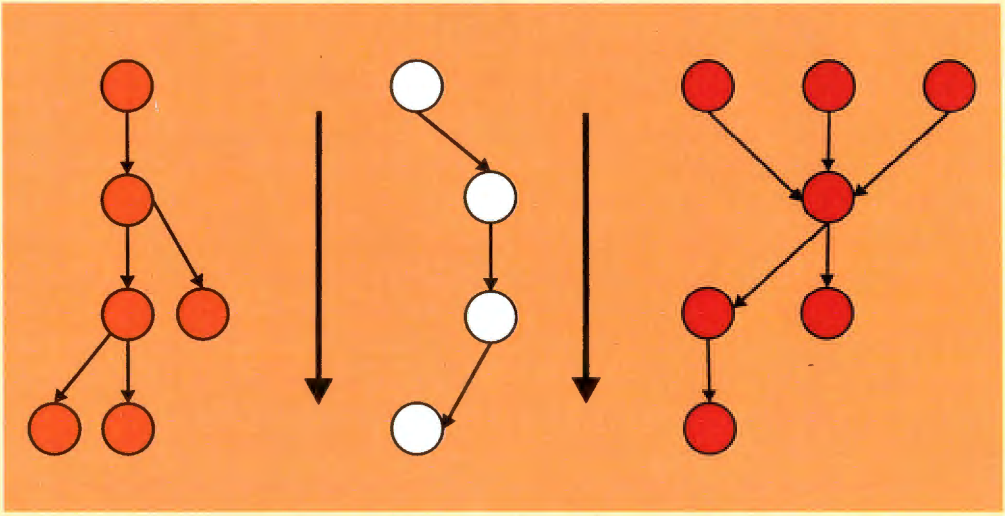
SYSTEMS RESEARCH INSTITUTE
POLISH ACADEMY OF SCIENCES

MULTICRITERIA ORDERING AND RANKING:
PARTIAL ORDERS, AMBIGUITIES AND APPLIED ISSUES



Jan W. Owsiński and Rainer Brüggemann
Editors

Warsaw 2008

SYSTEMS RESEARCH INSTITUTE
POLISH ACADEMY OF SCIENCES

MULTICRITERIA ORDERING AND RANKING: PARTIAL ORDERS, AMBIGUITIES AND APPLIED ISSUES

Jan W.Owsiński and Rainer Brüggemann
Editors

Warsaw 2008

This book is the outcome of the international workshop held in Warsaw in October 2008 within the premises of the Systems Research Institute. All papers were refereed and underwent appropriate modification in order to appear in the volume. The views contained in the papers are, however, not necessarily those officially held by the respective institutions involved, especially the Systems Research Institute of the Polish Academy of Sciences.

ISBN 83-894-7521-9
EAN 9788389475213

Technical editing and typesetting:
Jan W.Owsiński, Anna Gostyńska, Aneta M.Pielak

Theoretical Developments

Multicriteria Ordering and Ranking: Partial Orders, Ambiguities and Applied Issues
Jan W. Owsiński and Rainer Brüggemann, Editors

Informative Combination of Multiple Partial Order Relations

Michaël Rademaker*, **Bernard De Baets*** **and Hans De Meyer**†

*Department of Applied Mathematics, Biometrics and Process Control,
Ghent University, Coupure links 653, 9000 Gent, Belgium
(michael.rademaker@ugent.be, bernard.debaets@ugent.be)
†Department of Applied Mathematics and Computer Science,
Ghent University, Krijgslaan 281 (S9), 9000 Gent, Belgium
(hans.demeyer@ugent.be)

In a previous paper, we discussed various ways in which to construct and process partial order relations or partially ordered sets (posets) in the context of ranking objects on the basis of multiple criteria. We now provide a more straightforward characterization of the consistent and prioritized union operations, and provide straightforward algorithmic implementations.

