On Injections

Grigore Roșu

July 19, 2018

1 Introduction

2 Parametric Productions

Recall parametric productions (written using frontend syntax):

$$\texttt{syntax}\{P_1,\ldots,P_k\}\ N ::= T_1 N_1 \ldots T_n N_n T_{n+1}$$

The non-terminals $N, N_1, \ldots, N_n$ can make use of parameters $P_1, \ldots, P_k$. We also discussed about allowing the parameters to be optionally specified among the terminals $T_1, \ldots, T_n, T_{n+1}$ and nonterminals $N_1, \ldots, N_n$, e.g. $T_1\{P_1\}N_1 \ldots$, but that is irrelevant for this discussion. The implicit semantics of parametric productions is that of infinitely many instances, one for each concrete parameter instance. In KORE, to each production like above we associate a parametric symbol:

$$\sigma\{P_1,\ldots,P_k\}(N_1,\ldots,N_n) : N$$

3 Injections: The Problem

Now let's consider the special case of injections, which correspond to productions like above where $n = 1$ and $T_1 = T_2 = \epsilon$, that is,

$$\texttt{syntax}\ \{P_1,\ldots,P_k\}N ::= N_1$$

As a concrete example, assume the following parametric definitions:

$$\begin{array}{l}\texttt{sort Map}\{K,V\}\\ \texttt{syntax}\{V\}\ \texttt{MapInt}\{V\} ::= \texttt{Map}\{\texttt{Int},V\}\end{array}$$

Dorel: A minor observation. Even if we use curly brackets for the list of parameters, their order in the list is important for instantiation, Otherwise, we should write something like $\texttt{Map}\{K \Rightarrow \texttt{Int}, V \Rightarrow V\}$. If the parameters order matters, then their names in the "sort" definition/production does not matter.

Dorel: The second syntax production can be simply written as

$$\texttt{syntax}\{V\}\ \texttt{MapInt} ::= \texttt{Map}\{\texttt{Int}, V\}$$

There is no reason to specify the list of parameters twice; always the lhs will depend on all the parameters of the production. I.e.,

$$\texttt{syntax}\{P_1, \ldots, P_k\}\ N ::= \ldots$$

is equivalent to

$$\texttt{syntax}\{P_1, \ldots, P_k\}\ N\{P_1, \ldots, P_k\} ::= \ldots$$

By abuse of notation, let

$$\texttt{inj}_{N_1,N}\{P_1, \ldots, P_k\}(N_1) : N$$

be the parametric symbol corresponding to the generic injection production above. Note that, for now, $\texttt{inj}$ is *not* parametric in the two sorts, each $\texttt{inj}_{N_1,N}$ is a separately-defined ordinary symbol. For our example,

$$\texttt{inj}_{\texttt{Map}\{\texttt{Int},V\},\texttt{MapInt}\{V\}}\{V\}(\texttt{Map}\{\texttt{Int}, V\}) : \texttt{MapInt}\{V\}$$

Brandon: if each way of subscripting $\texttt{inj}$ is a distinct non-parametric symbol, how do we have parameter V

Injection symbols are different from ordinary symbols, in that they inherit an expected semantics of "injections". For example, take an instance θ of the parameters $P_1, \ldots, P_k$. Then any element of sort $\theta(N_1)$ is expected to also be found among the elements of sort $\theta(N)$, possibly renamed.

Dorel: It is not clear for me how the renaming can be involved here. I think that $P_1, \ldots, P_k$ are the only parameters allowed in the production definition.

Moreover, such injections are expected to be consistent.

Example 3.1. For example, assume another production

$$\texttt{syntax}\{Q_1, \ldots, Q_\ell\}\ M ::= M_1$$

with corresponding injection

$$\texttt{inj}_{M_1,M}\{Q_1, \ldots, Q_\ell\}(M_1) : M$$

such that there is some instance ρ of the parameters $Q_1, \ldots, Q_\ell$ with $\rho(M_1) = \theta(N_1)$ and $\rho(M) = \theta(N)$.

