Product rules and distributive laws

Joost Winter

University of Warsaw

April 2, 2016

Overview

- Main goal of this paper: a categorical framework for a *two-step* determinization process in which product rules, such as Brzozowski's rule $(xy)_a = x_a y + o(x) y_a$, or the familiar Leibniz rule from calculus $(xy)_a = x_a y + x y_a$, can be understood.
- First step (transforming a FST-coalgebra into a FS-coalgebra) is given by the product rule.
- Second step (transforming a FS-coalgebra into a F-coalgebra) is the usual determinization/linearization for weighted automata.
- We provide a general perspective on this process, including a coherence condition giving a sufficient condition for the two-step determinization process to be possible.

Distributive laws

Given:

1. A monad (T, η^T, μ^T).
2. Either a monad (S, η^S, μ^S), an endofunctor S or a copointed endofunctor (S, ϵ).

$$\begin{array}{lrcllrcl} \textbf{a.} & \lambda \circ \eta^T S & = & S\eta^T & \textbf{c.} & \lambda \circ \mu^T S & = & S\mu^T \circ \lambda T \circ T\lambda \\ \textbf{b.} & \lambda \circ T\eta^S & = & \eta^S T & \textbf{d.} & \lambda \circ T\mu^S & = & \mu^S T \circ S\lambda \circ \lambda S \\ \textbf{e.} & \epsilon T \circ \lambda & = & T\epsilon & & & & \end{array}$$

- A distributive law between monads satisfies **a.**, **b.**, **c.**, and **d.**
- A distributive law of a monad over an endofunctor satisfies **a.** and **c.**
- A distributive law of a monad over a copointed endofunctor satisfies **a.**, **c.**, and **e.**

Product rules – three examples

(frequently featured in work by Rutten and many others as coinductive definitions)

- Brzozowski rule – convolution product:

$$\begin{array}{rclrcl} o(1) & = & 1 & 1_a & = & 0 \\ o(xy) & = & o(x)o(y) & (xy)_a & = & x_a y + o(x) y_a \end{array}$$

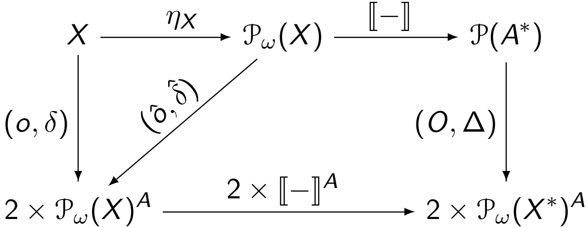
- Leibniz rule – shuffle product:

$$\begin{array}{rclrcl} o(1) & = & 1 & 1_a & = & 0 \\ o(x \otimes y) & = & o(x)o(y) & (x \otimes y)_a & = & x_a \otimes y + x \otimes y_a \end{array}$$

- Pointwise rule – Hadamard product:

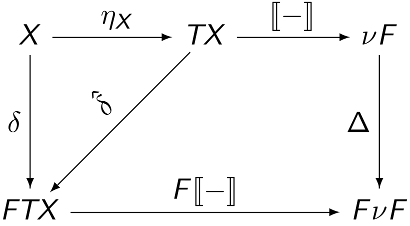
$$\begin{array}{rclrcl} o(\mathbf{1}) & = & 1 & \mathbf{1}_a & = & \mathbf{1} \\ o(x \odot y) & = & o(x)o(y) & (x \odot y)_a & = & x_a \odot y_a \end{array}$$

Deteriminizing a nondeterministic automaton, bialgebraically



Categorically well-understood (see e.g. the work by Bartels, Jacobs/Silva/Sokolova and many others) via bialgebras and distributive laws.
$(\hat{o}, \hat{\delta})$ can be obtained from (o, δ) using the distributive law.

The general picture

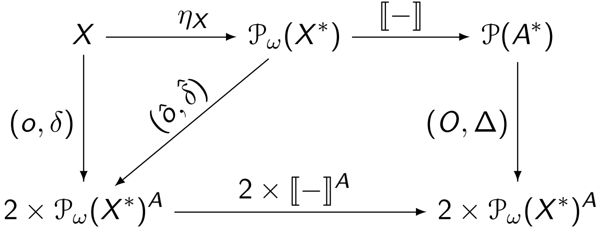


Given a distributive law $\lambda : TF \Rightarrow FT$ (monad over endofunctor) the extension $\hat{\delta}$ is obtained by:

$$\hat{\delta} = F\mu_X \circ \lambda TX \circ T\delta$$

The distributive law can be seen as *defining* a T-algebra structure on the final F-coalgebra.