Keywords: partially ordered set, transitive combination of partial order relations

1. Introduction

We extend on the work in a previous paper, see Rademaker et al. (2008), dealing with operations to combine two partial order relations in an informative way. In this paper, rather than restricting ourselves to two partial order relations, we describe a consistent framework to process an arbitrary number of partial order relations, and formulate easily and efficiently implementable algorithms. We introduce the required basic concepts in Section 2, and construct our specific basic algorithms and structure in Section 3. The specific union operations are described in Sections 4 and 5, once again providing workable algorithms. We examine some properties in Section 6, and conclude with a summary in Section 7.

2. Preliminaries

A (binary) relation R on a set of objects P denotes some property or characteristic objects of P can have w.r.t. each other, i.e. xRy means "x is R-related to y". For example, R could denote "smaller than" or "less polluted than". A relation R on P can be represented as a set of couples of objects from P, for example $R = \{(a,b),(c,d)\}$ denotes aRb and cRd. If a relation R fulfills the properties of reflexivity (xRx) and transitivity (xRy and yRz imply xRz), it is called a pre-order relation. If it also has the property of antisymmetry (xRy and yRx imply $x = y$), in addition to reflexivity and transitivity, the relation R constitutes a (partial) order relation. If xRy or yRx, x and y are commonly said to be comparable. An order relation is commonly denoted by the symbol $\leq$, and a couple $(P,\leq)$, with P a set of objects and $\leq$ an order relation on P, is called a partially ordered set, or poset for short.

We are dealing with a set of objects P which have been ordered on the basis of several order relations. As such, we have $(P,\leq_1)$, $(P,\leq_2)$, $(P,\leq_3)$ and so on.

A relation R' on P is called an extension of a relation R on P if it holds that $R \subseteq R'$ (it is equivalent to say that R is a subset of R'). The unique smallest transitive extension of R is called the transitive closure of R (commonly computed via the Floyd-Warshall algorithm, see Floyd, 1962, and Warshall, 1962, though other algorithms exist, see De Baets and De Meyer, 2003, and Naessens et al, 2002). Similarly, a poset $(P,\leq')$ is called an extension of a poset $(P,\leq)$ if $\leq'$ is an extension of $\leq$. We say that two posets $(P,\leq)$ and $(P,\leq')$ contradict each other on two objects $x,y \in P$ if we have $x < y$ and $y <' x$, or $y < x$ and $x <' y$.

3. The $\{T,N,F\}$ framework and the basic algorithms

Our previous definition of the consistent union catered to the special case of two partially ordered sets as input. We now extend this to an arbitrary number of sets, after which we will examine some properties of the consistent union operation. To this end, we formulate an intermediate structure composed of a triple of relations: the tentative relations T, which contains a relation as soon it is present in one of the sets, the necessary relations N, which contain those relations present in every set, and the forbidden relation F, initialized as the inverse of T. We first show in Algorithm 1 how to construct the set of $\{T,N,F\}$ relations on the basis of a set of partial order relations $\{P_1,\ldots,P_n\}$ on a single set of objects P (naturally, Algorithm 1 is also suited to constructing a triple $\{T,N,F\}$ for a single poset). The set T of tentative relations is straightforwardly constructed by taking the union of the set of partial orders. In contrast, the set N of necessary relations is the intersection of the set of partial orders. The set F of forbidden relations is the inverse of the set of tentative relations. Observe that it is possible, even probable, to have $T \cap F \neq \emptyset$. This will occur as soon as for two partial orders P_i and P_j from $\{P_1, \ldots, P_n\}$, we have a contradiction on some x and y, or, equivalently, we have $P_i \cap P_j^{-1} \neq \emptyset$. In a subsequent step, we will extract a unique partial order from $\{T, N, F\}$.

We now address the problem of how to extract a specific uniquely defined maximum informative partial order relation R from a set of $\{T, N, F\}$ relations, for which it holds that (1) it corresponds to a transitive closure of a subset of T, (2) it is an extension of N, and (3) $R \cap F = \emptyset$. We detail the required steps in Algorithm 2. As an aside, we draw the parallel to our definition in Rademaker et. al (2008): the relation R we use is uniquely defined because it is the intersection of all maximal relations which satisfy conditions (1), (2) and (3).

