (among positions in the word), observe that it says: in order that word m comes before (or equal) word n , we must have that for any x , the x th letter of m precedes or equals the x th letter of n , unless there was a smaller y such that the y th letter of m strictly precedes the y th letter of n .

A number of definitions of the lexicographic ordering, which are all equivalent when used with a finite linear priority, can be found in the literature:

1. $aR^<b$ iff $\exists z : aR_z^<b$ and $\forall x < z, aR_x^=b$ [23, p.49];
2. $aR^<b$ iff $D = \{x | aR_x^=b\}$ is not empty and $aR_z^<b$, where z is the $<$ -minimum element of D [12, p.1442];
3. aRb iff $\forall x(\forall y < x aR_y^=b) \Rightarrow aR_x b$ [14].

When we generalize to a *partially* ordered priority:

- Definition 1 may yield both $aR^<b$ and $bR^<a$, and is thus not useful in this context.
- Definition 2 needs to be generalized, since D will not have a single minimum but a set of minimals. So we could require that $aR_z^<b$ for all these minimals.
- Definition 3 is directly usable.

Definition 3, and our generalization of Definition 2, are each equivalent to our definition in equation (*) under the assumption that $<$ is well-founded (see Theorem 2.11). This is an assumption we will make frequently in the paper; it is generally valid for applications.

The formulation (*) of the lexicographic combination is not as general as we would like, however, because it forbids us from replicating an argument R_x several times in the prioritization. We can generalize it by considering the following notion of priority graph.

DEFINITION 2.4

A *priority graph* is a tuple $(N, <, v)$ where N is a set (of ‘nodes’), $<$ is a strict partial order on N (the ‘priority relation’) and v is a function from N to a set of variables. N may be infinite.

This definition and the following one are the most fundamental in the paper; everything else depends on them. So, what is a priority graph? It is just an ordering of variables, but crucially it allows some variables to be represented several times in the ordering, simply by repeating the variable in the priority graph. (A priority graph essentially represents a policy for prioritizing certain things represented by the variables, and the ability to allow repetition of the variables greatly increases the expressive power of the representation. We will prove this later.)

A priority graph denotes an operator on preference relations. The operator it denotes combines its arguments according to the given priority, using the lexicographic rule.

DEFINITION 2.5

The V -ary operator o denoted by the priority graph $(N, <, v)$ is given by

$$m o((R_x)_{x \in V}) n \iff \forall i \in N. (m R_{v(i)} n \vee \exists j \in N. (j < i \wedge m R_{v(j)}^< n))$$

where $V = v[N]$, the variables that occur in the graph.

This says that the variables in the priority graph are instantiated to be the argument preference relations. The operator returns the preference relation, which is their prioritized combination according to $<$, using the lexicographic rule.

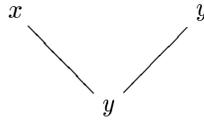
The difference between Definition 2.5 and equation (*) is that the elements of N are ordered, rather than the elements of V directly. The onus is on us to show that this added complication is really useful. It turns out to be useful because the ability to duplicate one of the arguments R_x in the ordering increases the expressive power we are giving to priority operators. This is shown by Example 2.8 below.

Our notion of priority operator can now be seen to generalize the notion of circumscription policy [20] in three ways.

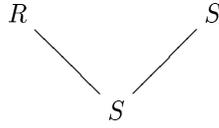
- it works for arbitrary preferential logics;
- it allows the priority to be partial;
- it allows repetition of the prioritized criteria in the ordering; and this increases the expressive power (Example 2.8 below).

EXAMPLE 2.6

Consider the priority graph $g_1 = (N, <, v)$ given by $N = \{1, 2, 3\}$ with $1 < 2$ and $1 < 3$ and $v(1) = y$, $v(2) = x$ and $v(3) = y$. Priority graphs will normally be written using a graphical notation in which we leave out the names of elements of N , showing the base of the partial order $<$ on the variables given by v (this is usually called the Hasse diagram of the priority). Recall that elements with the highest priority are, surprisingly perhaps, written at the bottom of our diagrams. The priority graph g_1 is:



This denotes a binary operator since there are only two distinct variables in the graph. It takes two preference relations, say R and S , and returns a preference relation which represents the combination of R and S with the priority which represents R once and S twice. One of the representations of S has priority over the other and over R . Thus, if o_1 is the operator denoted by the graph, then $o_1(R, S)$ is the following prioritized combination of R and S :



Applying the definition of the lexicographical rule (and simplifying), we obtain that $o_1(R, S) = (R \cap S) \cup S^<$. We may also write $o_1 = \lambda x, y. (x \cap y) \cup y^<$, although we will generally leave out λ s and details of variable binding, and write $o_1 = (x \cap y) \cup y^<$.

There may be several graphical representations of the same operator. As a trivial example, any priority graph whose nodes are all labelled by the same variable x denotes the identity operator, which is the only unary priority operator.

DEFINITION 2.7

Priority graphs g_1, g_2 are said to be *equivalent*, written $g_1 \equiv g_2$, if they denote the same operator on preference relations.

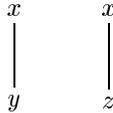
The graph g_1 in the preceding example is equivalent to the graph g_2



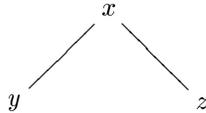
(which does not have any repetition of variables), in the sense that the two graphs denote the same operator $o_1 = (x \cap y) \cup y^<$.

EXAMPLE 2.8

The priority graph g_3

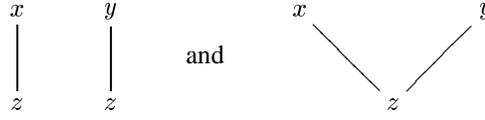


denotes the operator $o_3 = [x \cup (y^< \cap z^<)] \cap y \cap z$, and is *not* equivalent to any graph which does not repeat the variable x (this will be proved later, in Example 5.7). In particular, it is not equivalent to



which denotes $[x \cup (y \cap z)^<] \cap y \cap z$. To see that these expressions may be different, try $M = \{1, 2\}$, $x = \emptyset$, $y = M \times M$, $z = \{(1, 1), (1, 2), (2, 2)\}$. Then the first expression yields \emptyset , while the second one yields $\{(1, 2)\}$.

EXAMPLE 2.9
The graphs



denote the same operator, namely $(x \cap y \cap z) \cup z^<$.

The lexicographic rule applied to graphs is not the only way of defining operators on relations, but is an important one.

DEFINITION 2.10

A *priority operator* is an operator which is denoted by some priority graph.

By convention, we extend the usual properties of posets to priority graphs and thence to operators in the obvious way: for instance, we say that a priority operator is *well-founded* iff there is a graph $(N, <, v)$ denoting it such that $(N, <)$ is well-founded, (i.e. there is no infinite descending sequence $i_1 > i_2 > i_3 > \dots, i_n \in N$). A V -ary operator is *finitary* if V is finite.

Notice that the identity of nodes (elements of N) in a priority graph is irrelevant. For this reason we can think of priority graphs as partially ordered multisets (*pomsets* [24]) of variables.

The following theorem is useful in two respects. First, it should help the reader build up intuitions for the behaviour of the lexicographic rule coded into Definition 2.5. Secondly, it will be used for proving most results in all later sections, e.g. Theorems 3.2 and 4.1.

THEOREM 2.11

Suppose $(N, <)$ is well-founded, and let $R = o((R_x)_{x \in V})$. Then

1. mRn iff $\forall i \in N. (\forall j < i. mR_{v(j)}^{\equiv} n)$ implies $mR_{v(i)} n$.
2. mRn iff $\forall i \in N. (mR_{v(i)} n \text{ or } (\exists j < i. mR_{v(j)}^< n \text{ and } \forall j' < j. mR_{v(j')}^{\equiv} n))$.
3. $mR^< n$ iff mRn and $\exists i \in N. mR_{v(i)}^< n$.
4. $mR^{\equiv} n$ iff $\forall i \in N. mR_{v(i)}^{\equiv} n$.

3 Canonicity of the lexicographic rule

We have defined priority operators, which take as arguments some preference relations and combine them according to some priority, using the lexicographic rule. Arrow [1, 2] has studied operators taking sets of preference relations to preference relations, and proposed natural conditions that they should satisfy. Our aim in this section is to show that priority operators can be defined by a variant of Arrow's conditions, which is also very natural. Historically, we arrived at these conditions when looking for further preferential operators, mainly a counterpart for disjunction, only to discover that there are no further operators.

Let o be an operator taking $(R_x)_{x \in V}$ and returning $R = o((R_x)_{x \in V})$. To be natural, the operator o should:

- I. be *independent* of irrelevant alternatives: the resulting preference on elements in M depends only on the argument preferences on these elements. That is,

$$\forall M' \subseteq M, \quad o((R_x)_{x \in V})|_{M'} = o((R_x|_{M'})_{x \in V}).$$

This is condition 2 in [1] and [2];

- B. be *based* on preferences only: o is a function of the R_x 's only, and may not take into account the identity of any element of M . That is, if there is an isomorphism f between M and M' (i.e. a bijection f such that $\forall x \in V, \forall a, b \in M, aR_x b$ iff $f(a)R'_x f(b)$) then the results are the same: aRb iff $f(a)R'f(b)$. This condition is called permutation invariance in algebraic logic. It was not used by Arrow, but by algebraists, order theorists, and economists [12, p. 1448] and seems very natural;
- U. be *unanimous with abstentions*: for intuition, we use here analogies from the theory of social choice. Let us consider that each R_x represents the preference-or-indifference relation of the person called x , member of a group V of voters. To establish the preference of the group, each pair of alternatives a, b will be presented in a vote, where the members can vote on whether a is preferable to b . For a given pair, each member x has four possible votes, corresponding to the cases of Definition 2.2: vote for a ($aR_x^<b$); vote for b ($bR_x^<a$); a, b are considered incomparable ($aR_x^\#b$); or indifferent (also called equivalent) ($aR_x^\equiv b$). In this last case, we say that x *abstains* in the vote of a against b . Incomparability, on the contrary, is a strong opinion here: it means that the two alternatives cannot compete, and this vote will override decided votes of the same priority. In the first two cases, we say that x is *decided*.

If all the R_x s determine a certain vote between a and b (which could be $aR_x^<b$, $aR_x^\#b$, $bR_x^<a$, or $aR_x^\equiv b$) apart from those which abstain ($aR_x^\equiv b$), then the condition of unanimity states that R also determines the same vote between a and b . That is, for all $* \in \{<, >, \equiv, \#\}$ if $\exists V' \subseteq V$ such that $V' \neq \emptyset$ and $\forall y \in V', aR_y^*b$, and $\forall x \in V - V', aR_x^\equiv b$, then aR^*b .

Respecting unanimity is the motivation for condition 4 of [1], but after motivating this condition, [1] writes a much weaker mathematical condition;

- T. *preserve transitivity*: if all the argument preferences $(R_x)_{x \in V}$ are transitive, then the resulting preference R is also transitive. This condition is not stated in [1] but is implicitly used;
- N. be *non-dictatorial*: it does not simply return a fixed one of its arguments without regard to the others. We formulate this technically as follows: if $|V| > 1$ then there is no $z \in V$ such that $R = R_z$ for all possible values of the other R_x s. This definition comes from [2].

In the case of total pre-orders, Arrow's well-known theorem shows that the property of non-dictatoriality is incompatible with the other conditions. In our case of arbitrary relations in which we have generalized his conditions, it is easy to show an opposite result.

THEOREM 3.1

Every operator satisfying unanimity with abstentions is non-dictatorial. More generally, the result of such an operator cannot be independent of any of its arguments.

PROOF. Assume o is dictatorial in z ; thus $V \setminus \{z\}$ is not empty. Take some non-full relation S and define $R_z = F$ and $R_x = S$ for all other x . By U, $o((R_x)_{x \in V}) = S \neq R_z$. ■

Thus non-dictatorial is not only compatible with IBUT, but implied by U. There are two explanations for this inversion, depending on the version ([1] or [2]) to which we compare:

1. Unanimity with abstentions is a powerful and natural condition, for pre-orders. The proof of [2] relies strongly on linear orders, where abstentions are impossible.

2. The definition of dictatorship [2] we use is natural but restrictive: some of our operators would be dictatorial under the wider definition of [1]. Arrow (in both versions) uses a supplementary unstated condition: the preservation of totality. As shown in Theorem 4.1 below, this amounts to requiring a linear (total) priority. In this case, the relation with highest priority is a dictator in the sense of [1], but not of [2].

So, of course, there is no mathematical contradiction between Arrow's results and ours. But curiously, all informal explanations of [1] could be retained to justify the conditions of our inverse result—just draw opposite extra-mathematical generalizations.

The main result of this section shows that only lexicographic combinations of preferences satisfy conditions IBUT (or equivalently IBUTN). We may state it as follows.

THEOREM 3.2

A finitary operator satisfies conditions IBUT iff it is a priority operator.

The proof, found in Section A.3 in the Appendix, works by performing 'tests' on the operator in order to find a priority graph which denotes it.

It is not obvious that the conditions IBUT are all we should require; we could also think that a natural operator should:

1. *preserve reflexivity*: usually, one conventionally considers that preferences are reflexive. This convention should be preserved by the operator;
2. *preserve irreflexivity*: if we take the opposite convention, it should also be preserved;
3. *preserve antisymmetry*: often preferences are taken to be antisymmetric; then the result should also be;
4. *preserve well-foundedness*: the goal of preferences is to find minima, and to ensure their existence we must forbid infinite regression. It is clearly important that this property is preserved;
5. *allow majority extension or respond positively* [2]: Given a situation where the result is some vote (for instance, that a and b are indifferent), then any situation identical except that more individual preferences give that vote, should have the same resulting vote;
6. *be justified*: if the result is to prefer one of the interpretations, then at least one default (called the *justification*) must prefer this interpretation;
7. *obey Pareto rule or be benevolent*: if one criteria strictly prefers an alternative, and the other ones prefer it, it should be strictly preferred globally. $\forall xaR_xb \wedge \exists yaR_y^<b \Rightarrow aR^<b$.

Fortunately, all these conditions can be derived from the four basic ones (at least for finitary operators). The preservation properties (1–4) are theorems of the next section. Properties (5–6) are proved in Lemmas A.11 and A.9, respectively, of Appendix A.3. The Pareto rule is a special case of U. There is, however, one condition (proposed by [10]) that we cannot add, namely *decidedness*: that the global preference is decided (prefers one of the two interpretations to be compared) as soon as one of the individual preferences is decided. Intuitively, this condition seems rather strong: for instance, the operator cannot decide that two interpretations are incomparable, even if a vast majority of defaults share this opinion or if two equally important sets of defaults hold opposite opinions. If we add decidedness, no combination operator can be found, since we fall back on the conditions of the original Arrow theorem: the operator will preserve totality.

TABLE 2. Properties of a relation R and their closures

Property	Definition	'Closure(s)'
Reflexive	$\forall m \in M. mRm$	$mR^{\leq}n$ iff mRn or $m = n$
Irreflexive	$\forall m \in M. m\overline{R}m$	$mR^{\neq}n$ iff mRn and $m \neq n$
Symmetric	$\forall m, n \in M. (mRn \Rightarrow nRm)$	$mR^{\vee}n$ iff mRn or nRm $mR^{\equiv}n$ iff mRn and nRm $mR^{\#}n$ iff $m\overline{R}n$ and $n\overline{R}m$.
Antisymmetric	$\forall m, n \in M. (mRn \wedge nRm \Rightarrow m = n)$	$mR^{<}n$ iff mRn and $n\overline{R}m$
Transitive	$\forall m_1, m_2, m_3 \in M. (m_1Rm_2 \wedge m_2Rm_3 \Rightarrow m_1Rm_3)$	$mR^{+}n$ iff $\exists n. mR^n y$
Total	$\forall m, n \in M. (mRn \vee nRm)$	
Empty	$\forall m, n \in M. m\overline{R}n$	\emptyset (the empty relation)
Full	$\forall m, n \in M. mRn$	F (the full relation)
Well-founded	transitive, and there is no $R^{<}$ -sequence $\dots m_3 R^{<} m_2 R^{<} m_1$	
\downarrow Zorn	R transitive, and each chain (totally R -ordered subset) in M has a lower bound.	

