Pointwise extensions of GSOS-defined operations

Citation for published version (APA):

Document status and date:
Published: 01/01/2010

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 20. Sep. 2020
Pointwise Extensions of GSOS-Defined Operations

Helle Hvid Hansen
Eindhoven University of Technology

Bartek Klin
University of Cambridge, University of Warsaw

Distributive laws of syntax over behaviour (cf. [1, 3]) are, among other things, a well-structured way of defining algebraic operations on final coalgebras. For a simple example, consider the set $B\omega$ of infinite streams of elements of $B$; this carries a final coalgebra $w = (hd, tl) : B\omega \to B \times B\omega$ for the endofunctor $F = B \times -$ on Set. If $B$ comes with a binary operation $+$, one can define an addition operation $\oplus$ on streams coinductively:

$$hd(\sigma \oplus \tau) = hd(\sigma) + hd(\tau) \quad tl(\sigma \oplus \tau) = tl(\sigma) \oplus tl(\tau).$$

It is easy to see that these equations define a distributive law, i.e., a natural transformation $\lambda : \Sigma F \Rightarrow F \Sigma$, where $\Sigma X = X^2$ is the signature endofunctor corresponding to a single binary operation. The operation $\oplus : B\omega \times B\omega \to B\omega$ is now defined as the unique morphism to the final coalgebra as in:

$$\begin{array}{c}
\Sigma B\omega \xrightarrow{\Sigma w} \Sigma F B\omega \xrightarrow{\lambda_{\Sigma B\omega}} F \Sigma B\omega \\
\oplus \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
B\omega \xrightarrow{\text{it}} \xrightarrow{\text{w}} F B\omega \\
\end{array} \tag{1}$$

For another example, consider an operation $\boxplus$ where the first element of one stream is added to every element of the other one. This can be defined by equations:

$$hd(\sigma \boxplus \tau) = hd(\sigma) + hd(\tau) \quad tl(\sigma \boxplus \tau) = \sigma \boxplus tl(\tau).$$

These do not define a simple distributive law as above; however, they do define a slightly more complex law of the type $\lambda : \Sigma (\text{Id} \times \mathcal{F}) \Rightarrow F \Sigma$, and the operation $\boxplus$ is defined from it similarly as in (1).

Mealy machines are coalgebras for the functor $(B \times -$)$^\Delta$, where $A$ is the input, and $B$ the output alphabet. A final coalgebra structure is carried by the set $\Gamma$ of functions from $A^\omega$ to $B^\omega$ that are causal: An $n$-ary operation on streams is causal if the $k$th element of the result only depends on the first $k$ elements of each of the arguments. Operations on $B^\omega$ can be pointwise extended to operations on functions from $A^\omega$ to $B^\omega$ in the obvious way, e.g., one can define $(x \boxplus y)(\sigma) = x(\sigma) \boxplus y(\sigma)$ for $x, y : A^\omega \to B^\omega$. If an operation $*: (B^\omega)^n \to B^\omega$ is causal then the pointwise extension $\bar{*}$ preserves causality and hence is an operation on $\Gamma$.

Stream operations that are defined via a distributive law of syntax endofunctors $\Sigma$ over $B \times -$ have pointwise extensions that are defined by a distributive law of $\Sigma$ over $(B \times -$)$^\Delta$. It follows that the resulting extended operations on functions preserve causality. Relevant distributive laws can be syntactically presented with a simple inference rule formalism. For example, the operation $\boxplus$ on Mealy machines is defined by rules:

$$\begin{array}{c}
\frac{x \overset{a}{\xrightarrow{\text{it} p}} x'}{x \boxplus y \overset{a}{\xrightarrow{\text{it} (p \boxplus q)}} x' \boxplus y'}
\frac{y \overset{a}{\xrightarrow{\text{it} q}} y'}{x \boxplus y \overset{a}{\xrightarrow{\text{it} (p \boxplus q)}} x' \boxplus y'}
\end{array}$$

