Products in the category of forests and p-morphisms via Delannoy paths on Cartesian products

Pietro Codara

Dipartimento di Informatica, Università degli Studi di Milano

(joint work with Ottavio M. D’Antona, and Vincenzo Marra)

TACL 2015, Ischia (NA) — June 25, 2015
Basic notions.
A category of forests

- A **forest** is a finite poset $F$ such that for every $x \in F$, $\downarrow x$ is a chain. A **tree** is a forest with a bottom element.
A category of forests

- A **forest** is a finite poset $F$ such that for every $x \in F$, $\downarrow x$ is a chain. A **tree** is a forest with a bottom element.

- An order preserving map $f : F \to G$ is a **p-morphism** (or is **open**) iff, for every $x \in F$,

  $$f(\downarrow x) = \downarrow f(x).$$
A forest is a finite poset $F$ such that for every $x \in F$, $\downarrow x$ is a chain. A tree is a forest with a bottom element.

An order preserving map $f : F \to G$ is a p-morphism (or is open) iff, for every $x \in F$,

$$f(\downarrow x) = \downarrow f(x).$$
In this talk.
In this talk

We show how to compute products in the category $F$ of forests and p-morphisms.
We show how to compute products in the category $F$ of forests and $p$-morphisms.

- Dually, via Esakia duality, we show how to compute co-products of finitely presented Gödel algebras.
In this talk

We show how to compute products in the category $\mathcal{F}$ of forests and $p$-morphisms.

- Dually, via Esakia duality, we show how to compute co-products of finitely presented Gödel algebras.

Various techniques to perform this computation are known. Why should the one presented here be interesting?
We show how to compute products in the category $F$ of forests and $p$-morphisms.

- Dually, via Esakia duality, we show how to compute coproducts of finitely presented Gödel algebras.

Various techniques to perform this computation are known. Why should the one presented here be interesting?

- In the category $F$ products are not Cartesian.
- Our construction is “as Cartesian as possible”.
Product of forests,
known combinatorial methods.
Ordered partitions, and merged shuffles

An ordered partition $\sigma$ is a sequence of pairwise disjoint nonempty sets, called blocks. The union of the blocks of $\sigma$ is the support of $\sigma$. 

Ordered partitions, and merged shuffles


- An **ordered partition** $\sigma$ is a sequence of pairwise disjoint nonempty sets, called blocks. The union of the blocks of $\sigma$ is the support of $\sigma$.

- Let $\sigma$ and $\tau$ be ordered partitions with disjoint supports. An ordered partition $\theta$ is a **shuffle** of $\sigma$ and $\tau$ iff $\sigma$ and $\tau$ are subsequences of $\theta$, and $\text{supp}\theta = \text{supp}\sigma \cup \text{supp}\tau$. 
Ordered partitions, and merged shuffles


- An ordered partition $\sigma$ is a sequence of pairwise disjoint nonempty sets, called blocks. The union of the blocks of $\sigma$ is the support of $\sigma$.
- Let $\sigma$ and $\tau$ be ordered partitions with disjoint supports. An ordered partition $\theta$ is a shuffle of $\sigma$ and $\tau$ iff $\sigma$ and $\tau$ are subsequences of $\theta$, and $\text{supp}\theta = \text{supp}\sigma \cup \text{supp}\tau$.
- A merged shuffle is obtained from a shuffle $\theta$, by merging some consecutive pairs of blocks $A, B \in \theta$, with $A \in \sigma$, and $B \in \tau$. 
Ordered partitions, and merged shuffles


- An **ordered partition** $\sigma$ is a sequence of pairwise disjoint nonempty sets, called blocks. The union of the blocks of $\sigma$ is the support of $\sigma$.

- Let $\sigma$ and $\tau$ be ordered partitions with disjoint supports. An ordered partition $\theta$ is a **shuffle** of $\sigma$ and $\tau$ iff $\sigma$ and $\tau$ are subsequences of $\theta$, and $\text{supp}\theta = \text{supp}\sigma \cup \text{supp}\tau$.