Determinization for context-free languages (one step)

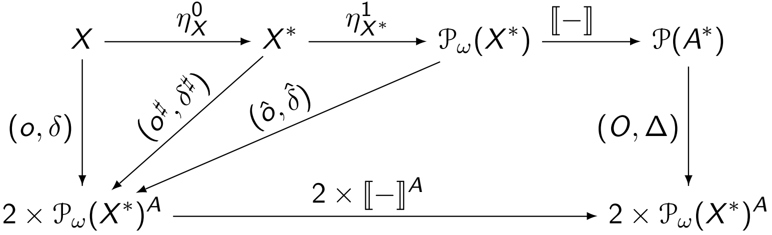


Various variants, most involving a distributive law of a monad over a (cofree) *copointed* functor. See e.g. Winter/Bonsangue/Rutten, and Bonsangue/Hansen/Kurz/Rot.

Can also be generalized from (context-free) languages to (constructively) algebraic power series.

When we regard the distributive law as defining an algebra structure on the final coalgebra, we can see it also as *defining* the convolution product on power series (together with linearity of derivative). Similarly, there are laws for the shuffle and Hadamard products/product rules.

Determinization for context-free languages (two steps)



This can again be generalized from languages to power series.
But, can we understand this diagram categorically, too?

Semimodules and algebras for a semiring

Given a semiring $(S, \bar{0}, \bar{1}, \oplus, \cdot)$, a (left) S-semimodule is a tuple $(X, 0, +, \times)$:

1. $(X, 0, +)$ is a commutative monoid,
2. $\times : S \times X \to X$ (left-scalar product) satisfies:

$$
\begin{array}{rcl rcl}
(s \oplus t) \times x & = & s \times x + t \times x & \bar{0} \times x & = & 0 \\
s \times (x + y) & = & s \times x + s \times y & s \times 0 & = & 0 \\
s \times (t \times x) & = & (st) \times x & \bar{1} \times x & = & x
\end{array}
$$

Given a *commutative* semiring S, a (unital, associative) S-algebra (see e.g. Eilenberg 1974) is a tuple $(X, 0, 1, +, \cdot, \times)$:

1. $(X, 0, 1, +, \cdot)$ is a semiring.
2. $(X, 0, +, \times)$ is a S-semimodule.
3. Satisfying: $s \times (xy) = (s \times x)y = x(s \times y)$.

…via distributive laws (between monads)

Recall:

1. Algebras for the monad $-^*$ are *monoids*.
2. Algebras for the monad Lin_S defined by

$$\mathsf{Lin}_S(X) = \{f \in S^X \mid \mathsf{supp}(f) \text{ is finite}\}$$

   with the 'expected' (see e.g. Jacobs/Silva/Sokolova 2012) multiplication (on the left/right) are left/right S-semimodules.

If S is commutative, there is a distributive law of $-^*$ over Lin_S, creating a monad structure on $S\langle - \rangle := \mathsf{Lin}_S(-^*)$. Its algebras are the S-algebras as just defined.

. . . via distributive laws (between monads) (2)

The distributive law $\lambda : (\mathsf{Lin}_S(-))^* \Rightarrow \mathsf{Lin}_K(-^*)$ can be given by:

$$\lambda_X\left(\prod_{i=1}^{n}\sum_{j=1}^{m_i} k_{ij} \times x_{ij}\right) = \sum_{j_1=1}^{m_1}\cdots\sum_{j_n=1}^{m_n}\left(\prod_{i=1}^{n} k_{ij_i} \times \prod_{i=1}^{n} x_{ij_i}\right)$$

(see e.g. Beck '69 for the case $S = \mathbb{Z}$)

S	S-semimodules	S-algebras
$\mathbb{B}$	(join) semilattices	idempotent semirings
$\mathbb{N}$	commutative monoids	semirings
$\mathbb{Z}$	Abelian groups	rings

Combining distributive laws (acc. to Cheng)

Let S, T, and U be monads, and let $\lambda^0 : UT \Rightarrow TU$, $\lambda^1 : US \Rightarrow SU$, and $\lambda^2 : TS \Rightarrow ST$ be distributive laws.