Dorel: We should formally define what exactly means an equality $\rho(M) = \theta(N)$. Here is an example showing how I understand it:
We suppose that the above example is written as follows:

$$\begin{aligned}&\texttt{sort}\ \texttt{Map}\{K, V\}\\ &\texttt{syntax}\{V'\}\ \texttt{MapInt}\{V'\} ::= \texttt{Map}\{\texttt{Int}, V'\}\end{aligned}$$

(this could be a particular example when the parameter V is renamed as V'). If $\theta(K) = \texttt{Int}$, $\theta(V) = \texttt{String}$, and $\rho(V') = \texttt{String}$ then $\theta(\texttt{Map}\{K, V\}) = \rho(\texttt{Map}\{\texttt{Int}, V'\})$.
Supposing that the grammar include only these productions, can we deduce that $\theta(\texttt{Map}\{K, V\}) = \rho(\texttt{MapInt}\{V'\})$? This could makes sense since $\texttt{MapInt}\{V'\}$ is the smallest set including $\texttt{Map}\{\texttt{Int}, V'\}$.

Brandon: I think this simply means applying the substitution gives identical instantiations of the same parameterized sort.
Then we certainly want the following to hold:

$$\texttt{inj}_{N_1,N}\{\theta(\bar{P})\}(\varphi : \theta(N_1)) = \texttt{inj}_{M_1,M}\{\rho(\bar{Q})\}(\varphi : \rho(M_1))$$

Example 3.2. As another example, assume productions

$$\begin{aligned}\texttt{syntax}\{A_1, \ldots, A_a\}\ X &::= X_1\\ \texttt{syntax}\{B_1, \ldots, B_b\}\ Y_1 &::= Y_2\\ \texttt{syntax}\{C_1, \ldots, C_c\}\ Z_1 &::= Z_2\end{aligned}$$

such that there are some parameter instances α, β, and γ with $\alpha(X_1) = \beta(Y_1)$, $\alpha(X) = \gamma(Z)$, and $\beta(Y_2) = \gamma(Z_2)$. Then we certainly want

$$\texttt{inj}_{X_1,X}\{\alpha(\bar{A})\}(\texttt{inj}_{Y_2,Y_1}\{\beta(\bar{B})\}(\varphi : \beta(Y_2))) = \texttt{inj}_{Z_2,Z}\{\gamma(\bar{C})\}(\varphi : \gamma(Z_2))$$

Dorel: The rule we can learn from this example is: "the transitivity of injections does make sense (only?) on the (legal) sort instances".

Example 3.3. We also want parametric sorts to lift subsorts. That is, if

$$\texttt{inj}_{N_1,N}\{P_1, \ldots, P_k\}(N_1) : N$$

is an injection parametric symbol corresponding to production

$$\texttt{syntax}\{P_1, \ldots, P_k\}\ N ::= N_1,$$

then we also want to have parametric symbols

$$\texttt{inj}_{sort\{N_1,\ldots\},sort\{N,\ldots\}}\{P_1, \ldots, P_k, \ldots\}(sort\{N_1, \ldots\}) : sort\{N, \ldots\}$$

for any other parametric sort $sort\{\_, \ldots\}$.
Dorel: Here is an instance of the above rule. Suppose that $sort\{\_, \ldots\}$ is

$$\texttt{syntax}\{S\}\ \texttt{List}\{S\}$$

then we have

$$\texttt{inj}_{\texttt{List}\{\texttt{Map}\{\texttt{Int},V\}\},\texttt{List}\{\texttt{MapInt}\{V\}\}}(\texttt{List}\{\texttt{MapInt}\{V\}\}) : \texttt{List}\{\texttt{Map}\{\texttt{Int}, V\}\}$$

The rule we can learn from this example is: "the injections propagate among the parameterized sorts". Shouldn't we consider here only the cases of instances, similar to the transitivity?