- A **merged shuffle** is obtained from a shuffle $\theta$, by merging some consecutive pairs of blocks $A, B \in \theta$, with $A \in \sigma$, and $B \in \tau$.

**Example.** Let $\sigma = \{ a|b \}$ and $\tau = \{ x \}$. The merged shuffles of $\sigma$ and $\tau$ are: $\{ a|b|x \}, \{ a|x|b \}, \{ x|a|b \}, \{ a|bx \}, \{ ax|b \}$.
Trees of ordered partitions

Given ordered partitions $\sigma = \{A_1|\ldots|A_m\}$, and $\tau = \{B_1|\ldots|B_n\}$ with $m \leq n$ we write $\sigma \leq \tau$ iff $A_i = B_i$ for every $i \in \{1, \ldots, m\}$. 
Trees of ordered partitions

Given ordered partitions $\sigma = \{A_1|\ldots|A_m\}$, and $\tau = \{B_1|\ldots|B_n\}$ with $m \leq n$ we write $\sigma \leq \tau$ iff $A_i = B_i$ for every $i \in \{1,\ldots,m\}$.

One can label trees with ordered partitions...
Trees of ordered partitions

Given ordered partitions $\sigma = \{A_1|...|A_m\}$, and $\tau = \{B_1|...|B_n\}$ with $m \leq n$ we write $\sigma \leq \tau$ iff $A_i = B_i$ for every $i \in \{1, \ldots, m\}$. One can label trees with ordered partitions...
Trees of ordered partitions

Given ordered partitions $\sigma = \{A_1|\ldots|A_m\}$, and $\tau = \{B_1|\ldots|B_n\}$ with $m \leq n$ we write $\sigma \leq \tau$ iff $A_i = B_i$ for every $i \in \{1,\ldots,m\}$.

One can label trees with ordered partitions...

One can build a tree from a set of ordered partitions. The tree of merged shuffles of $\sigma = \{a|b\}$ and $\tau = \{x\}$ is...
Trees of ordered partitions

Given ordered partitions \( \sigma = \{A_1|\ldots|A_m\} \), and \( \tau = \{B_1|\ldots|B_n\} \) with \( m \leq n \) we write \( \sigma \leq \tau \) iff \( A_i = B_i \) for every \( i \in \{1, \ldots, m\} \).

One can label trees with ordered partitions...

One can build a tree from a set of ordered partitions. The tree of merged shuffles of \( \sigma = \{a|b\} \) and \( \tau = \{x\} \) is...
Let $F = \{T_1, \ldots, T_r\}$ and $G = \{U_1, \ldots, U_s\}$ be forests. $F \times_F G = \{T_i \times_F U_j\}, \ i \in \{1, \ldots, r\}, \ j \in \{1, \ldots, s\}$.
Let $F = \{T_1, \ldots, T_r\}$ and $G = \{U_1, \ldots, U_s\}$ be forests. 

$F \times_F G = \{T_i \times_F U_j\}, \ i \in \{1, \ldots, r\}, \ j \in \{1, \ldots, s\}$. 

The problem of describing $F \times_F G$ is reduced to that of describing its trees.
Product of forests

Let $F = \{T_1, \ldots, T_r\}$ and $G = \{U_1, \ldots, U_s\}$ be forests. $F \times_F G = \{T_i \times_F U_j\}, i \in \{1, \ldots, r\}, j \in \{1, \ldots, s\}$.

The problem of describing $F \times_F G$ is reduced to that of describing its trees.

How to compute the product of trees?
Computing the product of trees (an example).
Computing the product of trees (an example).
Product of trees

Computing the product of trees (an example).
Product of forests, a recursive construction

Product of forests, a recursive construction


Let $F$, $G$, and $H$ be three forests.