Algorithm 1: Constructing the set of $\{T, N, F\}$ relations

Data: Set of partial orders $\{P_1, \ldots, P_n\}$ on a single set of objects P
Result: Set of $\{T, N, F\}$ relations
$T \leftarrow \emptyset$;
$N \leftarrow P \times P$;
$F \leftarrow \emptyset$;
foreach $P_i \in \{P_1, \ldots, P_n\}$ **do**
  $T = T \cup P_i$;
  $N = N \cap P_i$;
  $F = F \cup (P_i)^{-1}$;
end

Algorithm 2 uses an intermediate R, denoted R', to prevent that the result depends on the order in which we add or remove relations. We initialize R' to the set of tentative relations T, and immediately subtract the set of forbidden relations from R'. Subsequently, we iteratively perform a composition of R' with itself, denoted as $R' \circ R'$. Such a composition at first possibly includes the induction of relations present in F. After each composition step, we subtract F from R', so as to not use relations from F when composing R' with itself in the next step – the induced relations present in F are thus removed in this step. Naturally, these relations will be induced once more in the next composition step. Nevertheless, this iterative process will reach a status quo: at one point, we will induce a set of relations which is a subset (possibly empty) of F. We know now which forbidden relations can be induced from T. We copy R' to R. By subtracting all paths in R' inducing such a relation from R, except for those relations which are necessary (present in N), we finally end up with the desired partial order R. For completeness' sake, we also mention we use an intermediate F', to keep intact $\{T,N,F\}$.

Algorithm 2: Extraction of R from $\{T,N,F\}$

Data: Set of $\{T,N,F\}$ relations
Result: Consistent union R
$R' \leftarrow T \setminus F$;
while $(R' \circ R') \cap F^{-1} \neq \emptyset$ **do**
| $R' \leftarrow (R' \circ R') \setminus F$;
end
$R \leftarrow R'$ $F' \leftarrow F$ **foreach** $\{(a,c) \in F' \mid (a,b) \wedge (b,c) \in R'\}$ **do**
| $F' \leftarrow F' \cup \{(a,b),(b,c)\}$;
| $F' \leftarrow F' \setminus N$;
end
$R \leftarrow R \setminus F'$;

Observe that the way in which R is extracted does not depend on how the set of $\{T,N,F\}$ relations was constructed. Nevertheless, care must be taken when constructing the set of $\{T,N,F\}$ relations: for arbitrary relations T, N and F, we should not expect R to be a partial order. We describe different ways to combine two $\{T,N,F\}$ relations in Sections 4 and 5.

We briefly detail why it is advantageous to use a set of $\{T,N,F\}$ relations. Both the intersection and the union of two posets contain less information than the two posets themselves: information on which relations are present in only one poset, and which are present in both, has been lost. The same is true for our relation R extracted from a set of $\{T,N,F\}$ relations: R contains less information than $\{T,N,F\}$.

Suppose now we have a set $\mathcal{P}$ of partial order relations $\{P_1,\ldots,P_n\}$, and process this set in two different ways. The first is a stepwise construction of R, abandoning the residual information present in $\{T,N,F\}$ after each step. The second constructs a single $\{T,N,F\}$ structure on the basis of $\mathcal{P}$, and extracts an R from this.

We should not expect both approaches to yield the same outcome in general (we will return to this in Section 6 in more detail). In other words, in order to easily and accurately extend the consistent union operation from Rademaker et al. (2008) to an arbitrary number of input posets, we need to have a representation which does not result in a loss of information. To this end, we introduced the triplet of relations $\{T,N,F\}$.