4 Propagation of properties via priority operators

Grosz [14] has shown that a lexicographic combination of transitive preferences is transitive, provided the set of nodes is well-founded. A more systematic treatment of such properties is summarized in Table 3, for the classical properties described in Table 2. For example, Grosz's result is represented as line 5 of Table 3. This says that for any priority operator o and non-empty family $(R_x)_{x \in V}$ of arguments, the resultant relation $R = o((R_x)_{x \in V})$ is transitive if each of the argument relations R_x is transitive, and also the priority $<$ on N is well-founded.

Other conditions, such as reflexivity, irreflexivity and symmetry, propagate more simply, without extra conditions on the priority relation.

THEOREM 4.1

Table 3 holds; i.e. the properties are propagated by the lexicographic combination in the manner shown in the table.

In preferential logics, we are interested in finding the minimals of preference relations. A strong property guaranteeing the existence of minimals is well-foundedness. Assuming that the relation R is transitive, well-foundedness is equivalent to saying that R restricted to any non-empty subset M' of M has minimals, i.e. $\text{Min}_R(M') \neq \emptyset$. Table 3 shows that well-foundedness is propagated by the lexicographic rule under simple assumptions.

TABLE 3. How the properties propagate through priority operators

Let $(N, <, v)$ be a priority graph denoting the priority operator o .		
<i>The result of o is...</i>		
	<i>if...</i>	<i>argument is, and also...</i>
1.	reflexive	each
2.	irreflexive	some
3.	symmetric	each
4.	antisymmetric	some there is no infinite $<$ -chain below it.
5.	transitive	each the priority is well-founded.
6.	total	each the priority is total.
7.	empty	some its node is minimal in $(N, <)$.
8.	full	each
Now suppose N is finite, and each $R_{v(i)}$ is transitive.		
9.	well-founded	each
10.	\downarrow Zorn	each for each $K \subseteq N$ the relation $\bigcap_{i \in K} R_{v(i)}$ is \downarrow Zorn.

However, well-foundedness may be rather stronger than we actually need. This is because we do not require the existence of minimals in *any* non-empty set $M' \subseteq M$, but only in those sets which are denoted by a theory in the logic. This is the motivation behind the condition of stopperedness [21] (aka smoothness [16]) in the literature.

To study the propagation of stopperedness, let C be the set of subsets of M which are closed, i.e. which are the denotation of a theory. Take any $M' \in C$. We say that R has the \downarrow Zorn property (pronounced downwards-Zorn) with respect to M' if each R chain in M' has a lower bound in M' . That is the condition that is required in order to apply Zorn's lemma to find minimals in M' . Thus, to study the propagation of stopperedness it is sufficient to study the propagation of \downarrow Zorn in each of the sets in C . The propagation of \downarrow Zorn in any set is described in Table 3.

THEOREM 4.2

Well-foundedness and \downarrow Zorn are related as follows. Let R be a transitive relation on M . R is well-founded iff (for all $N \subseteq M$ $R|_N$ is \downarrow Zorn).

Line 10 of Table 3 is considerably harder to prove than the others, and requires several lemmas. The proofs are in the Appendix, Section A.4.

5 Proof rules for priority graphs

5.1 Refinement and equivalence

Checking equivalence between priority graphs by applying the lexicographic rule to convert them into priority operators is a time-consuming and error-prone process. Fortunately, there are some syntactical rules which can help us. We consider only well-founded priority graphs with finitely many variables. As well as checking equivalence, we develop proof rules for checking *refinement* between priority operators.

DEFINITION 5.1

We say that o_1 *refines* o_2 and write $o_1 \sqsubseteq o_2$ if, for all argument tuples $(R_x)_{x \in V}$, we have

$o_1((R_x)_{x \in V}) \subseteq o_2((R_x)_{x \in V})$ as relations. This notion is lifted naturally to priority graphs: $g_1 \sqsubseteq g_2$ if g_1, g_2 denote operators o_1, o_2 and $o_1 \sqsubseteq o_2$.

If $(N, <, v)$ is a priority graph and $i \in N$, we write $\downarrow i$ for the set $\{j \in N \mid j < i\}$ and $v[N']$ for $\{v(j) \mid j \in N'\}$ for any $N' \subseteq N$. Thus $v[\downarrow i] = \{v(j) \mid j < i\}$ is the set of variables occurring below the node i .

THEOREM 5.2

$g_1 \sqsubseteq g_2$ iff for each $j \in N_2$, there is a $i \in N_1$:

- $v_1(i) = v_2(j)$; and
- $v_1[\downarrow_1 i] \subseteq v_2[\downarrow_2 j]$.

COROLLARY 5.3

(Cf. Grosf [14], Theorem 3) If $N_1 = N_2$ and $v_1 = v_2$ and $<_1 \subseteq <_2$ then $g_1 \sqsubseteq g_2$.

COROLLARY 5.4

If $g_1 \sqsubseteq g_2$, then $v_2[N_2] \subseteq v_1[N_1]$.

The theorem is easily extended to a simple and effective test for equivalence between priority graphs (recall that two graphs are said to be equivalent if they denote the same operator):

COROLLARY 5.5

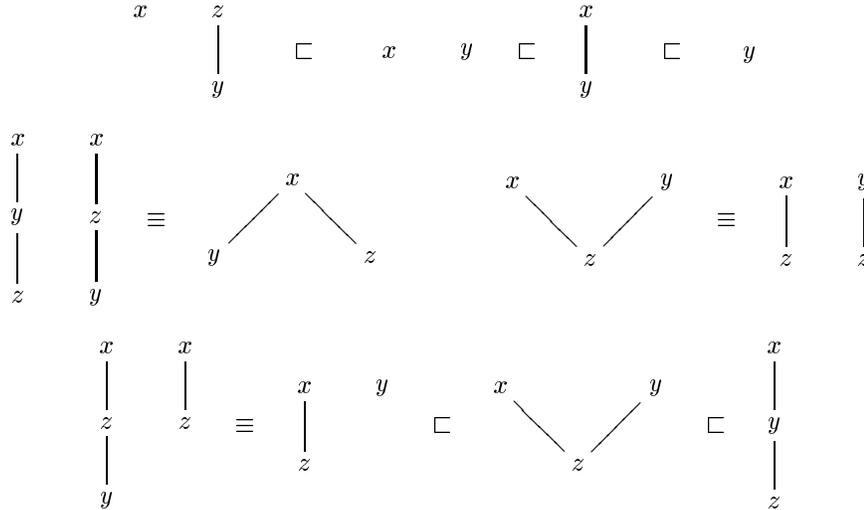
$g_1 \equiv g_2$ iff

- for each $i \in N_1$, there is a $j \in N_2$ such that $v_1(i) = v_2(j)$ and $v_2[\downarrow_2 j] \subseteq v_1[\downarrow_1 i]$, and
- for each $j \in N_2$, there is an $i \in N_1$ such that $v_1(i) = v_2(j)$ and $v_1[\downarrow_1 i] \subseteq v_2[\downarrow_2 j]$.

PROOF. Simply apply Theorem 5.2 to the refinements $g_1 \sqsubseteq g_2$ and $g_2 \sqsubseteq g_1$. ■

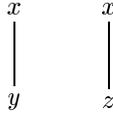
EXAMPLE 5.6

Some refinement and equivalence relationships between priority graphs, which are easily checkable using the rules expressed by these theorems:



EXAMPLE 5.7

The priority graph g_1



was presented in example 2.8, and it was stated that it could not be written with just one occurrence of the variable x . Corollary 5.5 can be used to prove this. Suppose g_2 has just a single occurrence of x , say at node $i \in N_2$, and $g_1 \equiv g_2$. Then by the first part of 5.5, $v_2[\downarrow_2 i]$ must be a subset of $\{y\}$ and of $\{z\}$, hence (since y, z are distinct variables) it must be empty. By the second part, either $\{y\} \subseteq v_2[\downarrow_2 i]$ or $\{z\} \subseteq v_2[\downarrow_2 i]$, so $v_2[\downarrow_2 i]$ cannot be empty. Contradiction.

COROLLARY 5.8

If $g_1 \equiv g_2$, then $v_1[N_1] = v_2[N_2]$.

We are interested in simplifying priority graphs without changing the operator they denote. To this end, we define the notion of a priority graph normal form; the normal form of a graph is the ‘simplest’ graph which is equivalent to it. (Here ‘simplest’ means with a minimal number of nodes, but surprisingly, with a maximal number of links.)

DEFINITION 5.9

Let $g = (N, <, v)$. A node $i \in N$ is *critical* if for all $k \in N$ with $v(i) = v(k)$, we have $v[\downarrow k] \not\subseteq v[\downarrow i]$.

That is to say, a node i is critical if the set of variables beneath it ($v[\downarrow i]$) is minimal compared with other nodes k labelled by the same variable. The importance of critical nodes can be seen in Definition 2.5: the $\forall i$ need only range over critical nodes, because if i is not critical then the existence of an appropriate j beneath it is guaranteed by its existence for a critical node.

DEFINITION 5.10

The *normal form* of a priority graph $g = (N, <, v)$ is the graph $(N', <', v')$ where

$$\begin{aligned} N' &= \{(v(i), v[\downarrow i]) \mid i \text{ critical in } g\} \\ (v(j), v[\downarrow j]) <' (v(i), v[\downarrow i]) &\Leftrightarrow v[\downarrow j] \cup \{v(j)\} \subseteq v[\downarrow i] \\ v'((v(i), v[\downarrow i])) &= v(i). \end{aligned}$$

(We will soon justify the term ‘normal form’ by giving rewrite rules for priority graphs.)

THEOREM 5.11

1. Any priority graph is equivalent to its normal form.
2. Two priority graphs are equivalent iff their normal form is the same.

COROLLARY 5.12

The normal form operator is idempotent.

We now give rewrite rules for transforming a finite graph into its normal form, up to renaming of the nodes.

DEFINITION 5.13

The rewrite rules for priority graphs are

(link) Link j below i if this does not change the down-set of i .

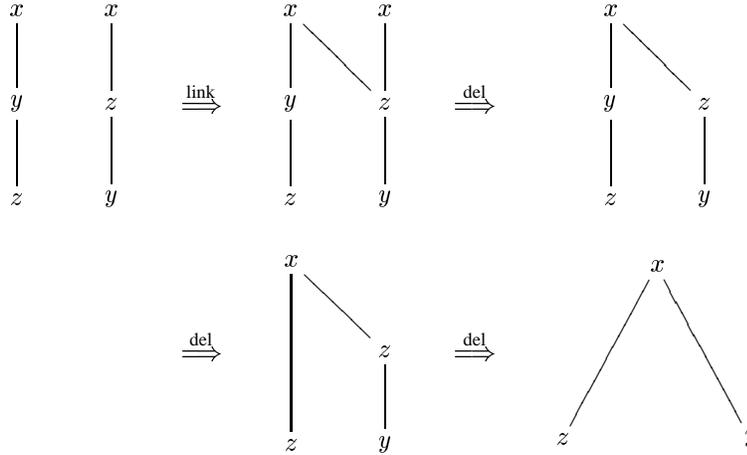
More formally: $g \xrightarrow{\text{link}} g'$ if: there are $i, j \in N$ with $i \not\leq j$, $v[\downarrow j] \cup \{v(j)\} \subseteq v[\downarrow i] \cup \{v(i)\}$, and $<'$ is the transitive closure of $< \cup \{(j, i)\}$.

(del) Delete a node if:

- it is not critical or there is an equivalent node, and
- deleting it does not change the down-sets of other nodes. Note that this last condition will eventually be obtained by application of (link), so that only one copy of each critical node will be kept.

More formally: $g \xrightarrow{\text{del}} g'$ if: there are distinct $i, j \in N$ with $v[\downarrow j] \subseteq v[\downarrow i] \cup \{v(i)\}$ and $v(i) = v(j) = x$ for some x , and for all $i' > i$ there exists $i'' < i'$ with $v(i'') = x$, and $N' = N - \{i\}$, and $<' = <|_{N'}$ (the restriction of $<$ to N'), and $v' = v|_{N'}$.

EXAMPLE 5.14



THEOREM 5.15

By applying rules (link) and (del) repeatedly in any order until none applies, any finite priority graph is brought into a form which is equal to its normal form, up to renaming of elements of N .

COROLLARY 5.16

Any priority graph in which each variable occurs at most once is in normal form.

Of course, there are priority graphs with several occurrences of a variable which are in normal form, such as the one corresponding to the term $(x/y) \parallel (x/z)$ (Example 5.7).

5.2 Preferential entailment and preferential equivalence

In the setting of preferential logics, the models of interest are the minimal models according to the preference (sometimes called *preferred models*).

$$\text{Min}(R) = \{m \in M \mid \nexists n \in M. nR < m\}.$$

Let us define the relation of *preferential entailment* between operators as inclusion of preferred models.

DEFINITION 5.17

o_1 *preferentially entails* o_2 , written $o_1 \sim o_2$ iff for any arguments $(R_x)_{x \in V}$, we have $\text{Min}(o_1((R_x)_{x \in V})) \subseteq \text{Min}(o_2((R_x)_{x \in V}))$. As for refinement, this notion naturally extends to priority graphs.

Note that preferential entailment (\sim) is distinct from refinement (\sqsubseteq). Analogously to refinement, however, we can check preferential entailment by means of a simple syntactic characterization on graphs denoting the operators.

THEOREM 5.18

$g_1 \sim g_2$ iff $v_2[N_2] \subseteq v_1[N_1]$ and for each node $i \in N_1$ either $v[N_2] \subseteq v[\downarrow_1 i]$, or there is a $j \in N_2$ such that $v(i) = v(j)$ and $v[\downarrow j] \subseteq v[\downarrow i]$.

COROLLARY 5.19

If $g_1 \sim g_2$, then $v_2[N_2] \subseteq v_1[N_1]$

DEFINITION 5.20

o_1, o_2 are *preferentially equivalent* if $o_1 \sim o_2$ and $o_2 \sim o_1$. Again, this extends naturally to graphs.

Although preferential entailment and refinement are distinct, it turns out rather surprisingly that preferential equivalence and equivalence are the same.

PROPOSITION 5.21

Two priority graphs are preferentially equivalent iff they are equivalent.

PROOF. \Rightarrow . Suppose without loss of generality that the graphs are in normal form. It is impossible that $v_1[\downarrow_1 i] \supseteq v[N_2]$ ($= v[N_1]$ by Corollary 5.19) because i wouldn't be critical. So we have the other case, which is just the characterization of inclusion (Theorem 5.2) in each direction, yielding equivalence. \Leftarrow . Obvious. ■

So the computation of the normal form can also be used for preferential equivalence. When constants for given relations are introduced, this property may fail.

The results of this section are directly operational, and yield algorithms for deciding equality, refinement, preferential entailment, preferential equivalence and computation of the normal form.

6 Composing priority graphs

6.1 Composition vs graphical insertion

Since an operator o maps some preferences $(R_x)_{x \in V}$ to a preference $o((R_x)_{x \in V})$, operators can be composed with each other to give further operators. Therefore, priority operators can be composed, but are their compositions also priority operators? In certain circumstances the answer is yes; indeed, we can compose priority operators simply by manipulations on the graphs that denote them.