Example 3.4. The situation is actually a lot more complex! The injection symbols need to be consistent not only among themselves, but also w.r.t. other symbols that end up being overloaded due to parametricity. Consider, for example:

$$\begin{aligned}\texttt{syntax Exp} &::= \texttt{Int}\\ \texttt{syntax}\{S\}\ \texttt{List}\{S\} &::= \texttt{cons}(S, \texttt{List}\{S\})\end{aligned}$$

or in KORE:

$$\begin{aligned}&\texttt{sort List}\{S\}\\ &\texttt{symbol inj}_{\texttt{Int},\texttt{Exp}}(\texttt{Int}) : \texttt{Exp}\\ &\texttt{symbol cons}\{S\}(S, \texttt{List}\{S\}) : \texttt{List}\{S\}\end{aligned}$$

Then we need to add the following axiom

$$\begin{aligned}&\texttt{inj}_{\texttt{List}\{\texttt{Int}\},\texttt{List}\{\texttt{Exp}\}}(\texttt{cons}\{\texttt{Int}\}(\varphi : \texttt{Int}, \psi : \texttt{List}\{\texttt{Int}\}))\\ &\qquad = \texttt{cons}\{\texttt{Exp}\}(\texttt{inj}_{\texttt{Int},\texttt{Exp}}(\varphi), \texttt{inj}_{\texttt{List}\{\texttt{Int}\},\texttt{List}\{\texttt{Exp}\}}(\psi))\end{aligned}$$

Dorel: The rule we can learn from this example is: "if the injections and a parametric symbol are *well-defined (legal)* on both the arguments and the result, then they may interchange (or, in other words, the injection propagates through parametric symbols).

Example 3.5. But what if the result of the parametric symbol does not depend on the parameter, that is, say

$$\texttt{symbol length}\{S\}(\texttt{List}\{S\}) : \texttt{Int}$$

Then we need to add an axiom as follows

$$\texttt{length}\{\texttt{Int}\}(\varphi : \texttt{List}\{\texttt{Int}\}) = \texttt{length}\{\texttt{Exp}\}(\texttt{inj}_{\texttt{List}\{\texttt{Int}\},\texttt{List}\{\texttt{Exp}\}}(\varphi))$$

Dorel: This example is a particular case of the previous one if we consider

$$\texttt{inj}_{S,S}(\varphi) = \varphi$$

Example 3.6. On the other hand, if the result of a symbol depends on a parameter that is not constrained by its arguments, then we cannot add any injection axioms. For example, consider

$$\texttt{symbol cast}\{P_1, P_2\}(P_l 1) : P_2$$

While it makes sense to add axioms like

$$\texttt{inj}_{\texttt{Int},\texttt{Exp}}(\texttt{cast}\{\texttt{Int}, \texttt{Int}\}(\varphi : \texttt{Int}) = \texttt{cast}\{\texttt{Int}, \texttt{Exp}\}(\varphi)$$

you cannot instantiate P_2 with everything, e.g., you cannot add

$$\texttt{inj}_{\texttt{Int},\texttt{String}}(\texttt{cast}\{\texttt{Int},\texttt{Int}\}(\varphi : \texttt{Int}) = \texttt{cast}\{\texttt{Int},\texttt{String}\}(\varphi)$$

because `Int` is not a subset of `String`.
Dorel: The rule we can learn from this example is: "we have to consider only those injections defined on sort instances that are in a *subsort partial-order relation*".

4 Injections: The Proposal

So things are really tricky. What I propose is to add the following axioms for now, and *in parallel* to have somebody really interested in this problem study it in depth. Note that knowledge of order-sorted algebra will be a big plus here!

(1) Define/assume one parametric symbol

$$\texttt{symbol inj}\{P_1, P_2\}(P_1) : P_2$$

This is what we do now, too. Note that this is *different* from having one `inj` symbol for each subsorting. In particular, for the subsorting discussed in the previous section,

$$\texttt{syntax}\{V\}\ \texttt{MapInt}\{V\} ::= \texttt{Map}\{\texttt{Int}, V\}$$

instead of the injection label we had there, namely

$$\texttt{symbol inj}_{\texttt{Map}\{\texttt{Int},V\},\texttt{MapInt}\{V\}}\{V\}(\texttt{Map}\{\texttt{Int}, V\}) : \texttt{MapInt}\{V\}$$

we refer to this subsorting using the generic injection

$$\texttt{inj}\{\texttt{Map}\{\texttt{Int}, V\}, \texttt{MapInt}\{V\}\}.$$

I do not know how to state that a subsorting has been declared, or if it is needed, so I postpone this aspect for now.

