How to Tell When a Product of Two Partially Ordered Spaces Has a Certain Property?

Francisco Zapata, Olga Kosheleva
University of Texas at El Paso
500 W. University
El Paso, TX 79968, USA
fazg74@gmail.com, olgak@utep.edu

Karen Villaverde
Department of Computer Science
New Mexico State University
Las Cruces, New Mexico 88003, USA
Email: kvillave@cs.nmsu.edu

1. Fuzzy Logic: Brief Reminder

- In the traditional 2-valued logic, every statement is either true or false.
- Thus, the set of possible truth values consists of two elements: true (1) and false (0).
- Fuzzy logic takes into account that people have different degrees of certainty in their statements.
- Traditionally, fuzzy logic uses values from the interval $[0,1]$ to describe uncertainty.
- In this interval, the order is *total* (*linear*) in the sense that for every $a, a' \in [0,1]$, either $a \le a'$ or $a' \le a$.
- However, often, *partial* orders provide a more adequate description of the expert's degree of confidence.

2. Towards General Partial Orders

- For example, an expert cannot describe her degree of certainty by an exact number.
- Thus, it makes sense to describe this degree by an *interval* $[\underline{d}, \overline{d}]$ of possible numbers.
- Intervals are only partially ordered; e.g., the intervals $[0.5, 0.5]$ and $[0, 1]$ are not easy to compare.
- More complex sets of possible degrees are also sometimes useful.
- Not to miss any new options, in this paper, we consider general partially ordered spaces.

3. Need for Product Operations

- Often, two (or more) experts evaluate a statement S.
- Then, our certainty in S is described by a pair (a_1, a_2), where $a_i \in A_i$ is the i-th expert's degree of certainty.
- To compare such pairs, we must therefore define a partial order on the set $A_1 \times A_2$ of all such pairs.
- One example of a partial order on $A_1 \times A_2$ is a *Cartesian* product: $(a_1, a_2) \leq (a'_1, a'_2) \Leftrightarrow ((a_1 \leq a'_1) \,\&\, (a_2 \leq a'_2))$.
- This is a *cautious* approach, when our confidence in S' is higher than in $S \Leftrightarrow$ it is higher for both experts.
- *Lexicographic* product: $(a_1, a_2) \leq (a'_1, a'_2) \Leftrightarrow$

$$((a_1 \leq a'_1) \,\&\, a_1 \neq a'_1) \vee ((a_1 = a'_1) \,\&\, (a_2 \leq a'_2))).$$

- Here, we are absolutely confident in the 1st expert – and only use the 2nd when the 1st is not sure.

4. Natural Questions

- *Question:* when does the resulting partially ordered set $A_1 \times A_2$ satisfy a certain property?
- *Examples:* is it a total order? is it a lattice order?
- *It is desirable* to reduce the question about $A_1 \times A_2$ to questions about properties of component spaces A_i.
- *Some such reductions are known*; e.g.:
  - A Cartesian product is a total order $\Leftrightarrow$ one of A_i is a total order, and the other has only one element.
  - A lexicographic product is a total order if and only if both components are totally ordered.
- *In this paper*, we provide a general algorithm for such reduction.

5. Similar Questions in Other Areas

- Similar questions arise in *other applications* of ordered sets.
- *Example:* in space-time geometry, $a \leq b$ means that an event a can influence the event b.
- *Our algorithm* does not use the fact that the original relations are orders.
- Thus, our algorithm is applicable to a *general* binary *relation* – equivalence, similarity, etc.
- Moreover, this algorithm can be applied to the case when we have a space with *several* binary relations.
- *Example:* we may have an order relation and a similarity relation.