4. Implementation of the consistent union operation

Our previous definition of the consistent union catered to the special case of two input partially ordered sets, see Rademaker et al. (2008). We now extend this to an arbitrary number of sets. As the operation to extract R from a $\{T,N,F\}$ set does not depend on the origin of $\{T,N,F\}$, the consistent union operation will have to be a way of combining two $\{T,N,F\}$ structures (correlating to a number of posets). As the intricate part is the extraction of R from $\{T,N,F\}$ (Algorithm 2), the consistent union operation can be very simple.

Combine two sets of $\{T,N,F\}$ relations as follows: take the union of the two T relations, the intersection of the two N relations, and the union of the two F relations. For completeness' sake, we provide Algorithm 3.

Algorithm 3: Non-priority combination of two sets of $\{T,N,F\}$ relations

Data: Two sets of $\{T,N,F\}$ relations, $\{T_1,N_1,F_1\}$ and $\{T_2,N_2,F_2\}$
Result: Set of $\{T,N,F\}$ relations
$T \leftarrow T_1 \cup T_2$;
$N \leftarrow N_1 \cap N_2$;
$F \leftarrow F_1 \cup F_2$;

As an aside, suppose now $\{T_1,N_1,F_1\}$ and $\{T_2,N_2,F_2\}$ have been constructed on the basis of two partial orders P_1 and P_2. The partial order relation R extracted as per Algorithm 2 from the set of relations $\{T,N,F\}$ constructed as per Algorithm 3, will correspond to the consistent union of P_1 and P_2 from Rademaker et al. (2008).

5. Implementation of the prioritized union operation

Rather than requiring a new way to extract a relation R from a set of $\{T,N,F\}$ relations, the prioritized consistent union operation will amount to a new way to construct a set of $\{T,N,F\}$ relations. We consider the case where we have a priority set of tentative, necessary and forbidden relations, denoted $\{T_1,N_1,F_1\}$, and a non-priority set $\{T_2,N_2,F_2\}$.

Two options exist, depending on how much heed we want to pay to $\{T_2,N_2,F_2\}$. Suppose we extract R from $\{T,N,F\}$ constructed on the basis of some prioritized union of $\{T_1,N_1,F_1\}$ and $\{T_2,N_2,F_2\}$. Clearly, we should have $R\cap F_1 = \emptyset$. It is less clear whether we should also demand $R\cap (F_2\setminus T_1) = \emptyset$, or rather allow $R\cap (F_2\setminus T_1) \neq \emptyset$. We will formulate algorithms for both options: a strong prioritized union operation, allowing $R\cap (F_2\setminus T_1) \neq \emptyset$, and a weak prioritized union operation, demanding $R\cap (F_2\setminus T_1) = \emptyset$. Demanding $R\cap (F_2\setminus T_1) = \emptyset$ (rather than allowing $R\cap (F_2\setminus T_1) \neq \emptyset$) yields the weaker operation, as it implies an increased importance of the non-priority $\{T,N,F\}$ set, which must come at the expense of the importance of the priority $\{T,N,F\}$ set. We show that, in order to make sure R is able to yield a partial order, we need to demand $R\cap (F_2\setminus T_1) = \emptyset$. Hence, we advocate the use of the weak prioritized union operation.

5.1. The strong prioritized union operation

The R extracted from the $\{T,N,F\}$ structure when constructed on the basis of two partial orders P_1 and P_2, will be equal to the R yielded by prioritized union operation from Rademaker et al. (2008). We extend it here for the prioritized combination of two $\{T,N,F\}$ sets, and (erroneously, as we will show) count on Algorithm 2 to yield a partial order. Algorithm 4 is again very simple. Observe how the non-priority poset only adds relations to T, and not to F.