DEFINITION 6.1

Let $g = (N, <, v)$ having variables $V = v[N]$, and for each $x \in V$ let $g_x = (N_x, <_x, v_x)$ be a priority graph. The graphical insertion $g' = g[(g_x)_{x \in V}]$ of the priority graphs g_x in the priority graph g is $(N', <', v')$ where

- $N' = \{(i, j) \mid i \in N, j \in N_{v(i)}\}$,

30 Operators and Laws for Combining Preference Relations

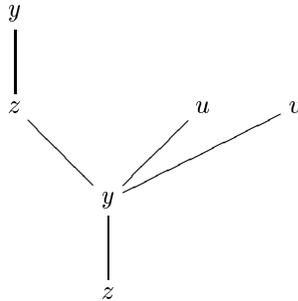
- $(i_1, j_1) <' (i_2, j_2)$ iff $(i_1 < i_2)$ or $(i_1 = i_2$ and $j_1 <_{v(i)} j_2)$,
- $v((i, j)) = v_{v(i)}(j)$.

EXAMPLE 6.2

If g, g_1, g_2 are respectively the priority graphs



then $g' = g[g_1, g_2]$ is the priority graph



For well-founded priority operators, graphical insertion is the syntactical counterpart of semantical composition of priority operators.

THEOREM 6.3

Let g be a well-founded graph denoting operator o with variables V . Let $(g_x)_{x \in V}$ be a family of well-founded graphs denoting operators $(o_x)_{x \in V}$ with variables $(V_x)_{x \in V}$. Let g' be the graphical insertion of $(g_x)_{x \in V}$ in g , and let o' be the operator denoted by g' .

Then o' is the composition of o with $(o_x)_{x \in V}$, i.e.

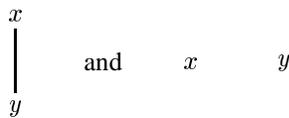
$$o' \left((R_y)_{y \in \bigcup \{V_x \mid x \in V\}} \right) = o \left((o_x((R_y)_{y \in V_x}))_{x \in V} \right).$$

COROLLARY 6.4

Well-founded priority operators are closed under composition.

6.2 The binary priority operators

There are essentially only two binary priority operators; they are denoted by the graphs



Strictly speaking, there is also a third one, which is like the first one but with x and y swapped around. All other binary priority graphs (i.e. graphs having possibly more than two nodes but

precisely two variables) are equivalent to one of these three. Since the third one is essentially the same as the first, we focus just on the first two.

The two binary priority operators are of great importance for the remainder of the paper. We will write them respectively as x/y and $x||y$, and call $/$ ‘but’ and $||$ ‘on the other hand’. The reason for these names is the following. From the point of view of default reasoning, the ‘but’ operator combines two defaults by putting the second in a position of greater priority than the first. Thus, x/y means ‘apply the criteria x and y , and where they conflict we apply y ’. This is like the natural language connective ‘but’. The operator ‘ $||$ ’ combines two defaults by putting them at incomparable priority. The expression ‘on the other hand’ does the same job in natural language.

Applying the lexicographic rule, we can see that

PROPOSITION 6.5

1. $x/y = (x \cup y^<) \cap y$, which is also equal to $(x \cap y) \cup y^<$.
2. $x || y = x \cap y$.

PROOF. Immediate from the definitions. ■

The importance of these two operators is that any finitary priority operator can be written in terms of these two, using graphical insertion, as we now explain.

The operators $/, ||$ apply to other operators in the standard compositional way: o_1/o_2 and $o_1||o_2$ are defined by $(o_1/o_2)((R_x)_{x \in V}) = o_1((R_x)_{x \in V})/o_2((R_x)_{x \in V})$, and $(o_1||o_2)((R_x)_{x \in V}) = o_1((R_x)_{x \in V})||o_2((R_x)_{x \in V})$. According to Theorem 6.3, the operators $/$ and $||$ can equivalently be applied at the level of priority graphs, in which case they correspond respectively to the graphical operations of *linear sum* and *disjoint union* [6].

THEOREM 6.6

Any finitary priority operator is denoted by a term built from $/, ||$ and the variables that occur in the priority graph for the operator.

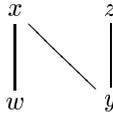
EXAMPLE 6.7

The 12 priority graphs in Example 5.6 are respectively equivalent to the following terms: $x||z/y, x||y, x/y, y, (x/y/z)||x/z/y, x/(y||z), (x||y)/z, (x/z)||y/z, (x/z/y)||x/z, (x/z)||y, (x||y)/z$, and $x/y/z$.

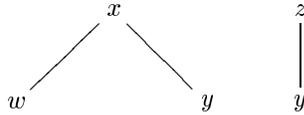
Notice how the $/, ||$ term can be obtained from the shape of the priority graph. When two equivalent priority graphs are given, we obtained the term using the second one. Extracting the term from the first graph in the first example, we obtain $(x/y/z)||x/z/y$, which can be shown to be equal to $x/(y||z)$.

EXAMPLE 6.8

We cannot graphically obtain a term from the ‘N’ shaped graph



However, it is equivalent to



and so it denotes the operator $(x/(w||y))||(z/y)$.

COROLLARY 6.9

Any finitary operator satisfying conditions IBUT is equivalent to a term built from $/, ||$ and the variables.

PROOF. Follows from Theorems 3.1 and 6.6. ■

The notions of refinement, equivalence, preferential entailment and preferential equivalence of the last section all extend naturally to terms.

EXAMPLE 6.10

$(x||y)/z \equiv (x/z)||y/z$; however, $(x/y)||x/z \sqsubseteq x/(y||z)$ but not conversely.

EXAMPLE 6.11

$x/y \sim y, x/y \sim x || y, x/y/z \sim y || z$.

We note in passing that, for any relation R (and where F is the full relation $M \times M$ and \emptyset the empty relation):

$$\begin{array}{ll} R/F & = R \\ F/R & = R \\ R/\emptyset & = \emptyset \\ \emptyset/R & = R^< \end{array} \qquad \begin{array}{ll} R||F & = F||R = R \\ R||\emptyset & = \emptyset||R = \emptyset. \end{array}$$

7 Algebraic treatment

Now that we have terms for describing priority operators, we can study their algebraic properties. Consider a set of relations on M which is closed under the binary operators $/$ and $||$, defined as before by

$$\begin{aligned} x/y &= (x \cap y) \cup y^< \\ x || y &= x \cap y. \end{aligned}$$

We call such an algebra a *preferential algebra*, or *PA*. Preferential algebras are a special case of algebras of binary relations, a survey on which can be found in Némethi [22] and Schein [29].

Terms in the language of PAs are made from variables and the binary operators $/, ||$. If V is the set of variables occurring in a term τ , then τ denotes the V -ary priority operator which evaluates the term after substituting its arguments in place of the variables. The following theorem rephrases Theorem 6.6 in algebraic terminology.

THEOREM 7.1

For any finitary V -ary priority operator o there is a term τ of the language of preferential algebras such that for any preferential algebra A and relations $(R_x)_{x \in V}$ in A we have that $o((R_x)_{x \in V}) = \tau((R_x)_{x \in V})$.

As usual with relational algebras, we may identify certain equalities which hold between terms, however their variables are substituted. For example, it was seen in Example 6.10 that $(x||y)/z = (x/z)||y/z$.

The following theorem gives a finite axiomatization of all the equations (equalities between terms) true in preferential algebras.

THEOREM 7.2

An equation is true in all preferential algebras iff it is derivable from the following seven axioms:

1. $x||x = x$ ($||$ Idempotent)
2. $x||(y||z) = (x||y)||z$ ($||$ Associative)
3. $x||y = y||x$ ($||$ Commutative)
4. $(x/x) = x$ ($/$ Idempotent)
5. $x/(y/z) = (x/y)/z$ ($/$ Associative)
6. $(x||y)/z = (x/z)||y/z$ ($/$ Distributes over $||$)
7. $(x/y)||x = x||y$ (Absorption)

Some subsets of these axioms are interesting on their own:

- Two terms yield the same priority graph by graphical insertion iff they can be proved equal by the axioms 2, 3, 5.
- We can define the *forest form* of a term, as the term obtained by normalizing it using the axiom 6 from left to right.
- The rules 1, 2, 3 form a complete axiomatization of the $||$ -reduct (a trivial class of algebras, isomorphic to sets with intersection).
- In contrast, the rules 4, 5 do not axiomatize the $/$ -reduct: we have to add $x/y/x = y/x$ (Example 7.3(3) below). This subclass is again rather trivial, since the free algebras are isomorphic to strings of variables without repetition.

EXAMPLE 7.3

Some interesting derived equations.

1.
$$\begin{aligned} (x/y)||y &= ((x/y)/y)||x/y && \text{absorption} \\ &= (x/y)||x/y && / \text{ associative, idempotent} \\ &= x/y && || \text{ idempotent} \end{aligned}$$
2.
$$\begin{aligned} x/(y||x) &= (x/(y||x))||y||x && (1) \\ &= ((x/(y||x))||x)||y && || \text{ associative, commutative} \\ &= ((y||x)||x)||y && \text{absorption} \\ &= x||y && || \text{ idempotent} \end{aligned}$$
3.
$$\begin{aligned} x/y/x &= (x/y/x)||y/x && (1) \text{ where } y = y/x \\ &= (x/y/x)||y/x/y/x && / \text{ idempotent} \\ &= y/x/y/x && (1) \text{ where } y = x/y/x \\ &= y/x && / \text{ idempotent} \end{aligned}$$
4.
$$\begin{aligned} (z/(x||y))||y &= [(z/(x||y))||x||y]||y && (1) \text{ where } y = x||y \\ &= [(z/(x||y))||x||y] && || \text{ associative, idempotent} \\ &= (z/(x||y)) && (1) \end{aligned}$$
5.
$$\begin{aligned} y/((x/y)||z) &= (y/((x/y)||z))||x/y && (4) \\ &= (y/((x/y)||z))||x/y||y && (2) \\ &= (x/y)||z||y||x/y && \text{absorption} \\ &= (x/y)||z && (1) \end{aligned}$$

$$\begin{aligned}
6. \quad x / ((x/y) \parallel z) &= x/y / ((x/y) \parallel z) & (5) \\
&= (x/y) \parallel z & (3)
\end{aligned}$$

$$\begin{aligned}
7. \quad y \parallel z \parallel (y/z) &= y \parallel z \parallel y & \text{absorption} \\
&= y \parallel z & \text{idempotence}
\end{aligned}$$

These axioms are also complete for inclusion, since $R_1 \subseteq R_2$ iff $(R_1 \parallel R_2) = R_2$. It is also possible to construct a (uninteresting) proof system for inclusion without resorting to equality.

Preferential algebras have turned out to be an interesting case of relational algebras. We gave in theorem 7.2 a finite set of axioms from which all equations true of PAs may be proved. There are many other issues in relational algebra which can be discussed. For example, is PA axiomatizable in the following stronger sense: is there a finite set of equations which are true of all and *only* all algebras in PA? If so, PA is a *variety*. The answer is no; this is proved in the appendix. However, PA is a quasi-variety (also proved in the appendix), which means that it can be axiomatized (in this strong sense) by conditional equations.

The following theorem gives a derivation system for preferential entailments true in preferential algebras.

THEOREM 7.4

A preferential entailment $\tau \sim \sigma$ holds in all preferential algebras iff it is derivable from the equality axioms 1–7, together with the following:

8. If $x \sim y$ then $z/x \sim y$ (C1)
9. If $y/x = x$ and $x \parallel y = y$ then $x \sim y$ (S1)

8 Conclusion

The paper develops the theory of generalized prioritization begun by Grosz [14]. It introduces priority operators, an analog of circumscription policies applicable in preferential logics. Furthermore:

- It shows that priority operators are canonical with respect to a generalization of Arrow's conditions;
- It gives criteria for deciding: refinement, equality and preferential entailment of priority operators;
- It shows that the two binary operators can express any priority operator, and hence any operator satisfying generalized Arrow's conditions;
- It gives a complete axiomatization of the operators and their relationships.

Topics for further study include investigating the supplementary laws that can be established for specific preferential logics, and for their combinations. We would also like to relax the requirement that operators be finitary, and study a logic for expressing infinitary operators.

9 Acknowledgements

Hajnal Andr eka acknowledges support from the Hungarian National Foundation for Scientific Research (grant nos. OTKA T30314 and T23234). Mark Ryan and Pierre-Yves Schobbens acknowledge support from the European Union through Esprit WGs ModelAge, ASPIRE and

FIREworks. Mark Ryan acknowledges the Nuffield Foundation, and British Telecom. Pierre-Yves Schobbens thanks the University of Birmingham for funding and invited professorship that allowed us to finalize this article.

References

- [1] K. J. Arrow. A difficulty in the concept of social welfare. *Journal of Political Economy*, **58**, 328–346, 1950.
- [2] K. J. Arrow and H. Raynaud. *Social Choice and Multicriterion Decision-Making*, MIT Press, 1986.
- [3] L. Blume, A. Brandenburger and E. Dekel. Lexicographic probabilities and choice under uncertainty. *Econometrica*, **59**, 61–79, 1991.
- [4] S. Brass and U. Lipeck. Bottom-up query evaluation with partially ordered defaults. In *Proceedings of the 3rd International Conference on Deductive and Object-Oriented Databases*, Lecture Notes in Computer Science. Springer Verlag, 1993.
- [5] G. Brewka. *Non-monotonic Reasoning: Logical Foundations of Commonsense*. Cambridge Tracts in Computer Science. Cambridge University Press, 1991.
- [6] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge Mathematical Textbooks. Cambridge University Press, 1990.
- [7] M. de Condorcet. *Essai sur l'application de l'analyse la probabilité des décisions rendues la pluralité des voix*. 1785.
- [8] N. Dershowitz and J. P. Jouannaud. Rewrite systems. In *Handbook of Theoretical Computer Science (Vol. B: Formal Models and Semantics)*, J. van Leeuwen, ed. North-Holland, Amsterdam, 1990.
- [9] F. M. Dionísio. *Composition of Hierarchic Default Specifications*. PhD thesis, University of Hannover, Germany, 1997. Supervised by U. Lipeck.
- [10] J. Doyle and M. P. Wellman. Impediments to universal preference-based default theories. *Artificial Intelligence*, **49**, 97–128, 1991.
- [11] P.C. Fishburn. A study of lexicographic expected utility. *Management Science*, **17**, 672–678, 1971.
- [12] P.C. Fishburn. Lexicographic orders, utilities, and decision rules: a survey. *Management Science*, **20**, 1442–1471, 1974.
- [13] M. L. Ginsberg, ed. *Readings in Nonmonotonic Reasoning*. Morgan Kaufmann, Los Altos, CA, 1987.
- [14] B. N. Grosz. Generalising prioritization. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, J. Allen, R. Fikes, and E. Sandewall, eds. pp. 289–300. Morgan Kaufmann, 1991.
- [15] A. Hunter and B. Nuseibeh. Managing inconsistent specifications: Reasoning, analysis and action. *ACM Transactions on Software Engineering and Methodology*, **7**, 335–367, 1998.
- [16] S. Kraus, D. Lehmann, and M. Magidor. Non-monotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence*, **44**, 167–207, 1990.
- [17] I. H. LaValle and P. C. Fishburn. Lexicographic state-dependent subjective expected utility. *Journal of Risk and Uncertainty*, **4**, 251–269, 1991.
- [18] I. H. LaValle and P. C. Fishburn. State-independent subjective expected lexicographic utility. *Journal of Risk and Uncertainty*, **5**, 217–240, 1992.
- [19] V. Lifschitz. Some results on circumscription. Technical Report 1019, Stanford University, Dept of Computer Science, 1984.
- [20] V. Lifschitz. Circumscription. In *Handbook of Logic in Artificial Intelligence*. D. Gabbay, C. Hogger, and J. Robinson, eds. Oxford University Press, 1992.
- [21] D. Makinson. General patterns in non-monotonic reasoning. In *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3. D. Gabbay, C. Hogger, and J. Robinson, eds. Oxford University Press, 1994.
- [22] I. Németi. Algebraizations for quantifier logics: an introductory overview. *Studia Logica*, **50**, 485–569, 1991. An extended and updated version with proofs will appear in *Proceedings of Summer School on Algebraic Logic (Budapest 1994)*.
- [23] M. Pirlot and Ph. Vincke. Lexicographic aggregation of semiorders. *Journal of Multi-Criteria Decision Analysis*, **1**, 47–58, 1992.
- [24] V. Pratt. The pomset model of parallel processes : Unifying the temporal and the spatial. Technical Report STAN-CS-85-1049, Stanford University, Jan 1985.