(2) Axiomatize that `inj` is functional, because otherwise we will not be able to prove that $1 + x$, etc., are "terms":

Dorel: $1 + \texttt{inj}\{\texttt{Id}, \texttt{Int}\}(x)$?

$$\texttt{axiom}\{P_1, P_2\}\ \forall x : P_1 . \exists y : P_2 . \texttt{inj}\{P_1, P_2\}(x) = y$$

Dorel: What if we want that an identifier not to be a `Bool` and an `Int` in the same time?

$$(\forall X)\neg(\texttt{inj}\{\texttt{Id}, \texttt{Bool}\}(X) \wedge \texttt{inj}\{\texttt{Id}, \texttt{Int}\}(X))$$

I do not think that we want to axiomatize that `inj` is an *injective* function, because we may want to inject sets of larger cardinalities (e.g.identifiers) into sets of smaller cardinalities (e.g., `Bool`: (x or y) and $x = x$, etc.).

Dorel: I do not understand the above example. If x and y are two different identifiers, then their injections denote different `Bool` names.

(3) Axiomatize that `inj` is reflexive:

$$\texttt{axiom}\{S\}\ \texttt{inj}\{S,S\}(\varphi : S) = \varphi$$

(4) Axiomatize that injections compose (or are transitive):

$$\texttt{axiom}\{P_1,P_2,P_3\}\ \texttt{inj}\{P_2,P_3\}(\texttt{inj}\{P_1,P_2\}(\varphi : P_1)) = \texttt{inj}\{P_1,P_3\}(\varphi)$$

(5) And here is an aggressive axiom, but which I dare to claim is OK: Unrestricted propagation through parametric symbols. For each symbol $\sigma\{P_1,\ldots,P_k\}(S_1,\ldots,S_n) : S$ where $P_1,\ldots,P_k$ are parameters and $S_1,\ldots,S_n,S$ are sorts potentially parametric in $P_1,\ldots,P_k$, add axiom

$$\begin{aligned}&\texttt{axiom}\{P_1,\ldots,P_k,P'_1,\ldots,P'_k\}\\ &\qquad\qquad \texttt{inj}\{S,S'\}(\sigma\{P_1,\ldots,P_k\}(\varphi_1 : S_1,\ldots,\varphi_n : S_n)\\ &\qquad\qquad\ = \sigma\{P'_1,\ldots,P'_k\}(\texttt{inj}\{S_1,S'_1\}(\varphi_1),\ldots,\texttt{inj}\{S_n,S'_n\}(\varphi_n))\end{aligned}$$

OK, now I can hear you guys yelling at me, "what the heck is this? It is too aggressive!". In particular, that it admits nonsensical properties to be proven, like the one for the cast symbol in Example 3.6 above. And I would agree, but I do believe that we should either disallow symbols like in Example 3.6 above, or otherwise give a serious warning. Note that `inj` itself is such a symbol ;-).

What makes me believe that axiom (5) above is OK is that it corresponds to the main requirement of order-sorted algebra, namely *regularity*. Let me elaborate.

4.1 Order-Sorted Regularity (or pre-regularity?)

In Order-Sorted Algebra(OSA), you have a partial order $(S, \leq)$ on sorts S. Symbols are allowed to be overloaded, but they must obey the *regularity* property in order for things to make sense mathematically:

Definition 4.1. (S, Σ') is regular iff for any $\sigma : s_1 \times \cdots \times s_n \to s$ and $\sigma : s'_1 \times \cdots \times s'_n \to s'$ such that $s_1 \leq s'_1, \ldots, s_n \leq s'_n$ we have $s \leq s'$.

In our setting, since we disallow overloaded symbols except for parametric ones, and since we build our subset relation constructively through parametric sorts starting with a base subsort relation, I dare to claim two things:

(a) That the axiom in (5) above corresponds to regularity in OSA, which is therefore a reasonable thing to have;

(b) Under a mild restriction, that in each

$$\texttt{symbol}\ \sigma\{P_1, \ldots, P_n\}(S_1, \ldots, S_n) : S$$

the sorts $S_1, \ldots, S_n$ already refer to *all* parameters $P_1, \ldots, P_k$, we can show that the resulting order-sorted signature, whose only overloaded symbols are the parametric ones, is *regular*. The "resulting" partial order on sorts is the least relation $\leq$ closed under the following.