Algorithm 4: Strong prioritized combination of two sets of $\{T,N,F\}$ relations

Data: Two sets of $\{T,N,F\}$ relations, the priority set $\{T_1,N_1,F_1\}$ and the non-priority set $\{T_2,N_2,F_2\}$
Result: Set of $\{T,N,F\}$ relations
$T \leftarrow T_1\cup(T_2\setminus F_1)$;
$N \leftarrow N_1$;
$F \leftarrow F_1$;

We now show Algorithm 2 is able to yield a cycle for a certain combination of $\{T_1,N_1,F_1\}$ and $\{T_2,N_2,F_2\}$. Suppose $T_1 = N_1 = F_1 = \emptyset$, and $T_2 = \{(a,b),(b,a)\}$, $N_2 = \emptyset$ and $F_2 = \{(a,b),(b,a)\}$. Algorithm 4 yields the following: $T = \{(a,b),(b,a)\}$, $N = F = \emptyset$. Applying Algorithm 2 to this $\{T,N,F\}$ structure, will result in the relation

$R\{(a,b),(b,a)\}$, as neither (a,b) nor (b,a) contradicts F_1. To be more exact, cycles can arise between objects for which the priority $\{T,N,F\}$ contains no information regarding their pairwise order. As we had mentioned in Section 3, $\{T,N,F\}$ needs to fulfill some (unspecified) properties in order for Algorithm 2 to yield a partial order. Apparently, the strong prioritized union operator does not fulfill these properties. We will formulate the weak prioritized union operation to prevent these problems from arising.

Finally, we would like to stress that the $\{T_2,N_2,F_2\}$ used in this example, is not a partial order. As such, the operation described in Rademaker et al. (2008), of which Algorithm 4 is an extension, will allow extraction of a partial order for the case when $\{T_1,N_1,F_1\}$ and $\{T_2,N_2,F_2\}$ both have been constructed on the basis of a single (different) partial order.

5.2. The weak prioritized union operation

We now make a slight adjustment to Algorithm 4. Rather than limiting the influence of the non-priority $\{T,N,F\}$ to the possible extension of T_1 by a subset of T_2, we now also allow the extension of F_1 by a subset of F_2 in Algorithm 5. We again count on Algorithm 2 to yield a partial order. The absence of cycles is guaranteed as now the inverse of at least one relation involved in the cycle will be present in F.

Algorithm 5: Weak prioritized combination of two sets of $\{T,N,F\}$ relations

---

Data: Two sets of $\{T,N,F\}$ relations, the priority set $\{T_1,N_1,F_1\}$ and the non-priority set $\{T_2,N_2,F_2\}$

Result: Set of $\{T,N,F\}$ relations

$T \leftarrow T_1 \cup (T_2 \setminus F_1)$;

$N \leftarrow N_1$;

$F \leftarrow F_1 \cup (F_2 \setminus T_1)$;

We return to the example that showed possible problems inherent to using Algorithm 4 for triples of $\{T,N,F\}$ that were not constructed on the basis of a single poset. For $T_1 = N_1 = F_1 = \emptyset$, and $T_2 = \{(a,b),(b,a)\}$, $N_2 = \emptyset$ and $F_2 = \{(a,b),(b,a)\}$, Algorithm 5 will yield $T = \{(a,b),(b,a)\}$, $N = \emptyset$, $F = \{(a,b),(b,a)\}$. Clearly, when extracting an R from this $\{T,N,F\}$ structure, the outcome will be the empty set.

We also provide an example to show how a single relation present in the priority set of $\{T,N,F\}$ relations, can prevent a cycle from occurring for both Algorithm 4 and Algorithm 5. Suppose $T_1 = N_1 = \{(a,b)\}$ and $F_1 = \{(b,a)\}$, and let $T_2 = \{(b,c),(c,a)\}$, $N_2 = \emptyset$ and $F_2 = \{(a,c),(c,b)\}$. The weak prioritized union (Algorithm 5) yields $T = \{(a,b),(b,c),(c,a)\}$, $N = \{(a,b)\}$ and $F = \{(a,c),(b,a),(c,b)\}$. Algorithm 2 will then yield an $R = \{(a,b)\}$, as accepting $\{(a,b),(b,c),(c,a)\}$ into R would yield a cycle containing at least one of $F = \{(a,c),(b,a),(c,b)\}$, while accepting $\{(a,b),(b,c)\}$ would also have a non-empty intersection with F. The strong prioritized union (Algorithm 4) yields $T = \{(a,b),(b,c),(c,a)\}$, $N = \{(a,b)\}$ and $F = \{(b,a)\}$. Algorithm 2 would then first see the arising of the cycle $\{(a,b),(b,c),(c,a)\}$, which will contain $F = \{(b,a)\}$, and lead to each path being cut, preserving only the necessary relation (a,b), which will be the final output.