- [25] M. D. Ryan. Defaults and revision in structured theories. In *Proceedings of the Sixth IEEE Symposium on Logic in Computer Science (LICS)*, pp. 362–373, 1991.
- [26] M. D. Ryan. Defaults in specifications. In *Proceedings of the IEEE Conf. on Requirements Engineering (RE'93)*, A. Finkelstein, ed. pp. 142–149, 1993.
- [27] M. D. Ryan. Towards specifying norms. *Annals of Mathematics and Artificial Intelligence*, **9**, 49–67, 1993.
- [28] M. D. Ryan. Belief revision and ordered theory presentations. In *Logic, Action and Information*, A. Fuhrmann and H. Rott, eds. De Gruyter Publishers, 1994. Also in *Proceedings of the Eighth Amsterdam Colloquium on Logic*, University of Amsterdam, 1991.
- [29] B. M. Schein. Representation of subreducts of Tarski relation algebras. In *Algebraic Logic*, H. Andréka, J. D. Monk, and I. Németi, eds. pp. 621–636. Volume 54 of *Colloquia Mathematica Societatis János Bolyai*, North-Holland, 1991.
- [30] P.-Y. Schobbens. Exceptions for algebraic specifications: on the meaning of ‘but’. *Science of Computer Programming*, **20**, 73–111, 1993.

Appendix

A Mathematical details

A.1 Introduction

This Appendix covers many mathematical details (including proofs of theorems stated in the text). Its structure mirrors the structure of the main part of the paper. New definitions and lemmas are given new numbers, but theorems which are stated in the text and proved here retain their old numbers.

A.2 Priority operators

Let $g = (N, <, v)$ be a priority graph denoting the operator o .

THEOREM 2.11

Suppose $(N, <)$ is well-founded, and let $R = o((R_x)_{x \in V})$. Then

1. mRn iff $\forall i \in N. (\forall j < i. mR_{v(j)}^{\equiv} n)$ implies $mR_{v(i)} n$.
2. mRn iff $\forall i \in N. (mR_{v(i)} n$ or $(\exists j < i. mR_{v(j)}^< n$ and $\forall j' < j. mR_{v(j')}^{\equiv} n))$.
3. $mR^< n$ iff mRn and $\exists i \in N. mR_{v(i)}^< n$.
4. $mR^{\equiv} n$ iff $\forall i \in N. mR_{v(i)}^{\equiv} n$.

PROOF. 1. (\Rightarrow) Suppose i is such that $\forall j < i. mR_{v(j)}^{\equiv} n$. We require to show that $mR_{v(i)} n$. Suppose not; then $\exists j < i. mR_{v(j)}^< n$, a contradiction.

(\Leftarrow) Suppose i is such that $mR_{v(i)} n$. We require to find $j < i$ such that $mR_{v(j)}^< n$. By hypothesis, $\exists j_1 < i. mR_{v(j_1)}^{\equiv} n$ or $nR_{v(j_1)} m$. If $mR_{v(j_1)} n$, then $nR_{v(j_1)} m$ so $mR_{v(j_1)}^< n$, so we set $j = j_1$. Otherwise, again using the hypothesis, $\exists j_2 < j_1. mR_{v(j_2)}^{\equiv} n$ or $nR_{v(j_2)} m$. Again, we set $j = j_2$ or we find j_3 with the same property. This procedure must terminate, for otherwise we have an infinite descending sequence $j_1 > j_2 > \dots$, contradicting the well-foundedness of $(N, <)$.

2. (\Leftarrow) immediate. (\Rightarrow) Similarly to part 1, find j minimal with $mR_{v(j)}^< n$.

3. (\Rightarrow) Suppose $m o((R_x)_{x \in V})^< n$. Then $m o((R_x)_{x \in V}) n$ is immediate. Also, $m o((R_x)_{x \in V})^< n$ implies $n o(\overline{(R_x)_{x \in V}}) m$, so $\exists i. nR_{v(i)} m$. Since $m o((R_x)_{x \in V}) n$, either $mR_{v(i)} n$, in which case $mR_i^< n$ as required; or $\exists j < i. mR_{v(j)}^< n$, also proving the result.

(\Leftarrow) Let i be minimal in the set $\{i \mid mR_{v(i)}^< n\}$. Then $nR_{v(i)} m$ and $\forall j < i. nR_{v(j)}^{\equiv} m$, so $n o(\overline{(R_x)_{x \in V}}) m$

4. Similar ideas. ■

A.3 Canonicity of the lexicographic rule

Our aim in this section is to prove Theorem 3.2. This will involve inventing a new view of priority operators in terms of what we call *votes*. We do this in a sequence of lemmas. The first one shows that an operator that is independent, unanimous, and based on preferences (in short: IBU) is determined by its responses to all possible relations on a fixed two-point domain.

LEMMA A.1

Let $M_2 = \{m, n\} \subseteq M, m \neq n$, and o_1, o_2 be two IBU operators. If for all families of relations $(R_x)_{x \in V}$ we have $o_1((R_x)_{x \in V})|_{\{m, n\}} = o_2((R_x)_{x \in V})|_{\{m, n\}}$ then, for all $(R_x)_{x \in V}, o_1((R_x)_{x \in V}) = o_2((R_x)_{x \in V})$.

PROOF. Take any $c, d \in M$. We show $c o_1((R_x)_{x \in V}) d$ iff $c o_2((R_x)_{x \in V}) d$.

- If $c = d$, we have $c R_x \equiv d$ or $c R_x \# d$ for all x . Then by U, we have either $c o_1((R_x)_{x \in V}) \# d$ and $c o_2((R_x)_{x \in V}) \# d$, or $c o_1((R_x)_{x \in V}) \equiv d$ and $c o_2((R_x)_{x \in V}) \equiv d$, depending on whether $c R_x \# d$ for some x or not. In any case, o_1, o_2 agree at c, d .
- If $c \neq d$: define the family $(R'_x)_{x \in V}$ in terms of $(R_x)_{x \in V}$ as follows: $R'_x = R_x$ except at (m, n) , where $m R'_x n \Leftrightarrow c R_x d$. Then

$$\begin{aligned}
 c o_1((R_x)_{x \in V})|_{\{c, d\}} d &\Leftrightarrow c o_1((R_x|_{\{c, d\}})_{x \in V}) d && \text{by I} \\
 &\Leftrightarrow m o_1((R'_x|_{\{m, n\}})_{x \in V}) n && \text{by B} \\
 &\Leftrightarrow m o_2((R'_x|_{\{m, n\}})_{x \in V}) n && \text{by hypothesis} \\
 &\Leftrightarrow c o_2((R_x|_{\{c, d\}})_{x \in V}) d && \text{by B} \\
 &\Leftrightarrow c o_2((R_x)_{x \in V})|_{\{c, d\}} d && \text{by I.}
 \end{aligned}$$

■

DEFINITION A.2

A *vote* is an element of $\mathcal{V} = \{\#, <, >, \equiv\}$.

DEFINITION A.3

A vector of $|V|$ votes, one per variable of V , is called an *entry*.

Lemma A.1 tells us that a V -ary IBU operator o determines a unique function $\mathcal{V}^{|V|} \rightarrow \mathcal{V}$, and conversely. The function takes as argument the vote each R_x gives on the two-point domain M_2 (i.e. an entry), and returns as result the vote that $o((R_x)_{x \in V})$ gives on M_2 . Such functions can be represented finitely by an *operator table*. For instance, the operator ‘but’ defined in Section 6.2 is described by Table 4:

TABLE 4. Table of ‘but’ (\wedge)

R_1	#	#	#	#	<	<	<	<	>	>	>	>	\equiv	\equiv	\equiv	\equiv
R_2	#	<	>	\equiv	#	<	>	\equiv	#	<	>	\equiv	#	<	>	\equiv
R	#	<	>	#	#	<	>	<	#	<	>	>	#	<	>	\equiv

Each column above the line is an entry, and the element in the same column below the line is the corresponding result. For an entry e and vote v , e^v is the subset of variables that gives vote v . In particular, The *winner* e^r of an entry e is the subset of V that gives the same vote as the result r ; the *abstainers* e^\equiv is the subset of V that abstains, i.e., votes \equiv ; the rest is called the *opposition*, which is divided in two subgroups, since four votes are possible. A vote is *decided* if it is $<$ or $>$.

DEFINITION A.4

The *converse* of a vote is defined by the table:

v	v^{-1}
#	#
<	>
>	<
\equiv	\equiv

38 Operators and Laws for Combining Preference Relations

LEMMA A.5

If an IBU operator gives a result r for entry $e = (e_i)_{i \in V}$ then it gives r^{-1} for entry e^{-1} .

PROOF. By B. ■

Note that any table with this property will give us an IB operator.

COROLLARY A.6

There are $2^{4^n - 3 \cdot (2^n - 1) - 1}$ n -ary IBU operators.

PROOF. The possible tables are 4^{4^n} . Symmetry (Lemma A.5) reduces this to 2^{4^n} , which is thus the number of IB operators. The cases eliminated by unanimity, are given by choosing a non-empty unanimous subset (there are $2^n - 1$), choosing its vote (three possibilities: either $<$, $>$ or $\#$), setting the rest to \equiv . Plus one for the case where all votes are \equiv and the result is \equiv . ■

We will illustrate proofs of the next few lemmas in tabular form, which should be understood as a schematic excerpt from an operator table such as Table 4. The leftmost column indicates subsets of the variables V . Each column will represent a possible combination of votes (an entry) and the result computed by the operator. New columns can be deduced from preceding columns, according to the following rules of inference, derived from the respective conditions on the operators.

- S. Symmetry: from an entry of the table with a given result, we deduce the converse entry with the converse result (Lemma A.5). In our tabular proofs, we will omit the entry on which it is applied when it operates on the previous column of the proof table.
- U. Unanimity: any unanimous column must have the result of the unanimous subset (unless it is empty). This rule operates on the current column.
- T. Transitivity: in Table 5, we compute the admissible compositions of votes for transitivity. The vertical dimension

TABLE 5. Table of compatible compositions

o	#	<	>	\equiv
#	\mathcal{V}	{<, #}	{>, #}	{#}
<	{<, #}	{<}	\mathcal{V}	{<}
>	{>, #}	\mathcal{V}	{>}	{>}
\equiv	{#}	{<}	{>}	{ \equiv }

indicates the relation between x and y , the horizontal dimension the relation between y and z . The corresponding cell shows the implied relation between x and z . For instance, the first cell states that if $xR\#y$ and $yR\#z$, then no restriction on xRz can be deduced. The cell diagonally below states that if $xR<y$ and $yR<z$, then $xR<z$. If two columns are known, and we build a third entry which is compatible for transitivity with these two columns, then the result of this entry must also be compatible for transitivity with the results of the two known columns. For otherwise we would have built a counterexample to preservation of transitivity, by using a domain $\{x, y, z\}$ where preferences between (x, y) are given by the first column, between (y, z) by the second, and between (x, z) by the third. For instance, if we compose two entries with results $<$, $\#$ respectively, we see in the table that the result of the composition must be $<$ or $\#$ for any entry which is compatible with the first two. If x is the only variable and the vote of R_x was $<$ in the first entry and $>$ in the second entry, then any value of R_x must yield $<$ or $\#$. During a proof we will usually try to constrain the result while letting the entry vary as widely as possible to get stronger results. By default, T uses the two previous columns of the proof table.

These table excerpts will be schematic: usually, the designation on the left will not be single variables, but sets of variables, indicating that the line has to be replicated as many times as they are variables in the set (sometimes 0). Also, the content of the cell can be a set. We will sometimes omit the set braces, for compactness. In the result, the comma (e.g. in $<$, $\#$) thus means ‘or’. We convene that e_1 is the name of the first entry (the second column), and e_i is the name of the i th entry (the $i + 1$ th column). The justification will be indicated below each entry. It will be one of the basic rules (S,U,T) or the number of a lemma. Further examples are provided in the proofs below.

For the rest of this section, we will omit the reference to the (fixed) IBUT operator. For instance, whenever we speak of ‘the result of an entry’, it means the result of applying the currently considered IBUT operator.

LEMMA A.7

The result of e is \equiv iff all arguments are \equiv .

PROOF. 'If': by U.

'Only if':

	e_1	e_2	e_3	e_4
$e \equiv$	\equiv	\equiv	\equiv	\equiv
$e <$	$<$	$>$	$<$	$<$
$e >$	$>$	$<$	$<$	$<$
$e \#$	$\#$	$\#$	$<$	$<$
r	\equiv	\equiv	\equiv	$<$
<i>by</i>		S	$T \times$	$U \times$

Read this table as follows. Suppose we supply a certain entry, e_1 , which of course is divided in $\equiv, <, >, \#$ votes. The result (by hypothesis) is \equiv . Construct the converse entry $e_2 = e_1^{-1}$; by S , the result is also \equiv . Now consider the argument votes e_3 of the 4th column. Since they are compatible for transitivity with e_1, e_2 , the result r_3 should also be compatible (justification: T). But that means it must be \equiv . Now consider the argument votes of the last column, e_4 ; by U, the result should be $<$. The last two columns contradict, as indicated by \times , unless the subsets $e <, e >, e \#$ of V are all empty, so that U cannot be applied on e_4 .

Hence the only way of making the result \equiv is by having $e <, e >, e \#$ empty, i.e. all votes for \equiv . ■

The sequence of lemmas that follows proves that IBUT operators have many of the properties of priority operators. For example, the next lemma says that if a definite result is obtained from a given entry, then the same result will be obtained *a fortiori* if some abstainers join the winners, whatever the opposition does.

LEMMA A.8

If an entry e yields $<$, then any entry with some arguments in $e \#, e >$ replaced by any vote, and/or some in $e \equiv$ replaced by $<$, will also yield $<$.

PROOF. Let C be the names of the votes changing from \equiv to $<$, and let v, w be any tuple of votes.

$e <$	$<$	$<$	$<$	$<$
$e >$	$>$	$<$	v	v
$e \#$	$\#$	$<$	$\#$	w
$e \equiv \cap C$	\equiv	$<$	$<$	$<$
$e \equiv \setminus C$	\equiv	\equiv	\equiv	\equiv
r	$<$	$<$	$<$	$<$
<i>by</i>		U	T	$T(e_1, e_3)$

LEMMA A.9

If the result of e is $<$, then some argument must be $<$.

PROOF. Assume $e <$ empty. Then:

$e >$	$>$	\equiv	\equiv
$e \#$	$\#$	\equiv	\equiv
$e \equiv$	\equiv	\equiv	\equiv
r	$<$	$<$	\equiv
<i>by</i>		$A.8 \times$	$U \times$

The next lemma is very similar to Lemma A.8: It says that if an incomparability result is obtained from a given entry, then the same result will be obtained *a fortiori* if some abstainers or opposition join the winners. But here, the opposition could change the result by making a coalition.

LEMMA A.10

If an entry yields $\#$, then the entry where some elements have been replaced by $\#$ also yields $\#$.

40 Operators and Laws for Combining Preference Relations

PROOF. Assume not: it cannot yield \equiv by Lemma A.7, so it yields $<$ (or symmetrically $>$) as shown in e_2 . Then

e^{\equiv}	\equiv	\equiv	\equiv
$e^<$	\equiv	$\#$	\equiv
$e^>$	$<$	$<$	$<$
$e^{\#}$	$<$	$\#$	$<$
r	$>$	$>$	$>$
by	$>$	$\#$	$>$
r	$\#$	$<$	$<$
by	\times	\times	$A.8 \times$

■

LEMMA A.11

If some elements are replaced by the result (in other words, if the winners are extended), then the result remains the same.