- `syntax` $S' ::= S$ implies $S \leq S'$
  Dorel: S and S' basic sorts?

  Brandon: Good point, what happens if these are instances of parametric sorts?
- reflexive and transitive
- If $s_1 \leq s'_1, \ldots, s_k \leq s'_k$ and $sort\{P_1, \ldots, P_k\}$ is a parametric sort, then $sort\{s_1, \ldots, s_k\} \leq sort\{s'_1, \ldots, s'_k\}$.

Dorel: The following definitions are from [GM92]:

Definition 4.2. An *order-sorted signature* is a triple $(S, \leq, \Sigma)$ such that (S, Σ) is a many-sorted signature, $(S, \leq)$ is a poset, and the operations satisfy the following *monotonicity condition*:

$$\sigma \in \Sigma_{w,s} \cap \Sigma_{w',s'} \text{ and } w \leq w' \text{ imply } s \leq s'.$$

Remark 4.1. If $w = s_1 \ldots s_m$ and $w' = s'_1 \ldots s'_n$ then $w \leq w'$ iff $m = n$ and $s_i \leq s'_i$ for $i = 1, \ldots, n$.

Definition 4.3. An order-sorted signature $(S, \leq, \Sigma)$ is *regular* iff

- given $\sigma \in \Sigma_{w',s'}$ and $w_0 \leq w'$ there is a least rank $\langle w, s\rangle \in S^* \times S$ such that $w_0 \leq w$ and $\sigma \in \Sigma_{w,s}$.

An order-sorted signature $(S, \leq, \Sigma)$ is *pre-regular* iff

- given $\sigma \in \Sigma_{w',s'}$ and $w_0 \leq w'$ there is a least sort $s \in S$ such that $\sigma \in \Sigma_{w,s}$ for some w with $w_0 \leq w$.

Fact 4.1. *[GM92] Let $(S, \leq, \Sigma)$ be an order-sorted signature such that $(S, \leq)$ is* coNoetherian*, i.e., there is no strictly decreasing infinite chain $s_1 > s_2 > s_3 > \cdots$. Then $(S, \leq, \Sigma)$ is regular iff whenever $\sigma \in \Sigma_{w',s'} \cap \Sigma_{w'',s''}$ and $w_0 \leq w', w''$ then there is some $w \leq w', w''$ such that $\sigma \in \Sigma_{w,s}$ and $w_0 \leq w$.*

Definition 4.4. Let $(S, \leq, \Sigma)$ be an order-sorted signature. An $(S, \leq, \Sigma)$-*algebra* is a many-sorted (S, Σ)-algebra M satisfying:

1. $s \leq s'$ implies $M_s \subseteq M_{s'}$;
2. $\sigma \in \Sigma_{w',s'} \cap \Sigma_{w'',s''}$ and $w' \leq w''$ implies $M_\sigma : M_{w'} \to M_{s'}$ equals $M_\sigma : M_{w''} \to M_{s''}$ on $M_{w'}$,

where if $w = s_1 \ldots s_n$ then $M_w = M_{s_1} \times \cdots \times M_{s_n}$.

I think that the "aggressive equation" **??** formalizes in fact the second condition in the above definition.

Dorel: Now we sketch out how a front-end grammar G defines an order-sorted signature $(S^{\mathrm{osa}}, \leq, \Sigma^{\mathrm{osa}})$:

1. each parametric sort declaration

   $$\texttt{sort}\ N\{P_1, \ldots, P_k\}$$

   together with each parameter valuation $\theta : \{P_1, \ldots, P_k\} \to S^{\mathrm{osa}}$ i define a sort $\theta(N) = N\{\theta(P_1), \ldots, \theta(P_k)\}$ in S^{osa}.
   If $k = 0$ then there is a unique function $\theta : \emptyset \to S^{\mathrm{osa}}$ and we identify $\theta(N)$ with N (or $N\{\}$?);

2. each parametric production π

   $$\texttt{syntax}\{P_1, \ldots, P_k\}\ N ::= T_1 N_1 \ldots T_n N_n T_{n+1}$$

   where $n > 1$ or at least T_i is diferent from "", and each parameter valuation $\theta : \{P_1, \ldots, P_k\} \to S^{\mathrm{osa}}$ define

   (a) a sort $\theta(N) = N\{\theta(P_1), \ldots, \theta(P_k)\}$ in S^{osa} and

   (b) a symbol σ_π in $\Sigma^{\mathrm{osa}}_{\theta(N_1)\ldots\theta(N_n),\theta(N)}$.