Theorem

T.f.a.e.:

1. *The diagram of natural transformations*

$$\begin{array}{ccccc}
UTS & \xrightarrow{U\lambda^2} & UST & \xrightarrow{\lambda^1 T} & SUT \\
{\scriptstyle \lambda^0 S}\downarrow & & & & \downarrow{\scriptstyle S\lambda^0} \\
TUS & \xrightarrow{T\lambda^1} & TSU & \xrightarrow{\lambda^2 U} & STU
\end{array}$$

   (the Yang Baxter diagram) commutes.

2. $\lambda^2 U \circ T\lambda^1$ *is a dist. law of the composite monad* TU *over* S.
3. $S\lambda^0 \circ \lambda^1 T$ *is a dist. law of* U *over the composite monad* ST.

Same result for dist. laws over endofunctors

Now, let S and T be monads, and F and endofunctor... λ^0 dist. law between monads, λ^1 and λ^2 dist. law of monad over endofunctor

Theorem

T.f.a.e.:

1. *The diagram of natural transformations*

$$
\begin{array}{ccccc}
TSF & \xrightarrow{T\lambda^2} & TFS & \xrightarrow{\lambda^1 S} & FTS \\
\Big\downarrow{\scriptstyle \lambda^0 F} & & & & \Big\downarrow{\scriptstyle F\lambda^0} \\
STF & \xrightarrow{S\lambda^1} & SFT & \xrightarrow{\lambda^2 T} & FST
\end{array}
$$

   (the Yang Baxter diagram) commutes.

2. $\lambda^2 T \circ S\lambda^1$ *is a dist. law of the composite monad* ST *over* F*.*

(also works for copointed)

Instance: the Hadamard product

We can use the framework of the previous page with the following instances (assume S to be commutative):

- $S = \mathrm{Lin}_S(-)$, the monad for S-semimodules
- $T = -^*$, the list monad
- $F = S \times -^A$, the endofunctor for Moore machines with output in S.

Take $\lambda^0 : TS \Rightarrow ST$ as defined before (making ST the monad for S-algebras), make λ^2 the pointwise distributive law for S-weighted automata, and define $\lambda^1 : (S \times -^A)^* \Rightarrow (S \times (-^*)^A)$ again pointwise, as follows:

$$\begin{array}{rcl} \lambda^1(\epsilon) & = & (1_S, (a \mapsto \varepsilon)) \\ \lambda^1((o, a \mapsto d_a)w) & = & \texttt{let } (p, a \mapsto e_a) \texttt{ = } \lambda^1(w) \texttt{ in} \\ & & \quad (op, a \mapsto d_a e_a) \end{array}$$

How about convolution and shuffle product

- λ^1 in the previous slide can be seen as *defining*, coinductively, the product rule $1_a = 1$, $(xy)_a = x_a y_a$ on the final coalgebra.
- How about other rules, such as $(xy)_a = x_a y + o(x) y_a$ or $(xy)_a = x a_y + x y_a$?

How about convolution and shuffle product

- λ^1 in the previous slide can be seen as *defining*, coinductively, the product rule $1_a = 1$, $(xy)_a = x_a y_a$ on the final coalgebra.
- How about other rules, such as $(xy)_a = x_a y + o(x) y_a$ or $(xy)_a = x a_y + x y_a$?
- Because of the presence of addition on the right hand side of the equations, it seems this will not work, and we need a law of the type
$$\lambda : TF \Rightarrow FST$$

Distributive laws into a composite monad

Given:

1. A monad (T, η^T, μ^T), a monad (S, η^S, μ^S), a distributive law between monads $\lambda^0 : TS \Rightarrow ST$, and an endofunctor F,

A natural transformation $\lambda : TF \Rightarrow FST$ is a distributive law of T over F into the composite monad ST whenever:

$$\begin{aligned} \lambda \circ \eta^T F &= F\eta^{ST} \\ \lambda \circ \mu^T F &= F\mu^{ST} \circ \lambda ST \circ T\lambda \end{aligned}$$