PROOF. If the result is:

- $\#$, the proof follows by Lemma A.10;
- $<$, $>$: by Lemma A.8;
- \equiv : by Lemma A.7, $e^{\equiv} = V$ and thus cannot be extended.

■

DEFINITION A.12

We say an operator *propagates* a property of relations, if its result has the property as soon as one of its arguments relation has it.

An operator *preserves* a property of relations, if its result has the property when all its argument relations have it.

Clearly, propagation implies preservation unless V is empty.

COROLLARY A.13

Any IBU operator preserves reflexivity; propagates irreflexivity; preserves symmetry. Any IBUT operator propagates antisymmetry.

PROOF. By U and Lemma A.7.

■

(These facts are recalled in Theorem 4.1 for the narrower class of priority operators.)

DEFINITION A.14

Let $S, X \subseteq V$ such that S is disjoint from X . S *shows* X iff the entry where all arguments in S are \equiv , all arguments in X are $>$, all other ones are $<$, yields either $>$ or $\#$. This result is called the *show-result*.

LEMMA A.15

If $S \subseteq W$, W disjoint from X , S shows X , then W shows X .

PROOF. Suppose that W does not show X , as indicated in e_1 below. Let $H = V \setminus W \setminus X$ be the rest of the variables.

X	$>$	$>$
S	\equiv	\equiv
$W \setminus S$	\equiv	$<$
H	$<$	$<$
r	$<$	$<$
by	\times	$A.8$

The second entry contradicts the hypothesis that S shows X .

■

LEMMA A.16

If $X \subseteq Y$, Y disjoint from S , S shows X , then S shows Y .

PROOF. Suppose that S does not show Y , as in e_1 . Let $H = V \setminus Y \setminus S$ be the rest of the variables.

X	$>$	$>$
$Y \setminus X$	$>$	$<$
S	\equiv	\equiv
H	$<$	$<$
by	$<$	$A.8$

Again, e_2 contradicts the hypothesis that S shows X . ■

LEMMA A.17

If $A \neq \emptyset$, $V \setminus A$ shows A .

PROOF. By U. ■

LEMMA A.18

If X is finite and disjoint from A , A shows X iff for some $x_i \in X$, A shows $\{x_i\}$.

PROOF. For the implication: We treat the case of $X = \{x_1, x_2, x_3\}$ for notational convenience, but the induction will work for any finite set. Let $H = V \setminus A \setminus X$. Assume (H1) A shows X and for all $x_i \in X$, (H2.i) A doesn't show $\{x_i\}$.

A	\equiv	\equiv	\equiv	\equiv	\equiv	\equiv
x_1	$<$	$>$	$<$	$>$	$<$	$<$
x_2	$>$	$<$	$<$	$>$	$<$	$<$
x_3	$>$	$>$	$>$	$<$	$<$	$<$
H	$>$	$>$	$>$	$>$	$>$	$>$
by	$>$	$>$	$>$	$>$	$>$	$<, \#$
	$H2.1$	$H2.2$	T	$H2.3$	$T \times$	$H1 \times$

The other direction is just Lemma A.16. ■

LEMMA A.19

If A shows disjoint X, Y , then both show-results are $\#$.

PROOF. Since, *a priori*, there two possibilities for both show-results, we have to exclude three cases, but two are symmetric. Let $H = V \setminus X \setminus Y \setminus A$ be the rest.

1. Both show-results are $<$.

A	\equiv	\equiv	\equiv	\equiv
X	$<$	$>$	\equiv	\equiv
Y	$>$	$<$	\equiv	\equiv
H	$>$	$>$	$>$	$>$
by	$<$	$<$	$<$	$>, \equiv$
	$H1$	$H2$	$T \times$	$U \times$

2. One show-result (say X) is $<$, the other is $\#$.

A	\equiv	\equiv	\equiv	\equiv
X	$<$	$>$	\equiv	\equiv
Y	$>$	$<$	\equiv	\equiv
H	$>$	$>$	$>$	$>$
by	$<$	$\#$	$<, \#$	$>, \equiv$
	$H1$	$H2$	$T \times$	$U \times$

The lemmas above demonstrate that 'shows' is completely determined by the sentences of the form ' S shows $\{x\}$ ' where S is minimal. We will now prove that these sentences can be encoded in a priority graph, and finally, that this graph can reconstruct the operator, which closes the cycle and proves the equivalence of all these representations (for V finite). ■

42 Operators and Laws for Combining Preference Relations

DEFINITION A.20

The *priority graph of an IBUT operator* is defined by:

- $N = \{(x, S) \mid S \text{ is a minimal subset of } V \text{ showing } \{x\}\}$.
- $(x_1, S_1) < (x_2, S_2)$ iff $(\{x_1\} \cup S_1) \subseteq S_2$.
- $v((x, S)) = x$.

Note that the node ordering $<$ is irreflexive and transitive and thus acyclic.

LEMMA A.21

If $(x, S) \in N$, then for any $z \in S$, $S \setminus \{z\}$ shows $\{z\}$.

PROOF. (H1) S shows $\{x\}$. Since S is a minimal showing set, (H2) $S \setminus \{z\}$ does not show $\{x\}$. Now assume (H3) $S \setminus \{z\}$ shows $\{z\}$ is false:

x	$<$	$<$	$>$	$<$
z	\equiv	$>$	$<$	\equiv
$S \setminus \{z\}$	\equiv	\equiv	\equiv	\equiv
<i>Rest</i>	$>$	$>$	$>$	$>$
	$<, \#$	$>$	$>$	$>$
	$H1 \times$	$H2$	$H3$	$T \times$

COROLLARY A.22

If V is finite, then for any $(x, S) \in N$, $S = \{z \mid \exists S_z (z, S_z) < (x, S)\}$.

PROOF. Clearly $\{z \mid (z, S_z) < (x, S)\} \subseteq S$ by the definition of the order. Conversely, take $z \in S$. By Lemma A.21, $S \setminus \{z\}$ shows z . Since S is finite, it is Zorn, and so there is a $S_z \subseteq S$ minimal such that S_z shows z , and $(z, S_z) < (x, S)$. ■

LEMMA A.23

Assume V is finite. A shows $\{x\}$ iff x is minimal in $V \setminus A$, i.e. $\exists i \in N (v(i) = x \wedge \nexists i'. v(i') \notin A \wedge i' < i)$.

PROOF. By contraposition, assume A doesn't show $\{x\}$. Since $V \setminus \{x\}$ shows $\{x\}$ by Lemma A.17, there must be a minimal M such that $M \supset A$, M shows $\{x\}$. Since $M \neq A$, we can pick some $z \in M \setminus A$. We have $(x, M) \in N$, and $(z, S_z) \in N$ for some S_z . By Corollary A.22, $(z, S_z) < (x, M)$, contradicting the minimality of x in $V \setminus A$.

Conversely, if x is minimal, all nodes below $i = (x, S)$ are in A . By Lemma A.22, they form S , so S shows $\{x\}$, $S \subseteq A$, $x \notin A$. By Lemma A.16, A shows $\{x\}$. ■

THEOREM 3.2

A finitary operator satisfies conditions IBUT iff it is a priority operator.

PROOF. We show that the priority operator denoted by the priority graph defined for it in Definition A.20, is identical to the given operator. By Lemma A.1, it is sufficient to show this for relations on a universe of two elements (i.e. votes), that is, for any entry e . The priority graph is well-founded, so that we can use Theorem 2.11. Look at the non-abstainers, $\bar{A} = \{x \in V \mid e_x \neq \equiv\}$ and take its minimals for priority $M = \text{Min}_{<}(\bar{A}) = \{x \in \bar{A} \mid \exists i \in N. v(i) = x \wedge \nexists i' \in N. i' < i, v(i') \in \bar{A}\}$. We note that the priority result (the result given by the priority graph) is $\bigcap_{v \in M} e_v$, by Theorem 2.11, and that $M = \{x \mid A \text{ shows } \{x\}\}$, by Lemma A.23. Consider the possible priority results:

- The priority result is \equiv : iff all arguments are \equiv by Theorem 2.11.4; iff the IBUT result is \equiv by Lemma A.7.
- The priority result is $<$: iff $M \neq \emptyset$ and all arguments in M are $<$ by Theorem 2.11(3). A shows M by Lemma A.16. By Lemma A.8, the IBUT result is also $<$.
- The priority result is $>$: symmetrically.
- The priority result is $\#$: iff one of the two following cases arises, by theorem 2.11:
 - some argument x in M is $\#$. *Ad absurdum*, assume that the result isn't $\#$. It can't be \equiv either, by Lemma A.7. Say (H) it is $>$. ($<$ is solved symmetrically.) then by Lemma A.8, A doesn't show $\{x\}$, contradicting Lemma A.23. Tabularly:

$A = e^{\equiv}$	\equiv	\equiv
$e^<$	$<$	$>$
$e^>$	$>$	$>$
x	$\#$	$<$
$e^{\#} \setminus \{x\}$	$\#$	$>$
	$>$	$>$
<i>by</i>	H	$A.8$

– some argument $x \in M$ is $<$, another $y \in M$ is $>$. Then let $X = e^<, Y = e^>$ in Lemma A.19. By Lemma A.10, the IBUT result is $\#$. Tabularly:

$A = e^{\equiv}$	\equiv	\equiv
$X = e^<$	$<$	$<$
$Y = e^>$	$>$	$>$
$R = e^{\#}$	$>$	$\#$
<i>by</i>	$\#$	$\#$
	A.19	A.10

■

A.4 Propagation of properties via priority operators

We prove the theorems implied by Table 3.

THEOREM 4.1

Items 1–8 of Table 3 hold; i.e. the properties reflexivity, irreflexivity, symmetry, antisymmetry, transitivity, totality, empty and full are propagated by the lexicographic combination in the manner shown in the table.

PROOF. Let $g = (N, <, v)$ be a priority graph denoting the operator o , and let $V = v[N]$.

1. Suppose for each $i \in N$, $R_{v(i)}$ is reflexive. We want to show that $o((R_x)_{x \in V})$ is reflexive. Take any $m \in M$. Since $\forall i \in N$, $mR_{v(i)}m$, it follows by Definition 2.5 that $m o((R_x)_{x \in V}) m$.
2. $m o((R_x)_{x \in V}) m$ iff $\forall i \in N$, $mR_{v(i)}m$ by Definition 2.5, since $mR_{v(j)}^<m$ is always false. But $\forall i \in N$, $mR_{v(i)}m$ is false if there there is an irreflexive preference.
3. $m o((R_x)_{x \in V}) n$ implies $\forall i$, $mR_{v(i)}n$ since each $R_{v(i)}$ is symmetric. Therefore $\forall i$, $nR_{v(i)}m$, so $n o((R_x)_{x \in V}) m$.
4. Let i be such that $R_{v(i)}$ is symmetric and there is no infinite $<$ -chain below it in the priority graph. Assume $m o((R_x)_{x \in V}) n$ and $n o((R_x)_{x \in V}) m$ and $m \neq n$. We will derive a contradiction. If $mR_{v(i)}nR_{v(i)}m$ then by symmetry of $R_{v(i)}$ we have $m = n$, a contradiction. Suppose (without loss of generality) that $m\overline{R}_{v(i)}n$. Then there's some $j < i$ such that $mR_{v(j)}^<n$. Therefore, $n\overline{R}_{v(j)}m$, so there is some $k < j$ such that $nR_{v(k)}^<m$. Therefore, $m\overline{R}_{v(k)}n$, and by continuing in this way an infinite chain of nodes below i is produced—a contradiction.
5. Suppose $m_1 o((R_x)_{x \in V}) m_2 o((R_x)_{x \in V}) m_3$; we will show $m_1 o((R_x)_{x \in V}) m_3$. Let $i \in N$; we show $m_1 R_{v(i)} m_3$ or $m_1 R_{v(j)}^< m_3$ for some $j < i$.
 Suppose $m_1 R_{v(i)} m_2$. If $m_2 R_{v(i)} m_3$ then $m_1 R_{v(i)} m_3$. Otherwise, $m_2 \overline{R}_{v(i)} m_3$, so let $i' < i$ be such that $m_2 R_{v(i')}^< m_3$, and let i' be minimal with this property, that is, we have $m_2 R_{v(i'')} m_3$ for $i'' < i'$; here we make use of the fact that $<$ is well-founded. If $m_1 \overline{R}_{v(i')} m_2$, then let $j < i'$ be such that $m_1 R_{v(j)}^< m_2$. Then $j < i$ and $m_1 R_{v(j)}^< m_3$ follows from $m_1 R_{v(j)}^< m_2$ and $m_2 R_{v(j)} m_3$. If $m_1 R_{v(i')} m_2$, let $j = i'$. Then $j < i$, and $m_1 R_{v(j)}^< m_3$ follows from $m_1 R_{v(j)} m_2$ and $m_2 R_{v(j)}^< m_3$.
 On the other hand, suppose $m_1 \overline{R}_{v(i)} m_2$ and let $i' < i$ be minimal such that $m_1 R_{v(i')}^< m_2$ (so again we have $m_1 R_{v(i'')} m_2$ for all $i'' < i'$). Again, consider separately the two cases $m_2 R_{v(i')} m_3$ and $m_2 \overline{R}_{v(i')} m_3$. If $m_2 R_{v(i')} m_3$, set $j = i'$; then $j < i$, and $m_1 R_{v(j)}^< m_3$ follows from $m_1 R_{v(j)}^< m_2$ and $m_2 R_{v(j)} m_3$. Otherwise, $m_2 \overline{R}_{v(i')} m_3$ so let $j < i'$ be such that $m_2 R_{v(j)}^< m_3$; then $j < i$, and $m_1 R_{v(j)}^< m_3$ follows from $m_1 R_{v(j)} m_2$ and $m_2 R_{v(j)}^< m_3$.
6. Suppose $n \overline{o((R_x)_{x \in V})} m$. We show that $m o((R_x)_{x \in V}) n$. Since $n \overline{o((R_x)_{x \in V})} m$, there is i such that $n\overline{R}_{v(i)}m$ and $\forall j < i$, $n\overline{R}_{v(j)}m$. But since these are total orders, this implies $mR_{v(i)}^<n$ and $\forall j < i$, $mR_{v(j)}n$. But $<$ is also total, so this proves that $m o((R_x)_{x \in V}) n$.
7. Let i be the minimal node such that $R_{v(i)}$ is empty. Suppose $m o((R_x)_{x \in V}) n$. Then either $mR_{v(i)}n$, or $\exists j < i \dots$, both alternatives contradicting our hypothesis.
8. Let $m, n \in M$. Since each $R_{v(i)}$ is full, $mR_{v(i)}n$. Thus, by definition 2.5, $m o((R_x)_{x \in V}) n$.
- 9,10. The last two cases are treated separately below due to their length.

■

44 Operators and Laws for Combining Preference Relations

LEMMA A.24

Item 9 of Table 3 holds; i.e. if N is finite, and each $R_{v(i)}$ is transitive and well-founded, then $o((R_x)_{x \in V})$ is well-founded.