   Similar to above, if $k = 0$ we identify $\theta(N)$ with N;

3. each parametric production

   $$\texttt{syntax}\{P_1, \ldots, P_k\}\ N ::= N_1$$

   and each parameter valuation $\theta : \{P_1, \ldots, P_k\} \to S^{\mathrm{osa}}$ define

   (a) a sort $\theta(N) = N\{\theta(P_1), \ldots, \theta(P_k)\}$ in S^{osa} and

   (b) a relation $\theta(N_1) \leq \theta(N)$.

   Again, if $k = 0$ we identify $\theta(N)$ with N.

4. each parametric production

   $$\texttt{syntax}\{P_1, \ldots, P_k\}\ N ::= \ldots$$

   and each pair of parameter valuations $\theta, \theta' : \{P_1, \ldots, P_k\} \to S^{\mathrm{osa}}$ such thar $\theta(P_i) \leq \theta'(P_i)$, for $i = 1, \ldots, n$, define

(a) a relation $\theta(N) \leq \theta'(N)$.

5. $\leq$ is extended such that it is reflexive and transitive.

In order to show that $(S^{\rm osa}, \leq, \Sigma^{\rm osa})$ is regular, we need a formal definition for the nonterminals occurring in a parametric production.

Definition 4.5. Let P denote the set of parameters $P = \{P_1, \ldots, P_k\}$. A *P-sort-term* is inductively defined as follows:

- a simple sort name (identifier) is a P-sort-name;
- a parameter P_i is a P-sort-name;
- if $N\{Q_1, \ldots, Q_\ell\}$ is a parametric sort name and $N_1, \ldots, N_\ell$ are P-sort-names, then $N\{N_1, \ldots, N_\ell\}$ is a P-sort-name.

In a parametric production

$$\texttt{syntax}\{P_1, \ldots, P_k\}\ N ::= T_1 N_1 \ldots T_n N_n T_{n+1}$$

we have:

1. N is a simple sort name (identifier);
2. N_i is is a $\{P_1, \ldots, P_k\}$-sort-name, $i = 1, \ldots, k$.

Lemma 4.1. *Let N be a P-sorted-term, where $P = \{P_1, \ldots, P_k\}$ denotes the set of parameters occuring in N. If $\theta, \theta' : P \to S^{\rm osa}$ such that $\theta(N) \leq \theta'(N)$, then $\theta(P_i) \leq \theta(P_i')$ for $i = 1, \ldots, n$.*

Proof. By structural induction on N. □

I do not know yet whether the following theorem holds and, if yes, how to prove it.

Theorem 4.1. *Let G be a input front-end grammar and $(S^{osa}, \leq, \Sigma^{osa})$ the order-sorted signature associated to G as above. Then $(S^{osa}, \leq, \Sigma^{osa})$ is regular.*

Having the order-sorted signature $(S^{\rm osa}, \leq, \Sigma^{\rm osa})$ associated to an input grrama G, it is quite easy to define a ML theory (S, Σ, F) encoding the order-sorted one:

1. $S = S^{\rm osa}$ (without partial order relation);
2. each overloaded symbol $\sigma \in \theta(N_1) \ldots \theta(N_n), \theta'(N)$ becomes a parametric one: $\sigma\{\theta(N_1) \ldots \theta(N_n), \theta'(N)\} \in \theta(N_1) \ldots \theta(N_n), \theta'(N)$
3. injections describe the partial order relation: $\texttt{inj}\{\theta(N), \theta(N')\} \in \Sigma_{\theta(N),\theta(N')}$ iff $\theta(N) \leq \theta(N')$;
4. the axioms F are the above ones (perhaps with a small different notation).

References

[GM92] Joseph A. Goguen and Jos Meseguer. 1992. *Order-sorted algebra I: equational deduction for multiple inheritance, overloading, exceptions and partial operations.* Theor. Comput. Sci. 105, 2 (November 1992), 217-273.