If (F, ϵ) is a copointed endofunctor, additionally:

$$\epsilon ST \circ \lambda = \eta_S \circ T\epsilon$$

The other two product rules

Brzozowski/convolution rule:

$$\begin{array}{rcl} \lambda_X(1) & = & (1, 1, a \mapsto 0) \\ \lambda_X(x, o, a \mapsto d_a)w & = & \texttt{let}\ (y, p, a \mapsto e_a) = \lambda_X(w)\ \texttt{in} \\ & & \quad (xy, op, a \mapsto d_a y + o e_a) \end{array}$$

Leibniz/shuffle rule:

$$\begin{array}{rcl} \lambda_X(1) & = & (1, 1, a \mapsto 0) \\ \lambda_X(x, o, a \mapsto d_a)w & = & \texttt{let}\ (y, p, a \mapsto e_a) = \lambda_X(w)\ \texttt{in} \\ & & \quad (xy, op, a \mapsto d_a y + x e_a) \end{array}$$

Another coherence condition (1)

$$\begin{array}{ccccccc}
TSF & \xrightarrow{T\lambda^2} & TFS & \xrightarrow{\lambda^1 S} & FSTS & \xrightarrow{FS\lambda^0} & FSST \\
\downarrow{\scriptstyle \lambda^0 F} & & & & & & \downarrow{\scriptstyle F\mu^S T} \\
STF & \xrightarrow{S\lambda^1} & SFST & \xrightarrow{\lambda^2 ST} & FSST & \xrightarrow{F\mu^S T} & FST
\end{array}$$

Theorem

Given monads (T, η^T, μ^T) *and* (S, η^S, μ^S)*, and an endofunctor* F *such that:*

- λ^0 *is a distributive law of the monad* T *over the monad* S*.*
- λ^1 *is a distributive law of the monad* T *over the endofunctor* F *into the composite monad* ST*...*
- λ^2 *is a distributive law of the monad* S *over the endofunctor* F*.*

Another coherence condition (1)

$$
\begin{array}{ccccccc}
TSF & \xrightarrow{T\lambda^2} & TFS & \xrightarrow{\lambda^1 S} & FSTS & \xrightarrow{FS\lambda^0} & FSST \\
\downarrow{\scriptstyle \lambda^0 F} & & & & & & \downarrow{\scriptstyle F\mu^S T} \\
STF & \xrightarrow{S\lambda^1} & SFST & \xrightarrow{\lambda^2 ST} & FSST & \xrightarrow{F\mu^S T} & FST
\end{array}
$$

Theorem

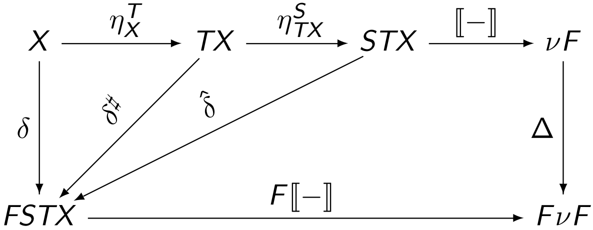
. . . $\hat{\lambda} : STF \to FST$, given by

$$STF \xrightarrow{S\lambda^1} SFST \xrightarrow{\lambda^2 ST} FSST \xrightarrow{F\mu^S T} FST$$

is a distributive law of the composite monad ST over F iff the coherence condition holds.

(again works for copointed)

Two-step determinization: general



$\delta^\sharp$ and $\hat{\delta}$ can now be expressed in terms of the various distributive laws and δ:

$$\delta^\sharp = F\mu_X^{ST} \circ \lambda^1_{STX} \circ T\delta$$

$$\hat{\delta} = F\mu_{TX}^{S} \circ \lambda^2_{STX} \circ S\delta\sharp$$

$\delta^\sharp$ can also be obtained, equivalently, in terms of the distributive law $\hat{\lambda}$ of ST over F.

Future work (maybe almost trivial?)

One possibly fascinating question is the following: can you use a similar generalization to the one given in this paper for distributive laws between monads into composite monads, so as to give a more general version of Cheng's result in the original setting?