PROOF. Suppose not, i.e. suppose $\dots m_3 \ o((R_x)_{x \in V})^< \ m_2 \ o((R_x)_{x \in V})^< \ m_1$ is an $o((R_x)_{x \in V})^<$ -sequence. Each $m_{n+1} \ o((R_x)_{x \in V})^< \ m_n$ gives us an i_n (by Theorem 2.11(3)) such that $m_{n+1} R_{v(i_n)}^< m_n$. Let $N_1 = \{i \in N \mid \{n \mid i = i_n\} \text{ is infinite}\}$. Since N is finite, $N_1 \neq \emptyset$. Let $N_2 \subseteq N_1$ be the $<$ -minimal points of N_1 ; also $N_2 \neq \emptyset$. Let $i \in N_2$, n_0 be the last n where $i_n \notin N_1$. We have $\forall n. \ > n_0 m_{n+1} R_{v(i)} m_n$ and for infinitely many n , $m_{n+1} R_{v(i)}^< m_n$. Since $R_{v(i)}$ is transitive, it is easy to pick a sequence showing that $R_{v(i)}$ is not well-founded, contradicting the hypothesis. ■

THEOREM 4.2

Well-foundedness and \downarrow Zorn are related as follows. Let R be a transitive relation on M . R is well-founded iff (for all $P \subseteq M$) $R|_P$ is \downarrow Zorn.

PROOF. (\Rightarrow .) Let $P \subseteq M$, and let C be an R chain in P . Since $C \subseteq M$ and R is well-founded, C has a minimal element, say c . We now show that c is a lower bound for C . Let $m \in C$. We must show that $c R m$. Since C is a chain, either $m R c$ or $c R m$. If $m R c$ then $c R m$. But also, if $m R c$, then $c R m$, otherwise we would contradict c 's minimality.

(\Leftarrow .) Suppose not; let P be an infinite descending R sequence. As R is transitive, it is an $R|_P$ -chain, but has no $R|_P$ -lower bound, so $R|_P$ is not \downarrow Zorn. ■

Theorem A.30 requires several lemmas. Fix a finite graph $(N, <, v)$ denoting operator o . Let us write R_i instead of $R_{v(i)}$ and R instead of $o((R_x)_{x \in V})$, in order to keep the notation lighter.

DEFINITION A.25

Let $m, n \in M$. The m, n -frontier, written $\text{fr}(m, n)$, is the set of $<$ -minimal elements of the set $\{i \in N \mid m \overline{R}_i n\}$.

Note that if $\{i \in N \mid m \overline{R}_i n\} = \emptyset$ then $\text{fr}(m, n) = \emptyset$.

LEMMA A.26

Suppose $m R n$. Then $i \in \text{fr}(m, n)$ iff $m R_i^< n$ and $\forall j < i. m \overline{R}_j n$.

PROOF. (If) Immediate. (Only if) Let $m R n$ and $i \in \text{fr}(m, n)$. (1) We prove $m R_i n$; for if not, by definition, $\exists j < i. m R_j^< n$, i.e. $m \overline{R}_j n$, contradicting s 's minimality. (2) Since $i \in \text{fr}(m, n)$, $m \overline{R}_i n$. Thus $m R_i^< n$.

Now suppose $j < i$. Since i is minimal in $\{i \in N \mid m \overline{R}_i n\}$, we have $m \overline{R}_j n$. ■

DEFINITION A.27

Let $K \subseteq N$. We write $m R_K n$ if $\forall j \in K. m R_j n$. We also write $\downarrow K$ for $\{i \in N \mid \exists j \in K. i \leq j\}$.

Now, and for the remainder of this subsection, suppose R_i is transitive for each $i \in N$ and N is finite.

LEMMA A.28

Let $P \subseteq M$ be a R -chain with no minimal element. Then there exists $K \subseteq N$ and $a \in P$ such that

1. $\forall j \in K. \forall i \in N. \forall m, n \in P. (n R m R a \text{ and } i \leq j) \text{ implies } n R_i m$ — that is, $\{n \in P \mid n R a\}$ forms a $R_{\downarrow K}$ -chain.
2. $\forall j \in K. \forall m \in P. m R a$ implies $\exists p \in P. (p R^< m \text{ and } p R_j^< m)$ — that is, the same set also forms a R_K -chain with no minimal element.
3. $\forall i \in N. \forall m, n \in P. n R m R a$ implies $(n R_i m \text{ or } \exists j \in K. j < i)$.

PROOF. The idea of the proof is the following. First, we obtain a set $N' \subseteq N$ which contains those i which participate in frontiers all the way down the chain P . Then find an element a of P below which *all* the frontiers are in N' . K is defined as the minimal elements of N' . Then it is possible to prove property 1. Property 2 follows because we have stipulated that P have no minimal element; that is, for each $n \in P$ there is a $n' \in P$ with $n' R^< n$. Property 3 follows because K is the set of minimal elements of N' .

Let $N' = \{i \in N \mid \forall m \in P. \exists n, p \in P. p R^< n R m \text{ and } i \in \text{fr}(n, p)\}$.

- If $N' = N$ then let a be an arbitrary element of P .
- Otherwise, for each $i \in N - N'$ let $m_i \in P$ be such that $\forall n, p \in P$, if $p R^< n R m_i$ then $i \notin \text{fr}(n, p)$, and let $a = \min_R \{m_i \mid i \in N - N'\}$. That each m_i can be found follows from the definition of N' , and that their minimum can be found is guaranteed by the facts that P is a chain and N is finite.

Now we show that N' is non-empty. Let $m, n \in P$ be such that $nR^< mRa$. The fact that P has no minimal element guarantees that these can be found. Since $nR^< m$, $\text{fr}(m, n) \neq \emptyset$, and since m, nRa , we have $\text{fr}(m, n) \subseteq N'$.

1. Let $j \in K$, $i \in N$ and $m, n \in P$ be such that $i \leq j$ and $nRmRa$. If $i \in \text{fr}(m, n)$ then $nR_i^< m$ (Lemma A.26); otherwise, if $i \notin \text{fr}(m, n)$ and $n\bar{R}_i m$ then $\exists j' < i$. $nR_j^< m$, contradicting the minimality of j in K .
2. Let $j \in K$ and $m \in P$ with mRa . Since $j \in N'$, we can pick $n, p \in P$ with $pR^< nRm$ and $j \in \text{fr}(n, p)$. By part 1, $pR_j nR_j m$; and since $j \in \text{fr}(n, p)$ we have $pR_j^< n$. By transitivity, $pR_j^< m$.
3. If $n\bar{R}_i m$ then $\exists j' \in \text{fr}(m, n) \subseteq N'$. $j' < i$ (Theorem 2.11(2)), and since K consists of the minimal elements of N' (and, since N is finite, $<$ is well-founded), $\exists j \in K$. $j < j'$. ■

Now we show, subject to a certain condition, that it is possible to find a lower bound for any R -chain. The condition says that lower bounds can be found for intersections (i.e. conjunctions) of the R_i relations.

LEMMA A.29

Suppose for every $K \subseteq N$, every R_K -chain has a lower bound. Then every R -chain has a lower bound.

PROOF. Let P be a R -chain. If P has a minimal element, then that serves as its lower bound. Suppose, then, that P has no minimal element. Let $K \subseteq N$ and $a \in P$ be as defined in Lemma A.28. Let $U = K \cup \{j' \in N \mid \forall j \in K. j \not\leq j'\}$. We now show that the set $\{m \in P \mid mRa\}$ forms a $R_{\downarrow U}$ chain. Without loss of generality, let $m, n \in P$ be such that $nRmRa$, and $i \in N$ and $j' \in U$ be such that $i \leq j'$. We need to show that $nR_i m$. If $j' \in K$ then $nR_i m$ by Lemma A.28(1). Otherwise, $\forall j \in K. j \not\leq j'$ (definition of U). Therefore, $j \not\leq i$. Suppose $n\bar{R}_i m$. Then by Lemma A.28(3), $\exists j \in K. j < i$, a contradiction. So $nR_i m$.

Now let b be a $R_{\downarrow U}$ lower bound for $\{m \in P \mid mRa\}$. We show that it is also a R lower bound for that set, and hence for P . Let $m \in P$ with mRa ; we show that bRm , using the lexicographic rule.

First note that (i) $j \in \downarrow U$ implies $bR_j m$ (by definition of b). Also, (ii) $j \in K$ implies $mR_j^< b$. To see this, take n such that $nR_j^< m$ by Lemma A.28(2); but then $bR_j n$, so $bR_j^< m$.

Now let $i \in N$. We show that either $bR_i m$ or $\exists j < i$. $bR_j^< m$. If $i \in \downarrow U$, $bR_i m$ by (i). If $i \notin \downarrow U$, then $i \notin U$. By definition of U , $\exists j \in K. j < i$; by (ii), $bR_j^< m$. ■

Hence we have:

LEMMA A.30

Item 10 of Table 3 holds; i.e. if N is finite, and each $R_{v(i)}$ is transitive and for each $K \subseteq N$ the relation $\bigcap_{i \in K} R_{v(i)}$ is $\downarrow\text{Zorn}$, then R is $\downarrow\text{Zorn}$.

A.5 Proof rules for priority graphs

THEOREM 5.2

$g_1 \sqsubseteq g_2$ iff for each $j \in N_2$, there is a $i \in N_1$:

- $v_1(i) = v_2(j)$; and
- $v_1[\downarrow_1 i] \subseteq v_2[\downarrow_2 j]$.

PROOF. Let o_1, o_2 be the operators denoted by g_1, g_2 .

\Rightarrow : Suppose not, i.e. suppose there's an j in N_2 s.t. for every i in N_1 with $v_1(i) = v_2(j) = z$ there is a $k < i$ in N_1 with $v(k) = y$ s.t. $y \notin v[\downarrow j]$.

- Either there is no such i ; then let us set $R_x = F$ for all $x \in V$ except z , and $R_z = \emptyset$. So $o_1((R_x)_{x \in V}) = F$ (since z doesn't occur in it), and $o_2((R_x)_{x \in V}) = \emptyset$ (since z does occur in it): but clearly $F \not\subseteq \emptyset$, so contradiction.
- Or, if some i exists, each i might give us a different y . Let $R_z = \emptyset$; for each of those y s, let $R_y = R$ for some relation R s.t. $R^< \neq \emptyset$ (such a relation exists since M contains two elements); and let $R_x = F$, the full relation, for every other variable x .

Then $o_1((R_x)_{x \in V})$ is just the relation $R^<$. That is because, graphically, it has a collection of F s, \emptyset s and R s (the last two occurring at least once), but there is an R below each \emptyset ; so we just use Definition 2.5. On the other hand, in the graph for $o_2((R_x)_{x \in V})$ we have an \emptyset with only F occurring below it, and by Definition 2.5 the result is \emptyset . Therefore, $o_2((R_x)_{x \in V}) = \emptyset$, so the inclusion fails; contradiction.

\Leftarrow : Suppose $m \ o_1((R_x)_{x \in V}) \ n$. We show $m \ o_2((R_x)_{x \in V}) \ n$. Suppose for some node j in N_2 we have $m \overline{R}_{v_2(j)} \ n$. By the hypothesis, $\exists i \in N_1$. $m \overline{R}_{v_1(i)} \ n$, and since $m \ o_1((R_x)_{x \in V}) \ n$, there is a $k <_1 i$ s.t. $m \overline{R}_{v_1(k)} \ n$. But $v_1[\downarrow_1 i] \subseteq v_2[\downarrow_2 j]$, so there is a $k' \in N_2$ with $v_2(k') = v_1(k)$ and therefore, $m \overline{R}_{v_2(k')} \ n$. Therefore, $m \ o_2((R_x)_{x \in V}) \ n$. ■

Normal forms

In the main text, a canonical form of priority graphs was defined. An important property of this definition is that the variables below a critical node in a graph are the same as those below the corresponding node in the normal form. This lemma will be used in the proof of the theorem that follows.

LEMMA A.31

Let $g' = (N', <', v')$ be the normal form of $g = (N, <, v)$. If $i \in N$ is critical, then $v[\downarrow i] = v'[\downarrow'(v(i), v[\downarrow i])]$.

PROOF. Suppose that $x \in v[\downarrow i]$. Then there's a node $k \in N$ with $k < i$ and $v(k) = x$. k need not be critical, but we know that there is a $j \in N$ critical with $v[\downarrow j] \subseteq v[\downarrow k]$, and $v(j) = v(k)$. Therefore, $v(j) \in v[\downarrow i]$ and $v[\downarrow j] \subseteq v[\downarrow i]$, so $x \in \{v(j) \mid v[\downarrow j] \cup \{v(j)\} \subseteq v[\downarrow i]\} = v'[\downarrow'(v(i), v[\downarrow i])]$.

Conversely, if $x \in v'[\downarrow'(v(i), v[\downarrow i])]$ then there's a $j \in N$ with $v(j) = x$ and $v[\downarrow j] \cup \{v(j)\} \subseteq v[\downarrow i]$, so $x \in v[\downarrow i]$. ■

THEOREM 5.11

1. Any priority graph is equivalent to its normal form.
2. Two priority graphs are equivalent iff their normal form is the same.

PROOF. 1. We apply Corollary 5.5. Suppose $g' = (N', <', v')$ is the normal form of $g = (N, <, v)$, as given in Definition 5.10.

$g \sqsubseteq g'$: If $(v(i), v[\downarrow i])$ is a node in N' then we pick the critical node i in N . We must show (i) that $v'(v(i), v[\downarrow i]) = v(i)$, which is immediate, and (ii) that $v[\downarrow i] \subseteq v'[\downarrow'(v(i), v[\downarrow i])]$, which follows from the Lemma A.31.

$g' \sqsubseteq g$: If i is a node in N , we must find a node in N' with the relevant properties. First, if i is not critical in N , then pick a critical node i' such that $v(i) = v(i')$ and $v[\downarrow i'] \subset v[\downarrow i]$. Now take $(v(i'), v[\downarrow i']) \in N'$. We must show (i) that $v(i) = v'(v(i'), v[\downarrow i'])$, which is immediate, and (ii) that $v'[\downarrow'(v(i'), v[\downarrow i'])] \subseteq v[\downarrow i]$. For that, it is sufficient to show that $v'[\downarrow'(v(i'), v[\downarrow i'])] \subseteq v[\downarrow i']$, which follows from the Lemma A.31.

2. \Rightarrow Let g_1, g_2 be two equivalent graphs, g'_1, g'_2 their normal forms. By 1., the normal forms are equivalent, so by Corollary 5.5, we have two functions, say $f : N'_1 \rightarrow N'_2$ and $g : N'_2 \rightarrow N'_1$, that respect labels ($v(i) = v(f(i))$) and decrease down-sets ($v[\downarrow f(i)] \subseteq v[\downarrow i]$). Let $k = g(f(i))$; $v[\downarrow k] \subseteq v[\downarrow i]$. But $v[\downarrow k] \subset v[\downarrow i]$ is impossible, for then i would not be critical. So $v[\downarrow k] = v[\downarrow i]$. Thus $v[\downarrow f(i)] = v[\downarrow i]$; symmetrically $v[\downarrow g(j)] = v[\downarrow j]$. Using the definition of normal form, we get $f(i) = i$ and $g(j) = j$. Thus $g'_1 = g'_2$.
 \Leftarrow from 1. ■

LEMMA A.32

1. If $g \xrightarrow{\text{link}} g'$ by linking j below some i , then $v[\downarrow i] \subseteq v'[\downarrow' i] \subseteq v[\downarrow i] \cup \{v(i)\}$; and, for all for all $k \in N$ with $k \neq i$, $v[\downarrow k] = v'[\downarrow' k]$.
2. If $g \xrightarrow{\text{del}} g'$ then, for all $k \in N'$, $v[\downarrow k] = v'[\downarrow' k]$.

PROOF. 1. In the case of i , $v'[\downarrow' i] = v[\downarrow i] \cup v[\downarrow j] \cup \{v(j)\} \subseteq v[\downarrow i] \cup \{v(i)\}$. In the case of other ks , the only non-trivial case is where $k > i$. But then, the fact that $v[\downarrow i] \cup \{v(i)\}$ hasn't changed guarantees that $v[\downarrow k]$ hasn't either.

2. The only non-trivial ks are those above the deleted i ; we must show that $v(i) \in v[\downarrow k]$ for those. But that is what is guaranteed by the condition that for all $i' > i$ there exists $i'' < i'$ with $v(i'') = x$. ■

THEOREM 5.15

By applying rules (link) and (del) repeatedly in any order until none applies, any finite priority graph is brought into a form which is equal to its normal form, up to renaming of elements of N .