Finally, we mention that we feel it is very probable the weak and strict prioritized union operations will be one and the same when taking two partially ordered sets as input. However, fully exploring this possibility falls outside the scope of this paper.

6. Properties of $\{T,N,F\}$ construction operations

We now outline some desired properties of $\{T,N,F\}$ constructing operations. We will contrast the use of the $\{T,N,F\}$ structure to the alternative of constructing intermediate R relations, such as would be necessary when supplied with an operation only capable of processing two partial orders. The behavior of such an operation when processing a set $\mathcal{P}$ of partial orders $\{P_1,\ldots,P_n\}$ can be emulated as follows: Construct $\{T_1,N_1,F_1\}$ and $\{T_2,N_2,F_2\}$ on the basis of P_1 and P_2, combine them into $\{T_{1,2},N_{1,2},F_{1,2}\}$ and extract R. From now on, we will write $\{T,N,F\}_{1,2}$, rather than $\{T_{1,2},N_{1,2},F_{1,2}\}$, to make for easier reading. Construct $\{T,N,F\}_R$ on the basis of R, and combine it with $\{T,N,F\}_3$ to yield $\{T,N,F\}_{R,3}$. Extract a new R from $\{T,N,F\}_{R,3}$, combine it with $\{T,N,F\}_4$ and so on. We will show this method yields very different results from the approach we advocate.

Constructing a set of relations $\{T,N,F\}$ on the basis of a set $\mathcal{P}$ of partial orders $\{P_1,\ldots,P_n\}$, denoted as $\{T,N,F\}_{\mathcal{P}}$, should be insensitive to the inclusion of multiple identical partial orders. In other words, suppose $\mathcal{P}' \subseteq \mathcal{P}$, it would be natural to have $\{T,N,F\}_{\mathcal{P}} = \{T,N,F\}_{\mathcal{P}\cup\mathcal{P}'}$. When using an operation that does not preserve $\{T,N,F\}$ as an intermediate result, it will be clear that this property does not hold. Suppose $P_1 = \{(a,b)\}$ and $P_2 = \{(b,a)\}$. Clearly, the consistent union (denote it as $R_{1,2}$) will be the empty set. If we now take the consistent union of this intermediate

$R_{1,2}$ and P_2, the outcome becomes $R_{1,2,2} = \{(b,a)\}$. When processing the partial orders in the inverse order, the outcome ($R_{2,2,1}$) would be the empty set. Clearly, this method is sensitive to the repetition of partial orders.

The example discussed above also shows that we should not expect the consistent union of three different partial orders to be independent of the order in which they are processed when using an operation which outputs only an R relation without conserving a $\{T,N,F\}$ structure. When keeping the $\{T,N,F\}$ structure as an intermediate result however, sensitivity to the order of the processed posets does not occur. We provide an illustrative example on the basis of $P_1 = \{(a,b)\}$ and $P_2 = \{(b,a)\}$. We have $T_1 = N_1 = \{(a,b)\}$, $F_1 = \{(b,a)\}$ and $T_2 = N_2 = \{(b,a)\}$, $F_2 = \{(a,b)\}$. Combining these via Algorithm 3 yields $T = \{(a,b),(b,a)\}$, $N = \emptyset$ and $F = \{(a,b),(b,a)\}$. Clearly, the number of additional times P_1 or P_2 would be added to this $\{T,N,F\}$ structure will not matter, nor will the order, in which they were to be added. As the simplicity of Algorithm 3 allows for an easy verification of these properties, we will not discuss this any further.