6. Definitions

- *By a* space, *we mean a set A with m binary relations $P_1(a, a'), \ldots, P_m(a, a')$.*
- *By a* 1st order property, *we mean a formula F obtained from $P_i(x, x')$ by using logical $\vee$, $\&$, $\neg$, $\rightarrow$, $\exists x$ and $\forall x$.*
- *Note:* most properties of interest are 1st order; e.g. to be a total order means $\forall a \forall a' \, ((a \leq a') \vee (a' \leq a))$.
- *By a* product operation, *we mean a collection of m propositional formulas that*
  - *describe the relation $P_i((a_1, a_2), (a'_1, a'_2))$ between the elements $(a_1, a_2), (a'_1, a'_2) \in A_1 \times A_2$*
  - *in terms of the relations between the components $a_1, a'_1 \in A_1$ and $a_2, a'_2 \in A_2$ of these elements.*
- *Note:* both Cartesian and lexicographic order are product operations in this sense.

7. Main Result

- **Main Result.** *There exists an algorithm that, given*
  - *a product operation and*
  - *a property F,*

  generates a list of properties F_{11}, F_{12}, ..., F_{p1}, F_{p2} s.t.:

$$F(A_1 \times A_2) \Leftrightarrow ((F_{11}(A_1) \,\&\, F_{12}(A_2)) \vee \ldots \vee (F_{p1}(A_1) \,\&\, F_{p2}(A_2))).$$

- *Example:* For Cartesian product and total order F, we have

$$F(A_1 \times A_2) \Leftrightarrow ((F_{11}(A_1) \,\&\, F_{12}(A_2)) \vee (F_{21}(A_1) \,\&\, F_{22}(A_2))):$$

  - $F_{11}(A_1)$ means that A_1 is a total order,
  - $F_{12}(A_2)$ means that A_2 is a one-element set,
  - $F_{21}(A_1)$ means that A_1 is a one-element set, and
  - $F_{22}(A_2)$ means that A_2 is a total order.

8. Auxiliary Results

- *Generalization:*
  - A similar algorithm can be formulated for a product of three or more spaces.
  - A similar algorithm can be formulated for the case when we allow ternary and higher order operations.
- *Specifically for partial orders:*
  - The only product operations that always leads to a partial order on $A_1 \times A_2$ for which

    $$(a_1 \leq_1 a_1' \,\&\, a_2 \leq_2 a_2') \rightarrow (a_1, a_2) \leq (a_1', a_2')$$

    are Cartesian and lexicographic products.

9. Proof of the Main Result

- The desired property $F(A_1 \times A_2)$ uses:
  - relations $P_i(a, a')$ between elements $a, a' \in A_1 \times A_2$;
  - quantifiers $\forall a$ and $\exists a$ over elements $a \in A_1 \times A_2$.
- Every element $a \in A_1 \times A_2$ is, by definition, a pair (a_1, a_2) in which $a_1 \in A_1$ and $a_2 \in A_2$.
- Let us explicitly replace each variable with such a pair.
- By definition of a product operation:
  - each relation $P_i((a_1, a_2), (a_1', a_2'))$
  - is a propositional combination of relations betw. elements $a_1, a_1' \in A_1$ and betw. elements $a_2, a_2' \in A_2$.
- Let us perform the corresponding replacement.
- Each quantifier can be replaced by quantifiers corresponding to components: e.g., $\forall (a_1, a_2) \Leftrightarrow \forall a_1 \forall a_2$.

10. Proof of the Main Result (cont-d)

- So, we get an equivalent reformulation of F s.t.:
  - elementary formulas are relations between elements of A_1 or between A_2, and
  - quantifiers are over A_1 or over A_2.
- We use induction to reduce to the desired form

$$((F_{11}(A_1) \mathbin{\&} F_{12}(A_2)) \vee \ldots \vee (F_{p1}(A_1) \mathbin{\&} F_{p2}(A_2))).$$

- Elementary formulas are already of the desired form – provided, of course, that we allow free variables.
- We will show that:
  - if we apply a propositional connective or a quantifier to a formula of this type,
  - then we can reduce the result again to the formula of this type.