Let $F$, $G$, and $H$ be three forests.

- If $|F| = 1$, then $F \times_F G \cong G$. 
Let $F$, $G$, and $H$ be three forests.

- If $|F| = 1$, then $F \times_F G \cong G$.
- $(F + G) \times_F H \cong (F \times_F H) + (G \times_F H)$.

Let $F$, $G$, and $H$ be three forests.

- If $|F| = 1$, then $F \times_F G \cong G$.
- $(F + G) \times_F H \cong (F \times_F H) + (G \times_F H)$.
- $F \perp \times_F G \perp \cong ((F \times_F G) \perp) + (F \times_F G) + (F \perp \times_F G) \perp$. 
Computing the product of trees (an example).
Computing the product of trees (an example).
Computing the product of trees (an example).

\[ F \times F = (\quad + \quad F \times F \quad + \quad F \times F) \]
Computing the product of trees (an example).
Product of trees, a recursive formula

Computing the product of trees (an example).

\[ \times_F = ( + ( + , + )_{\perp} ) \]
Computing the product of trees (an example).
Computing the product of trees (an example).
Products of forests via Delannoy paths on Cartesian products.
A Delannoy path is a path on the first integer quadrant $\mathbb{N}^2 \subseteq \mathbb{Z}^2$ that starts from the origin and only uses northward, eastward, and north-eastward steps.
A Delannoy path is a path on the first integer quadrant $\mathbb{N}^2 \subseteq \mathbb{Z}^2$ that starts from the origin and only uses northward, eastward, and north-eastward steps.

A (finite) path on a poset $P$ is a non-empty sequence $\langle p_1, p_2, \ldots, p_h \rangle$ of elements of $P$ such that $p_i < p_j$ whenever $i < j$. (A path on $P$ is therefore the same thing as a chain of $P$.)
A **Delannoy path** is a path on the first integer quadrant $\mathbb{N}^2 \subseteq \mathbb{Z}^2$ that starts from the origin and only uses northward, eastward, and north-eastward steps.

A (finite) **path** on a poset $P$ is a non-empty sequence $\langle p_1, p_2, \ldots, p_h \rangle$ of elements of $P$ such that $p_i < p_j$ whenever $i < j$. (A path on $P$ is therefore the same thing as a chain of $P$.)

For each $i \in \{1, \ldots, n-1\}$, the pair $p_i, p_{i+1}$ is called a step of the path.
A Delannoy path is a path on the first integer quadrant $\mathbb{N}^2 \subseteq \mathbb{Z}^2$ that starts from the origin and only uses northward, eastward, and north-eastward steps.

A (finite) path on a poset $P$ is a non-empty sequence $\langle p_1, p_2, \ldots, p_h \rangle$ of elements of $P$ such that $p_i < p_j$ whenever $i < j$. (A path on $P$ is therefore the same thing as a chain of $P$.)

For each $i \in \{1, \ldots, n - 1\}$, the pair $p_i, p_{i+1}$ is called a step of the path.

Given a poset $P$, and two elements $p, q \in P$, we write $p \triangleleft q$ to indicate that $q$ covers $p$ in $P$, that is, $p < q$ and for every $s \in P$, if $p \leq s \leq q$, then either $s = p$ or $s = q$. 
Delannoy paths on Cartesian products of posets

Definition

Let $P_1, \ldots, P_n$ be posets, and let $P = P_1 \times \cdots \times P_n$ be their (Cartesian) product. Let $\langle p_1, \ldots, p_h \rangle$ be a path on $P$.

The step from $p_i = (p_{i,1}, \ldots, p_{i,n})$ to $p_{i+1} = (p_{i+1,1}, \ldots, p_{i+1,n})$ is a Delannoy step, written $p_i \prec p_{i+1}$, if and only if there exists $k \in \{1, \ldots, n\}$ such that $p_{i,k} \neq p_{i+1,k}$, and for each $j \in \{1, \ldots, n\}$, $p_{i,j} \leq p_{i+1,j}$.