PROOF. First we show that (link) and (del) are sound. This can be done using Corollary 5.5. Suppose g rewrites to g'

by (link). Corollary 5.5 requires us to find a correspondent in N' for each node in N , and vice versa. Lemma A.32 tells us that usually $v'[\downarrow' i] = v[\downarrow i]$ for all $i \in N$, and hence the correspondent of a node can be the node itself. The only exception occurs in the case that in the link of j below i , we had $v(i) = v(j)$. In that case, $v'[\downarrow' i] = v[\downarrow i] \cup v(i)$, and the correspondent of $i \in N$ should be chosen to be $j \in N'$.

by (del). Again, we must show how to pick the correspondents for Corollary 5.5. For each node in N other than the deleted node, pick the same node in N' . For the deleted node, pick the node in N' referred to as j in the (del) rule. For each node in N' pick the same node in N . Lemma A.32 ensures that these correspondents have the right properties.

To show that the order of application does not matter, we must also show that the term-rewriting system consisting of the set of V -ary finite priority graphs with the rules (link) and (del) is *terminating* and *confluent* [8].

Terminating. Since the graphs are finite, and (link) adds one edge and (del) removes one node, the number of rewrites is bounded by $n^2 + n$, where $n = |N|$.

Confluent. We show that a rule applies unless g is a renaming of the normal form, so that we cannot terminate elsewhere. This implies confluence. Let g be distinct from its normal form.

- Either a node i of g is not critical: (for instance, the node y at mid-height in Example 5.14.1) then by Definition 5.9 of critical, there is a k that either can be linked below i (in Example 5.14.1, the low y), or is already below i , and then i can be deleted.
- Or, several i, j are mapped to the same node of the normal form: (for instance, the two nodes x in Example 5.14.1) if they are not linked, any of them can be linked below the other; else the top one can be deleted.
- Or, all nodes are critical and correspond to a single node of the normal form, but some links are different: In this case, the links of g are a subset of those of the normal form. Then we can add a missing link.

In all three cases, an application of link or del was possible. ■

A.5.1 Preferential entailment

THEOREM 5.18

$g_1 \vdash g_2$ iff $v_2[N_2] \subseteq v_1[N_1]$ and for each node $i \in N_1$ either $v[N_2] \subseteq v_1[\downarrow_1 i]$, or there is a $j \in N_2$ such that $v(i) = v(j)$ and $v[\downarrow j] \subseteq v[\downarrow i]$.

PROOF. Let o_1, o_2 be the operators denoted by g_1, g_2 .

\Rightarrow . Choose some relation S such that $\text{Min}_S(M) \neq M$. (This is possible; as there are at least two elements a, b in M , we could take mSn iff $m = a \wedge n = b$.) Suppose the RHS is false, i.e. either

- $v_2[N_2] \setminus v_1[N_1] \neq \emptyset$. Choose z in this difference, and set $R_z = S, R_x = F$ for any other x ;
 - there is $i \in N_1$ such that $v_2[N_2] \not\subseteq v_1[\downarrow_1 i]$ and for all $j \in N_2$ such that $v_1(i) = v_2(j)$, there is a $x_j \in v_2[\downarrow_2 j] - v_1[\downarrow_1 i]$. If there is such a j , set $R_{v_1(i)} = \emptyset$; for each j set $R_{x_j} = S$; and $R_x = F$ for all other variables x . Else, pick $y \in v_2[N_2] \setminus v_1[\downarrow_1 i]$, set $R_y = S$, set again $R_{v_1(i)} = \emptyset$, and set everything else to F .
- In either case, by an argument similar to that in the proof of Theorem 5.2, we have $o_1((R_x)_{x \in V}) = \emptyset$ and $o_2((R_x)_{x \in V}) = S$. But $\text{Min}(o_1((R_x)_{x \in V})) = M \not\subseteq o_2((R_x)_{x \in V})$, so the LHS is false.

\Leftarrow . Suppose RHS and $n \in \text{Min}(o_1((R_x)_{x \in V}))$. We show that $n \in \text{Min}(o_2((R_x)_{x \in V}))$. Suppose not, i.e. there is an m such that $m o_2((R_x)_{x \in V}) < n$, i.e. $m o_2((R_x)_{x \in V}) n$ and $\exists j \in N_2. mR_{v_2(j)}^< n$. We'll show $m o_1((R_x)_{x \in V}) < n$, i.e. (a) $m o_1((R_x)_{x \in V}) n$ and (b) $\exists j' \in N_1. mR_{v_1(j')}^< n$.

- (a) Suppose $m \overline{R_{v_1(i)}} n$; then by hypothesis, either $v_2[N_2] \subseteq v_1[\downarrow_1 i]$, so $\exists j_1 <_1 i. mR_{v_1(j_1)}^< n$; or there is a $j \in N_2$ such that $v_1(i) = v_2(j)$ and $v_2[\downarrow_2 j] \subseteq v_1[\downarrow_1 i]$; so $m \overline{R_{v_2(j)}} n$ so $\exists k <_2 j$ with $mR_{v_2(k)}^< n$, but using $v_2[\downarrow_2 j] \subseteq v_1[\downarrow_1 i]$ we have that $\exists k' <_1 i$ with $mR_{v_1(k')}^< n$.

- (b) Either case of the hypothesis again provides $j \in N_2$ such that $mR_{v_2(j)}^< n$ and $v_2[N_2] \subseteq v_1[N_1]$. ■

A.6 Composing priority graphs

THEOREM 6.3

Let g be a well-founded graph denoting operator o with variables V . Let $(g_x)_{x \in V}$ be a family of well-founded graphs denoting operators $(o_x)_{x \in V}$ with variables $(V_x)_{x \in V}$. Let g' be the graphical insertion of $(g_x)_{x \in V}$ in g , and let o' be the operator denoted by g' .

Then o' is the composition of o with $(o_x)_{x \in V}$, i.e.

$$o' \left((R_y)_{y \in \bigcup \{V_x \mid x \in V\}} \right) = o \left((o_x((R_y)_{y \in V_x}))_{x \in V} \right).$$

PROOF. First observe that if g, g_1, \dots, g_n are well-founded, then so is g' . This enables us to use Theorem 2.11. Let us write $g = (N, <, v)$ and $g_x = (N_x, <_x, v_x)$ for each $x \in V = \{1, \dots, n\}$. Now,

$$\begin{aligned} m \ o'((R_x)_{x \in V}) \ n \\ \iff \forall i \in N. \forall i' \in N_{v(i)}. \left(m R_{v_{v(i)}(i')} \ n \right. \\ \quad \vee \exists j' \in N_{v(i)}. (j' <_{v(i)} i' \wedge m R_{v_{v(i)}(j')} \ n) \\ \quad \left. \vee \exists j \in N. \exists j' \in N_{v(j)}. (j < i \wedge m R_{v_{v(j)}(j')} \ n) \right). \end{aligned}$$

We simplify notation for this proof, by writing N_i and $<_i$ in place of $N_{v(i)}$ and $<_{v(i)}$, and by writing Rij' instead of $m R_{v_{v(i)}(j')} \ n$ (m, n are fixed). We will consistently use unprimed variables for the 'outer' level indices, and primed variables for the 'inner' ones. Thus

$$\begin{aligned} m \ o'((R_x)_{x \in V}) \ n \\ \iff \forall i \in N. \forall i' \in N_i. \left(Rii' \right. & (1a) \\ \quad \vee \exists j' \in N_i. (j' <_i i' \wedge R < ij') & (1b) \\ \quad \left. \vee \exists j \in N. \exists j' \in N_j. (j < i \wedge R < jj') \right) & (1c) \\ \iff \forall i \in N. \forall i' \in N_i. \left(Rii' \right. & (2a) \\ \quad \vee \exists j' \in N_i. (j' <_i i' \wedge R < ij') & (2b) \\ \quad \vee \exists j \in N. \exists j' \in N_j. (j < i \wedge R < jj' \\ \quad \quad \wedge \forall k \in N. (k < j \rightarrow \forall i' \in N_k. Rki')) & (2d) \end{aligned}$$

version (2) following from version (1) by Theorem 2.11(2). But now,

$$\begin{aligned} m \ o(o_1((R_x)_{x \in V}), \dots, o_n((R_x)_{x \in V})) \ n \\ \iff \forall p \in N. (m \ o_{v(p)}((R_x)_{x \in V}) \ n \vee \exists q \in N. (q < p \wedge m \ (o_{v(q)}((R_x)_{x \in V})) < n)) \\ \iff \forall p \in N. \left((\forall p' \in N_p. (Rpp' \vee \exists q' \in N_p. (q' <_p p' \wedge R < pq'))) \right. & (3a) \\ \quad \vee \exists q \in N. \left(q < p \wedge \forall p' \in N_q. \left(Rqp' \right. & (3b) \\ \quad \quad \vee \exists q' \in N_q. (q' <_q p' \wedge R < qq') \right) & (3c) \\ \quad \left. \wedge \exists q' \in N_q. R < qq' \right) \right). & (3d) \end{aligned}$$

3b-d comes from the expansion of $m \ (o_{v(j)}((R_x)_{x \in V})) < n$ using Theorem 2.11(3).

That (3) implies (1) is easy: if 1a and 1b are not satisfied, set $j = q$ in 3b and $j' = q'$ in 3d to satisfy 1c. So all that remains is to show that (2) implies (3).

Suppose we have p, p' which do not satisfy the disjuncts in 3a. We need to find an appropriate q . Setting $q = j$ from 2c might work; if it does, we are home. If it doesn't, we have a troublesome $p' \in N_q$ for which not Rqp' and there is no appropriate q' .

Use (2) again with $i = q$ and $i' = p'$, to obtain a $j < q$ and $j' \in N_j$, which we will call $r < q, r' \in N_r$. Since $r < q$, we have by 2d $\forall s' \in N_r. Rrs'$; and by transitivity we have $r < p$, so r satisfies the conditions for q in 3b. Moreover, $R < rr'$ (from 2c) guarantees 3d. ■

The extraction of terms from priority graphs was given by example in the main text. Here, we give formal definitions in order to prove Theorem 6.6.

DEFINITION A.33

To eliminate such shapes as the N shape in Example 6.8, we define the *forest form* $g' = F(g)$ of g as:

- N' is the the set of maximal up-branches in G . Formally:

$$N' = \{(i_1, \dots, i_n) \mid n > 0, \forall l < n (i_l < i_{l+1} \wedge \nexists j \in N. (i_l < j < i_{l+1}) \wedge \nexists j \in N. (i_n < j))\}.$$

- $<'$ is the suffix ordering. Formally $\sigma <' \tau$ iff there is a non-empty sequence of nodes π such that $\sigma = \pi; \tau$.
- v' takes the label where the branch starts, i.e. if $\sigma = (i_1, \dots, i_n)$ then $v'(\sigma) = v(i_1)$.

Actually this definition simply removes any ‘V’ shape from the graph by replicating the node at the bottom of the ‘V’ that becomes ‘II’. In particular, we replace any ‘N’ shape by a ‘Λ I’ shape. This is not always necessary, for instance in Example 5.6 the V-shaped example could be expressed directly as $(x||y)/z$.

PROPOSITION A.34

$$g \equiv F(g).$$

PROOF. All down-sets are preserved, so we can use Corollary 5.5. ■

DEFINITION A.35

Termifying a finite priority graph g to $T(g)$ is done as follows:

- if g is made of a single node labelled by x , set $T(g)x$;
- if g is made of disjoint components g_1, \dots, g_n , then we set $T(g) = T(g_1)|| \dots || T(g_n)$;
- else, find a $M \subset N$ such that $\forall m \in M, n \in N \setminus M$ we have $n < m$, as follows: Start by setting M to the maximal nodes of N ; and while there is a node which is not below all elements of M , add it to M . This algorithm may stop with $M = N$, in which case it signals failure; else, we set $T(g) = T(M)/T(N \setminus M)$.

We see that the algorithm succeeds exactly when g is the graphical insertion of some term (equivalently, when no N shape is included in g); this term is unique up to associativity of $/$ and $||$, and commutativity of $||$. ($T(g)$ will have $/$ associated to the left, since we started from the top.)

THEOREM 6.6

Any finitary priority operator is denoted by a term built from $/, ||$ and the variables that occur in the priority graph for the operator.

PROOF. Take any finitary V -ary operator o . Let g be a graph denoting o . Let g' be the forest form of g . It is easy to check that we can always termify a forest form: The last step succeeds immediately, and M contains the single maximum element (the root of the tree). So o can be expressed by $T(g')$. ■

A.7 Algebraic treatment

DEFINITION A.36

\vdash denotes equational derivation from axioms 1–7. This means that a proof can use axioms 1–7, and the classical rules of equality:

Reflexivity	$\vdash \tau = \tau$
Symmetry	$\tau = \sigma \vdash \sigma = \tau$
Transitivity	$\rho = \sigma, \sigma = \tau \vdash \rho = \tau$
Congruence	$\tau = \sigma \vdash \rho[x := \tau] = \rho[x := \sigma]$

In order to prove the soundness and completeness of the axioms of Theorem 7.2, we need a lemma.

LEMMA A.37

$\vdash \sigma/\tau = \tau$, if $v(\sigma) \subseteq v(\tau)$.

PROOF. (Note that this is obviously valid semantically, since all occurrences in the σ part of σ/τ are non-critical.)

We first induce on the structure of σ :

1. if σ is the variable x : we proceed by induction on the structure of the term τ .

- (a) τ is a variable; since $x \in v(\tau)$, τ is the variable x , so use idempotence of $/$.
- (b) $\tau = (\rho||\sigma)$: Then $x \in v(\rho)$ or $x \in v(\sigma)$. Without loss of generality, assume $x \in v(\rho)$. Then $\vdash \rho = x/\rho$ by the inductive hypothesis, and thus $\vdash x/\tau = x/((x/\rho)||\sigma)$.
But $\vdash x/((x/y)||z) = (x/y)||z$ is derivable (Example 7.3(6)), thus $\vdash x/((x/\rho)||\sigma) = (x/\rho)||\sigma = \rho||\sigma = \tau$.

50 Operators and Laws for Combining Preference Relations

(c) $\tau = \rho/\sigma$.

- $x \in v(\sigma)$. Then

$$\begin{aligned} x/\tau &= x/\rho/\sigma && \text{def. } \tau \\ &= x/\rho/x/\sigma && \text{ind. hyp.} \\ &= \rho/x/\sigma && \text{example 7.3(3)} \\ &= \rho/\sigma && \text{ind. hyp.} \\ &= \tau && \text{def. } \tau. \end{aligned}$$

- $x \in v(\rho)$. Then

$$\begin{aligned} x/\tau &= x/\rho/\sigma && \text{def. } \tau \\ &= \rho/\sigma && \text{ind. hyp.} \\ &= \tau && \text{def. } \tau. \end{aligned}$$

2. $\sigma = (\sigma_1/\sigma_2)$: we use associativity of $/$ to obtain $\sigma_1/(\sigma_2/\tau)$, and first eliminate σ_1 inductively, then σ_2 .

3. $\sigma = (\sigma_1\|\sigma_2)$: we use distributivity to obtain $(\sigma_1/\tau)\|(\sigma_2/\tau)$, and process inductively each part. ■

THEOREM 7.2

An equation is true in all preferential algebras iff it is derivable from the following seven axioms:

- | | | |
|----|---------------------------|-------------------------------|
| 1. | $x\ x = x$ | ($\ $ Idempotent) |
| 2. | $x\ (y\ z) = (x\ y)\ z$ | ($\ $ Associative) |
| 3. | $x\ y = y\ x$ | ($\ $ Commutative) |
| 4. | $(x/x) = x$ | ($/$ Idempotent) |
| 5. | $x/(y/z) = (x/y)/z$ | ($/$ Associative) |
| 6. | $(x\ y)/z = (x/z)\ (y/z)$ | ($/$ Distributes over $\ $) |
| 7. | $(x/y)\ x = x\ y$ | (Absorption) |

PROOF. The soundness of the axioms is obvious. (For example, apply Corollary 5.5 to the graph forms of each side of the axioms.)