11. Applying Propositional Connectives

- We apply propositional connectives to formula of the type
$$((F_{11}(A_1)\,\&\,F_{12}(A_2))\vee\ldots\vee(F_{p1}(A_1)\,\&\,F_{p2}(A_2))).$$
- We thus get a propositional combination of the formulas of the type $F_{ij}(A_j)$.
- An arbitrary propositional combination can be described as a disjunction of conjunctions (DNF form).
- Each conjunction combines properties related to A_1 and properties related to A_2, i.e., has the form
$$G_1(A_1)\,\&\,\ldots\,\&\,G_p(A_1)\,\&\,G_{p+1}(A_2)\,\&\ldots\,\&\,G_q(A_2).$$
- Thus, each conjunction has the from $G(A_1)\,\&\,G'(A_2)$, where $G(A_1)\Leftrightarrow(G_1(A_1)\,\&\,\ldots\,\&\,G_p(A_1))$.
- Thus, the disjunction of such properties has the desired form.

12. Applying Existential Quantifiers

- When we apply $\exists a_1$, we get a formula
  $\exists a_1\,((F_{11}(A_1)\,\&\,F_{12}(A_2)) \vee \ldots \vee (F_{p1}(A_1)\,\&\,F_{p2}(A_2)))$.
- It is known that $\exists a\,(A \vee B)$ is equivalent to $\exists a\,A \vee \exists a\,B$.
- Thus, the above formula is equivalent to a disjunction
  $\exists a_1\,(F_{11}(A_1)\,\&\,F_{12}(A_2)) \vee \ldots \vee \exists a_1\,(F_{p1}(A_1)\,\&\,F_{p2}(A_2))$.
- Thus, it is sufficient to prove that each formula $\exists a_1\,(F_{i1}(A_1)\,\&\,F_{i2}(A_2))$ has the desired form.
- The term $F_{i2}(A_2)$ does not depend on a_1 at all, it is all about elements of A_2.
- Thus, the above formula is equivalent to

$$(\exists a_1\,F_{i1}(A_1))\,\&\,F_{i2}(A_2).$$

- So, it is equivalent to the formula $F'_{i1}(A_1)\,\&\,F_{i2}(A_2)$, where $F'_{i1} \Leftrightarrow \exists a_1\,F_{i1}(A_1)$.

13. Applying Universal Quantifiers

- When we apply a universal quantifier, e.g., $\forall a_1$, then we can use the fact that $\forall a_1\, F$ is equivalent to $\neg\exists a_1\, \neg F$.
- We assumed that the formula F is of the desired type

$$(F_{11}(A_1)\,\&\,F_{12}(A_2)) \vee \ldots \vee (F_{p1}(A_1)\,\&\,F_{p2}(A_2)).$$

- By using the propositional part of this proof, we conclude that $\neg F$ can be reduced to the desired type.
- Now, by applying the $\exists$ part of this proof, we conclude that $\exists a_1\,(\neg F)$ can also be reduced to the desired type.
- By using the propositional part again, we conclude that $\neg(\exists a_1\, \neg F)$ can be reduced to the desired type.
- By induction, we can now conclude that the original formula can be reduced to the desired type.
- The main result is proven.

14. Example of Applying the Algorithm

- Let us apply our algorithm to checking whether a Cartesian product is totally ordered.
- In this case, F has the form $\forall a \forall a' \, ((a \le a') \vee (a' \le a))$.
- We first replace each variable $a, a' \in A_1 \times A_2$ with the corresponding pair:

$$\forall (a_1, a_2) \forall (a'_1, a'_2) \, (((a_1, a_2) \le (a'_1, a'_2)) \vee ((a'_1, a'_2) \le (a_1, a_2))).$$

- Replacing the ordering relation on the Cartesian product with its definition, we get

$$\forall (a_1, a_2) \forall (a'_1, a'_2) \, ((a_1 \le a'_1 \,\&\, a_2 \le a'_2) \vee (a'_1 \le a_1 \,\&\, a'_2 \le a_2)).$$

- Replacing $\forall a$ over pairs with individual $\forall a_i$, we get:

$$\forall a_1 \forall a_2 \forall a'_1 \forall a'_2 \, ((a_1 \le a'_1 \,\&\, a_2 \le a'_2)) \vee ((a'_1 \le a_1 \,\&\, a'_2 \le a_2))).$$

- By using the $\forall \Leftrightarrow \neg \exists \neg$, we get an equivalent form

$$\neg \exists a_1 \exists a_2 \exists a'_1 \exists a'_2 \, \neg((a_1 \le a'_1 \,\&\, a_2 \le a'_2) \vee (a'_1 \le a_1 \,\&\, a'_2 \le a_2))).$$