The path $\langle p_1, \ldots, p_h \rangle$ on $P$ is a Delannoy path if and only if $p_1$ is a minimal element of $P$, and for each $i \in \{1, \ldots, n - 1\}$, $p_i \prec p_{i+1}$.
A Delannoy path on $P$ is thus a sequence of Delannoy steps starting from a minimal element of $P$. 
Delannoy paths on Cartesian products of posets

- A Delannoy path on $P$ is thus a sequence of Delannoy steps starting from a minimal element of $P$.

- Delannoy paths on a poset $P = P_1 \times \cdots \times P_n$ can be partially ordered by $\langle q_1, \ldots, q_m \rangle \leq \langle p_1, \ldots, p_h \rangle$ if and only if $m \leq h$ and $q_i = p_i$ for each $i \in \{1, \ldots, m\}$. 
Delannoy paths on Cartesian products of posets

- A Delannoy path on $P$ is thus a sequence of Delannoy steps starting from a minimal element of $P$.

- Delannoy paths on a poset $P = P_1 \times \cdots \times P_n$ can be partially ordered by $\langle q_1, \ldots, q_m \rangle \leq \langle p_1, \ldots, p_h \rangle$ if and only if $m \leq h$ and $q_i = p_i$ for each $i \in \{1, \ldots, m\}$.

- We denote by $D(P_1, \ldots, P_n)$ the poset of all Delannoy paths on $P$. 
A Delannoy path on $P$ is thus a sequence of Delannoy steps starting from a minimal element of $P$.

Delannoy paths on a poset $P = P_1 \times \cdots \times P_n$ can be partially ordered by $\langle q_1, \ldots, q_m \rangle \leq \langle p_1, \ldots, p_h \rangle$ if and only if $m \leq h$ and $q_i = p_i$ for each $i \in \{1, \ldots, m\}$.

We denote by $\mathcal{D}(P_1, \ldots, P_n)$ the poset of all Delannoy paths on $P$.

Clearly, $\mathcal{D}(P_1, \ldots, P_n)$ is a forest.
**Theorem**

Let $F$ and $G$ be forests. Then $D(F, G)$ is the product $F \times_F G$ in the category $F$:

$$F \xleftarrow{\pi_F} F \times_F G \xrightarrow{\pi_G} G.$$

Let $d \in F \times_F G$, with $d = \langle (f_1, g_1), \ldots, (f_n, g_n) \rangle$. The projection functions $\pi_F : F \times_F G \to F$ and $\pi_G : F \times_F G \to G$ are defined by

$$\pi_F(d) = f_n, \text{ and } \pi_G(d) = g_n.$$
Computing the product in $F$, an example
Computing the product in $F$, an example

\[ x \times_F y = \langle (x, a) \rangle \langle (x, a), (y, a) \rangle \]
Computing the product in $F$, an example

$$
\langle (x, a), (x, b) \rangle \langle (x, a), (y, a) \rangle \langle (x, a) \rangle
$$

Diagram:

- Nodes: $x, y, a, b, c, d$
- Edges: $x \rightarrow y$, $y \rightarrow b$, $b \rightarrow c$, $b \rightarrow d$

Product of forests

Enumerable
Computing the product in $F$, an example

\[ x \times_F y = \langle (x, a) \rangle \langle (x, a), (y, b) \rangle \langle (x, a), (y, a) \rangle \langle (x, a), (x, b) \rangle \]
Computing the product in $F$, an example

$$\langle (x, a) \rangle \langle (x, a), (y, b) \rangle \langle (x, a), (y, a) \rangle \langle (x, a), (x, b) \rangle \langle (x, a), (y, b), (y, c) \rangle$$
Computing the product in $F$, an example
Computing the product in $F$, an example
Computing the product in $F$, an example