Finally, we show that we have the pleasing property that for two $\{T,N,F\}$ structures $\{T_1,N_1,F_1\}$ and $\{T_2,N_2,F_2\}$, the consistent union of both structures will be equal to the consistent union of the weak prioritized unions when taking each structure in turn as the priority structure. Algorithm 3 immediately contains the outcome of the consistent union of the two structures. We now let $\{T,N,F\}_{2|1}$ denote the weak prioritized union taking $\{T_1,N_1,F_1\}$ as the priority structure, and let $\{T,N,F\}_{1|2}$ denote the weak prioritized union taking $\{T_2,N_2,F_2\}$ to be the priority structure. As per Algorithm 5, this immediately yields

$$\begin{aligned} T_{2|1} &= T_1 \cup (T_2 \setminus F_1) \\ N_{2|1} &= N_1 \\ F_{2|1} &= F_1 \cup (F_2 \setminus T_1) \ . \end{aligned}$$

For $\{T,N,F\}_{1|2}$, the relations are analogous. Observe now that $T_1 \subseteq T_{2|1} \subseteq T_1 \cup T_2$, and likewise $T_2 \subseteq T_{1|2} \subseteq T_1 \cup T_2$, while $F_1 \subseteq F_{2|1} \subseteq F_1 \cup F_2$ and likewise $F_2 \subseteq F_{1|2} \subseteq F_1 \cup F_2$. Applying Algorithm 1 on $\{T,N,F\}_{1|2}$ and $\{T,N,F\}_{2|1}$ then yields for $T = T_{2|1} \cup T_{1|2}$, which will be equal to $T_1 \cup T_2$ precisely because $T_{2|1}$ and $T_{1|2}$ are extensions of the priority set of tentative relations of which the union is in turn an extension. Consequently, the union of $T_{2|1}$ and $T_{1|2}$ must then be equal to the union of T_1 and T_2. The same holds for the set of forbidden relations F, while the set of necessary relations is immediately defined as the intersection of both sets of necessary relations. Thus, we are able to conclude that the consistent union of $\{T_1, N_1, F_1\}$ and $\{T_2, N_2, F_2\}$, is equal to the consistent union of $\{T, N, F\}_{2|1}$ and $\{T, N, F\}_{1|2}$.

7. Summary

We have extended our previous operations described in Rademaker et al. (2008) to be able to process multiple partial orders. To this end, we needed to introduce a new structure of a triplet of relations, to be able to conserve the necessary information when dealing with processing more than two partial order relations. Both the consistent union and the prioritized union operation have been extended in this way. Extending the prioritized union to multiple partial orders identified a possible variant, with only one yielding a unique partial order. We hope the clear and easily implementable algorithms detailed in this text will allow these operations to be disseminated in more data-exploratory papers.

References

De Baets B. and De Meyer H. (2003) On the existence and construction of T-transitive closures. *Information Sciences*, 152, 167-179.

Floyd R.W. (1962) Algorithm 97: Shortest path. *Communications of the ACM*, 5, 345.

Naessens H. et al. (2002) Algorithms for the computation of T-transitive closures. *IEEE Transactions on Fuzzy Systems*, 10, 541-551.

Rademaker M. et al. (2008) New operations for informative combination of two partial order relations with illustrations on pollution data. *Computational Chemistry and High Throughput Screening*, 11, 745-755.

Warshall S. (1962) A theorem on boolean matrices. *Journal of the ACM*, 9,11-12.

This book is a collection of papers, prepared in connection with the 8th International Workshop on partial orders, their theoretical and applied developments, which took place in Warsaw, at the Systems Research Institute, in October 2008. The papers deal with software developments (PYHASSE and other existing software), theoretical problems of ranking and ordering under various assumed analytic and decision-making-oriented conditions, as well as experimental studies and down-to-earth pragmatic questions.

ISBN 83-894-7521-9
EAN 9788389475213