Completeness: let $\tau \sqsubseteq \delta$ abbreviate $\tau\|\delta = \tau$ (indeed, this use of \sqsubseteq matches that in the semantics). We need only prove statements of the form $\tau \sqsubseteq \delta$, since to prove $\tau = \delta$ we just prove $\tau \sqsubseteq \delta$ and $\delta \sqsubseteq \tau$, which expands to $\tau = \tau\|\delta = \delta$.

Suppose $\tau \sqsubseteq \delta$ semantically. We prove $\tau \sqsubseteq \delta$ by induction on δ .

1. δ is the variable x . We perform induction on τ .

(a) τ is a variable. Since $\tau \sqsubseteq \delta$, τ must also be x (by Theorem 5.2). Idempotence finishes the proof.

(b) $\tau = \tau_1/\tau_2$. By Theorem 5.2 we know $\tau_1/\tau_2 \sqsubseteq x$ iff $\tau_2 \sqsubseteq x$, and by inductive hypothesis $\tau_2 \sqsubseteq x$. We prove $\tau_1/\tau_2 \sqsubseteq x$ as follows:

$$\begin{aligned} (\tau_1/\tau_2)\|x &= (\tau_1/\tau_2)\|\tau_2\|x && \text{Example 7.3(1)} \\ &= (\tau_1/\tau_2)\|\tau_2 && \text{since } \tau_2 \sqsubseteq x \\ &= \tau_1/\tau_2 && \text{Example 7.3(1)} \end{aligned}$$

(c) $\tau = \tau_1\|\tau_2$. By Theorem 5.2 we know $\tau_1\|\tau_2 \sqsubseteq x$ iff $\tau_1 \sqsubseteq x$ or $\tau_2 \sqsubseteq x$. Without loss of generality we suppose it is τ_1 , and by inductive hypothesis we have $\tau_1 \sqsubseteq x$. Now $\tau_1\|\tau_2\|x = (\tau_1\|x)\|\tau_2 = \tau_1\|\tau_2$, so $\tau_1\|\tau_2 \sqsubseteq x$.

2. $\delta = \gamma\|\sigma$. By the semantics we know that $\tau \sqsubseteq (\gamma\|\sigma)$ is valid iff $\tau \sqsubseteq \gamma$ and $\tau \sqsubseteq \sigma$, so by inductive hypothesis we prove $\tau \sqsubseteq \gamma$ and $\tau \sqsubseteq \sigma$, which expand to $\tau\|\gamma = \tau$ and $\tau\|\sigma = \tau$, from which we prove $\tau = \tau\|(\gamma\|\sigma)$ using associativity, commutativity and idempotence.

3. $\delta = \gamma/\sigma$. By induction on γ . γ can be:

(a) $\gamma_1\|\gamma_2$: then we use distribution.

(b) γ_1/γ_2 : then we use associativity to obtain $\delta = \gamma_1/(\gamma_2/\sigma)$.

(c) A variable x . If x occurs in σ , we suppress it using Lemma A.37. The remaining case is to prove inequalities of form $\tau \sqsubseteq x/\sigma$, where x is a variable not occurring in σ . By Theorem 5.2, an inequation of this form is valid iff $\tau \sqsubseteq \sigma$ and in the graph of τ there is a node labelled by x such that $v[\downarrow x] \sqsubseteq v[\sigma]$. We can assume without loss of generality that τ is in forest form, since we just have to apply distribution repeatedly to obtain this form. Let η denote the subterm below x in the forest form ($\tau = \dots/(\dots\|(x/\eta))$). By convention, we treat the case where η is empty uniformly.

i. We prove $\tau \sqsubseteq x/\eta$ by induction on τ . Since it is in forest form, τ can be:

- A. y/τ_2 : If $y = x$ and $\tau_2 = \eta$ we are done.
 Otherwise we rewrite τ to $(y/\tau_2)\|\tau_2$ using Example 7.3(1) backwards, and we proceed on this last τ_2 which must have an occurrence of x/η since $y \neq x$. Then Theorem 5.2 gives $\tau_2 \sqsubseteq x/\eta$, which by inductive hypothesis gives $\vdash \tau_2 \sqsubseteq x/\eta$, thus $\vdash (y/\tau_2)\|\tau_2 \sqsubseteq x/\eta$ using associativity of $\|$.
- B. $\tau_1\|\tau_2$: We know x/η must occur in τ_1 or τ_2 (or both); we proceed inductively on that part, say τ_2 . Again $\tau_2 \sqsubseteq x/\eta$ implies $\vdash \tau \sqsubseteq x/\eta$ using by Theorem 5.2, inductive hypothesis, and associativity of $\|$.
- ii. Let's put this together:
- $\vdash \tau \sqsubseteq x/\eta$ just proved
 - $\vdash \tau \sqsubseteq \sigma$ by inductive hypothesis
 - $\vdash \tau \sqsubseteq (x/\eta)\|\sigma$ as in case 2
 - $\vdash \tau \sqsubseteq x/\eta/\sigma$ by 7.3(7)
 - $\vdash \tau \sqsubseteq x/\sigma = \delta$ by Lemma A.37

EXAMPLE A.38

We apply the algorithm of the proof of Theorem 7.2 to construct a proof of $x/(y\|z) = (x/y/z)\|(x/z/y)$:

$$\begin{aligned}
 x/(y\|z) &= (x/(y\|z))\|y\|z && 7.3(1) \\
 &= (x/(y\|z))\|y\|z\|(z/y) && 7.3(7) \\
 &= (x/(y\|z))\|(z/y) && 7.3(1) \\
 &= (x/(y\|z))\|((x/(y\|z))/(z/y)) && \text{axiom 7} \\
 &= (x/(y\|z))\|(x/z/y) && \text{A.37} \\
 &= (x/(y\|z))\|y\|z\|(x/z/y) && 7.3(1) \\
 &= (x/(y\|z))\|y\|z\|(y/z)\|(x/z/y) && 7.3(7) \\
 &= (x/(y\|z))\|(y/z)\|(x/z/y) && 7.3(1) \\
 &= (x/(y\|z))\|((x/(y\|z))/(y/z))\|(x/z/y) && 7.3(1) \\
 &= (x/(y\|z))\|(x/y/z)\|(x/z/y) && \text{A.37} \\
 &= ((x/z/y)/(y\|z))\|(x/y/z)\|(x/z/y) && \text{A.37} \\
 &= (x/y/z)\|z\|(x/z/y)\|y && \text{axiom 7} \\
 &= (x/y/z)\|(x/z/y)\|y && 7.3(1) \\
 &= (x/y/z)\|(x/z/y) && 7.3(1)
 \end{aligned}$$

This identity is the basis of the *Tuscan form*: given a term, rewrite it first using distributivity, and then this identity. By this process, any term is brought in a form where $\|$ are outside and $/$ inside. We can use 3, 4, 1 and 7 to eliminate some duplicates, but this will not yield some unique normal form. For instance, $x/(y\|z)/w = x/((y/w)\|(z/w)) = (x/y/w/z/w)\|(x/z/w/y/w) = (x/y/z/w)\|(x/z/y/w) = (x/y/z/w)\|(y/w) = (z/w)\|(x/z/y/w)$; the last four are Tuscan forms, the last two are simplified.

The equations 1–7 given in Theorem 7.2 are not complete, however, with respect to conditional equations (implications between equations).

THEOREM A.39

There is a conditional equation true in all preferential algebras which is not a consequence of 1–7; for example, $x/y/z = z/y/x \vdash x/z = z/x$ is such a conditional equation.

PROOF. *The conditional equation is true in all PAs:* expand $/, \|\$ using the equations in Proposition 6.5; now, we want to prove that $(x \cap y \cap z) \cup (y^< \cap z) \cup z^< = (x \cap y \cap z) \cup (y^< \cap x) \cup x^<$ implies $(x \cap z) \cup z^< = (x \cap z) \cup x^<$. Suppose the premiss and that $m((x \cap z) \cup z^<) n$. Then either $m(x \cap z) n$, so $m((x \cap z) \cup x^<) n$, and we are done; or $m z^< n$ and $m(x \cap z) n$. $m z^< n$ implies $m(x \cap y \cap z) \cup (y^< \cap z) \cup z^< n$, since the last disjunct is true. $m(x \cap z) n$ means $m \bar{x} n$ or $m \bar{z} n$. Since $z^< \sqsubseteq z$, the second half is impossible and we have $m \bar{x} n$. Using the premiss, $m(x \cap y \cap z) \cup (y^< \cap x) \cup x^< n$, so $m x n$, a contradiction.

The conditional equation cannot be derived from the axioms 1–7: in axioms 1–7, and here in the antecedents, the same variables occur in the left- and right-hand side. By examining the rules for deriving equations (Definition A.36), we notice that no rule can eliminate a variable from the antecedent; thus the conclusion must contain y if the proof uses the antecedent. On the other hand, the proof must use the antecedent, since the consequent is not valid and thus not a consequence of axioms 1–7. ■

This means that the class PA of all isomorphic copies of preferential algebras is not axiomatizable by equations, but we now show that PA can be axiomatized by conditional equations.

52 Operators and Laws for Combining Preference Relations

THEOREM A.40

PA is a quasi-variety.

PROOF. We use standard techniques [22] of algebras of relations, namely, we prove that the class K of algebras isomorphic to a preferential algebra is closed under taking subalgebras, direct products, and ultraproducts.

- K is closed under taking subalgebras, by definition.
- K is closed under taking direct products: let I be a set and for each $i \in I$ let $\langle A_i, \cap, / \rangle$ be a preferential algebra. That is, A_i is a set of binary relations on some U_i closed under intersection and lexicographic combination. We may assume that the U_i s are pairwise disjoint. Let U be the union of these U_i s. For any tuple $a = \langle a_i : i \in I \rangle$ of elements of the product ($a_i \in A_i$), let $f(a)$ be the union of a_i s, which is indeed a binary relation on U . Let A be the set of the all these $f(a)$ s. Then A is closed under:
 - intersection: $(\bigcup_i a_i) \cap (\bigcup_i b_i) = \bigcup_i (a_i \cap b_i)$, since the U_i are disjoint. Now since each A_i is closed, A is;
 - lexicographic combination: $(\bigcup_i a_i) / (\bigcup_i b_i) = \bigcup_i (a_i / b_i)$, for if $m(\bigcup_i b_i) \equiv n$, it means that $m, n \in U_i$ for some unique i , and thus $mb_i \equiv n$.
 The function f is an isomorphism from the direct product of the algebras A_i to the algebra $\langle A, \cup, / \rangle$: its inverse is just the tuple of projections on the U_i s.
- K is closed under taking ultraproducts: The operations of K are definable in BRA , the class of binary relation algebras (i.e. K is a generalized reduct of BRA). It is known that BRA is closed under taking ultraproducts (claim 1.1 of [22]). Hence K is closed under taking ultraproducts. ■

The axioms presented in Theorem 7.2 are also complete for inclusion, since $R_1 \subseteq R_2$ iff $(R_1 \parallel R_2) = R_2$. It is also possible to construct a proof system for inclusion without resorting to equality:

- | | |
|--|--------------------------------------|
| 1. $x \sqsubseteq x$ | (reflexivity) |
| 2. $x \sqsubseteq y, y \sqsubseteq z$ implies $x \sqsubseteq z$ | (transitivity) |
| 3. $x \sqsubseteq y$ implies $x \parallel z \sqsubseteq y \parallel z$ | (monotonicity \parallel) |
| 4. $x \sqsubseteq y$ implies $x / z \sqsubseteq y / z$ | (monotonicity $/a$) |
| 5. $x \sqsubseteq y, y \sqsubseteq x$ implies $z / x \sqsubseteq z / y$ | (monotonicity $/b$) |
| 6. $x \sqsubseteq x \parallel x$ | (\parallel Idempotent) |
| 7. $x \parallel (y \parallel z) \sqsubseteq (x \parallel y) \parallel z$ | (\parallel Associative) |
| 8. $x \parallel y \sqsubseteq y \parallel x$ | (\parallel Commutative) |
| 9. $x \sqsubseteq (x / x)$ | ($/$ Idempotent) |
| 10. $x / (y / z) \sqsubseteq (x / y) / z$ | ($/$ Associative) |
| 11. $(x / y) / z \sqsubseteq x / (y / z)$ | ($/$ Associative) |
| 12. $(x \parallel y) / z \sqsubseteq (x / z) \parallel (y / z)$ | ($/$ Distributes over \parallel) |
| 13. $(x / z) \parallel (y / z) \sqsubseteq (x \parallel y) / z$ | ($/$ Distributes over \parallel) |
| 14. $x \parallel y \sqsubseteq (x / y) \parallel x$ | (Absorption) |
| 15. $x / y \sqsubseteq y$ | ($/$ -refinement) |

THEOREM 7.4

A preferential entailment $\tau \vdash \sigma$ holds in all preferential algebras iff it is derivable from the equality axioms 1–7, together with the following:

- | | |
|--|------|
| 16. If $x \vdash y$ then $z / x \vdash y$ | (C1) |
| 17. If $y / x = x$ and $x \parallel y = y$ then $x \vdash y$ | (S1) |

PROOF. \Leftarrow . We check the soundness of the two new rules.

C1. If $\text{Min}_x(M) \subseteq \text{Min}_y(M)$, then indeed $\text{Min}_{z/x}(M) \subseteq \text{Min}_y(M)$, since the minimals of z/x are among the minimals of x .

S1. $x \parallel y = y$ means that $y \sqsubseteq x$. $y / x = x$ means that $mx \equiv n \Rightarrow my \equiv n$. So if $m' y < m$, then also $m' x < m$, for all three other possibilities are excluded. So if m is not minimal for y , it means that $\exists m'. m' y < m$, thus $m' x < m$, and m is neither minimal for x .

\Rightarrow . We want to prove, say, $\tau \vdash \sigma$. Let g_1, g_2 be their graphs. We use Theorem 5.18. First let $I = \{i \mid v[\downarrow i] \supseteq v(N_2)\}$. I is upward-closed. If $I = \emptyset$, let τ_2 be a term representing σ . Otherwise we construct τ_2 as follows:

For all $k \notin I$, if $i < k$ then $i \notin I$, so that by Theorem 5.18 $v(i) = v(j)$ for some $j \in N_2$; therefore we have $v[\downarrow k] \subseteq v(N_2) \subseteq v(i)$, for any $i \in I$, so that we link any node $k \notin I$ below each minimal $i \in I$ using rule (link). Therefore, the graph is now of the form g_1 / g_2 where g_1 contains all nodes of I and g_2 the rest. We find terms τ_1, τ_2 expressing g_1, g_2 by Theorem 6.6. Since τ is equivalent to τ_1 / τ_2 by their construction, this is provable by completeness (Theorem 7.2).

Since τ_2 only contains nodes outside I : by Theorem 5.18, $v(\tau_2) \subseteq v(\sigma)$. Also, by Theorem 5.2, $\sigma \sqsubseteq \tau_2$. By completeness, $\tau_2 \parallel \sigma = \sigma$ is provable. By Corollary 5.4, $v(\tau_2) \supseteq v(\sigma)$ and thus $v(\tau_2) = v(\sigma)$. So in σ/τ_2 , all occurrences in σ are non-critical, implying that $\sigma/\tau_2 = \tau_2$ is valid, and thus provable by completeness. Thus, we can use rule S1 to prove $\tau_2 \vdash \sigma$, and then rule C1 to prove $\tau_1/\tau_2 \vdash \sigma$. ■

Received 19 October 1998