$\langle x, a \rangle \times \langle y, a \rangle = \langle (x, a), (y, a) \rangle$

$\langle x, a \rangle \times \langle y, b \rangle = \langle (x, a), (y, b) \rangle$

$\langle x, a \rangle \times \langle y, c \rangle = \langle (x, a), (y, c) \rangle$

$\langle x, a \rangle \times \langle y, d \rangle = \langle (x, a), (y, d) \rangle$
Computing the product in $F$, an example
Computing the product in $F$, an example
Computing the product in $F$, an example

$\langle (x, a), (y, b), (y, c) \rangle$

$\langle (x, a), (y, b) \rangle$

$\langle (x, a) \rangle$
<table>
<thead>
<tr>
<th>Basic notions</th>
<th>In this talk</th>
<th>Product of forests</th>
<th>Enumeration</th>
</tr>
</thead>
</table>

Enumeration.
Delannoy numbers

The **Delannoy number** $D_{n,m}$ counts the number of Delannoy paths from $(0,0)$ to $(n,m)$. Delannoy numbers satisfy the following recurrence relation.

$$D_{n,m} = D_{n-1,m} + D_{n,m-1} + D_{n-1,m-1}$$
Delannoy numbers

The Delannoy number $D_{n,m}$ counts the number of Delannoy paths from $(0,0)$ to $(n,m)$. Delannoy numbers satisfy the following recurrence relation.

$$D_{n,m} = D_{n-1,m} + D_{n,m-1} + D_{n-1,m-1}$$

The following table shows some values of Delannoy numbers.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>13</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5</td>
<td>13</td>
<td>25</td>
<td>41</td>
<td>61</td>
<td>85</td>
<td>113</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>7</td>
<td>25</td>
<td>63</td>
<td>129</td>
<td>231</td>
<td>377</td>
<td>575</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>9</td>
<td>41</td>
<td>129</td>
<td>321</td>
<td>681</td>
<td>1289</td>
<td>2241</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>11</td>
<td>61</td>
<td>231</td>
<td>681</td>
<td>1683</td>
<td>3653</td>
<td>7183</td>
</tr>
</tbody>
</table>
A Formula for the number of elements of the products

Let $T, U$ be trees.

$$|T \times_F U| = \sum_{i \geq 0} \sum_{j \geq 0} t_i u_j D_{i,j},$$

where $t_i$ is the number of elements at level $i$ of $T$, and $u_j$ is the number of elements at level $j$ of $U$. 
A Formula for the number of elements of the products

Let $T, U$ be trees.

$$|T \times_F U| = \sum_{i \geq 0} \sum_{j \geq 0} t_i u_j D_{i,j},$$

where $t_i$ is the number of elements at level $i$ of $T$, and $u_j$ is the number of elements at level $j$ of $U$.

Example.

$$= 1 \cdot 1 \cdot D_{0,0} + 1 \cdot 1 \cdot D_{0,1} + 1 \cdot 2 \cdot D_{0,2} + 1 \cdot 1 \cdot D_{1,0} + 1 \cdot 1 \cdot D_{1,1} + 1 \cdot 2 \cdot D_{1,2} =$$

$$= 1 + 1 + 2 + 1 + 3 + 10 = 18.$$
A Formula for the number of elements of the products

\[ |T \times_F U| = \sum_{i \geq 0} \sum_{j \geq 0} t_i u_j D_{i,j} \]
A Formula for the number of elements of the products

$$| T \times_F U | = \sum_{i \geq 0} \sum_{j \geq 0} t_i u_j D_{i,j}$$
A Formula for the number of elements of the products

\[ |T \times_F U| = \sum_{i \geq 0} \sum_{j \geq 0} t_i u_j D_{i,j} \]
A Formula for the number of elements of the products

$$|T \times_F U| = \sum_{i \geq 0} \sum_{j \geq 0} t_i u_j D_{i,j}$$
References


Thank you for your attention.