15. Example (cont-d)

- So far, we got:

$$\neg\exists a_1\exists a_2\exists a_1'\exists a_2'\,\neg((a_1\le a_1'\,\&\,a_2\le a_2')\vee(a_1'\le a_1\,\&\,a_2'\le a_2))).$$

- Moving $\neg$ inside the propositional formula, we get

$$\neg\exists a_1\exists a_1\exists a_1'\exists a_2'\,((a_1\not\le a_1'\vee a_2\not\le a_2')\,\&\,(a_1'\not\le a_1\vee a_2'\le a_2))).$$

- The formula $(a_1\not\le a_1'\vee a_2\not\le a_2'))\,\&\,(a_1'\not\le a_1\vee a_2'\not\le a_2)$ must now be transformed into a DNF form.

- The result is $(a_1\not\le a_1'\,\&\,a_1'\not\le a_1)\vee(a_1\not\le a_1'\,\&\,a_2'\not\le a_2)\vee$ $(a_2\not\le a_2'\,\&\,a_1'\not\le a_1)\vee(a_2\not\le a_2'\,\&\,a_2'\not\le a_2)$.

- Thus, our formula is $\Leftrightarrow\neg(F_1\vee F_2\vee F_3\vee F_4)$, where

$$F_1\Leftrightarrow\exists a_1\exists a_2\exists a_1'\exists a_2'\,(a_1\not\le a_1'\,\&\,a_1'\not\le a_1),$$

$$F_2\Leftrightarrow\exists a_1\exists a_2\exists a_1'\exists a_2'\,(a_1\not\le a_1'\,\&\,a_2'\not\le a_2),$$

$$F_3\Leftrightarrow\exists a_1\exists a_2\exists a_1'\exists a_2'\,(a_2\not\le a_2'\,\&\,a_1'\not\le a_1),$$

$$F_4\Leftrightarrow\exists a_1\exists a_2\exists a_1'\exists a_2'\,(a_2\not\le a_2'\,\&\,a_2'\not\le a_2).$$

16. Example (cont-d)

- So far, we got $\Leftrightarrow \neg(F_1 \vee F_2 \vee F_3 \vee F_4)$, where

$$F_1 \Leftrightarrow \exists a_1 \exists a_2 \exists a_1' \exists a_2' \, (a_1 \not\leq a_1' \,\&\, a_1' \not\leq a_1),$$

$$F_2 \Leftrightarrow \exists a_1 \exists a_2 \exists a_1' \exists a_2' \, (a_1 \not\leq a_1' \,\&\, a_2' \not\leq a_2),$$

$$F_3 \Leftrightarrow \exists a_1 \exists a_2 \exists a_1' \exists a_2' \, (a_2 \not\leq a_2' \,\&\, a_1' \not\leq a_1),$$

$$F_4 \Leftrightarrow \exists a_1 \exists a_2 \exists a_1' \exists a_2' \, (a_2 \not\leq a_2' \,\&\, a_2' \not\leq a_2).$$

- By applying the quantifiers to the corresponding parts of the formulas, we get

$$F_1 \Leftrightarrow \exists a_1 \exists a_1' \, (a_1 \not\leq a_1' \,\&\, a_1' \not\leq a_1),$$

$$F_2 \Leftrightarrow (\exists a_1 \exists a_1' \, a_1 \not\leq a_1') \,\&\, (\exists a_2 \exists a_2' \, a_2' \not\leq a_2),$$

$$F_3 \Leftrightarrow (\exists a_1 \exists a_1' \, a_1' \not\leq a_1) \,\&\, (\exists a_2 \exists a_2' \, a_2 \not\leq a_2'),$$

$$F_4 \Leftrightarrow \exists a_2 \exists a_1' \exists a_2' \, (a_2 \not\leq a_2' \,\&\, a_2' \not\leq a_2).$$

- Then, we again reduce $\neg(F_1 \vee F_2 \vee F_3 \vee F_4)$ to DNF.