with $a$ ranging over $A$ and $p, q$ ranging over $B$. The meaning of a clause $x \overset{a}{\xrightarrow{\text{it} b}} x'$ is that a Mealy machine $x$, upon receiving input $a$, outputs $b$ and transforms into a machine $x'$. 
Pointwise extensions of some other operations are more problematic. For example, the naive idea to define the pointwise extension of $\boxplus$ by:

$$
\begin{align*}
  & x \frac{a \triangleright p}{a \triangleright p \downarrow x' \Box y}\quad y \frac{a \triangleright q}{x \triangleright y'} \\
\end{align*}
$$

does not work, i.e., the resulting operation does not behave as the stream operation $\boxplus$ pointwise. Nevertheless, the pointwise extension of $\boxplus$ can be defined in terms of a distributive law, if one extends the syntax with a family of auxiliary operators. Specifically, for every $a \in A$ we add a unary operator $a \triangleright -$ and take rules:

$$
\begin{align*}
  & x \frac{a \triangleright p}{a \triangleright b \downarrow p \triangleright x' \Box y'} \quad y \frac{a \triangleright q}{x \triangleright y'} \\
\end{align*}
$$

where $a, b$ range over $A$ and $p, q$ over $B$. Intuitively, the new operators act like one-slot input buffers: a Mealy machine $a \triangleright x$, upon receiving input $b$, passes $a$ as input to $x$, returns its output, and stores $b$ in the buffer.

The above rules define a distributive law of the type $\lambda : F(\Sigma A) \Rightarrow FA$, where $\Sigma$ corresponds to the syntax extended with auxiliary operators $a \triangleright -$ and $\Sigma'$ is the free monad over $\Sigma$. Such laws are called (abstract) GSOS laws (cf. [1, 3]), and they induce operations on final $F$-coalgebras similarly as in (1). It turns out that the induced operation is the pointwise extension of the operation $\boxplus$ on streams.

So far we have described just two very simple examples of operations and their pointwise extensions. However, the techniques presented here are far more general: they apply to arbitrary operations definable with certain distributive laws, and even to arbitrary behaviour endofunctors $F$ on $\text{Set}$. Specifically:

- Simple distributive laws $\lambda : \Sigma FA \Rightarrow F\Sigma A$ can be extended pointwise to laws $\overline{\lambda} : \Sigma (F \rightarrow)^A \Rightarrow (F\Sigma A)^A$,

- GSOS distributive laws $\lambda : \Sigma (\text{Id} \times F) \Rightarrow F\Sigma'$ can be extended to laws $\overline{\lambda} : \Sigma (\text{Id} \times (F \rightarrow)^A) \Rightarrow (F\Sigma')^A$, where $\Sigma$ arises from $\Sigma$ by adding a family of unary “buffer” operations $a \triangleright -$ for $a \in A$, so that the resulting operations on final $(F \rightarrow)^A$-coalgebras are pointwise extensions of their counterparts on $F$-coalgebras.

It is interesting that the technique of adding “buffer” operations is enough to solve the problem of pointwise extension for arbitrary GSOS operations and arbitrary behaviour functors. We would like to understand better the relationship of the buffer operations to the structure of the final $(F \rightarrow)^A$-coalgebra. For example, in the case of streams and Mealy machines, for every causal function $f : A^\omega \Rightarrow B^\omega$, $a \in A$ and $\sigma \in A^\omega$, we have that $(a \triangleright f)(\sigma) = f((a : \sigma))$, and the final $(B \rightarrow)^A$-structure on $\Gamma$ is the map $\gamma : A^\omega \Rightarrow (B \times A)^\omega$ defined by $\gamma(f)(\sigma) = (\text{hd}, \text{tl}) \circ (a \triangleright f)$. Another observation is that the buffer operations are already expressible in Rutten’s stream calculus (cf. [2]): if an expression $e(\sigma)$ is a stream calculus specification of a stream function $f$, then the expression obtained by substituting $[a] + X \times \sigma$ for $\sigma$ in $e$ specifies $a \triangleright f$, where $[a]$ denotes the stream $(a, 0, 0, 0, \ldots)$.

References

