On Injections

Formal Systems Laboratory
University of Illinois at Urbana-Champaign

July 2, 2018

Written by Grigore on paper and edited by Radu Mereuta. This is supposed to start a conversation on injections in K and eventually be integrated into semantics-of-k.

1 Injections

Recall parametric productions (in front-end):

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

The non-terminals $N, N_1, \dots, N_n$ can make use of parameters $P_1, \dots, P_k$. We also discussed about allowing the parameters to be optionally specified among the terminals $T_1, \dots, T_n, T_{n+1}$ and non-terminals $N_1, \dots, N_n$, e.g. $T_1\{P_1\}N_1 \dots$, 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:

$$\sigma\{P_1, \dots, P_n\}(N_1, \dots, N_n) : N$$

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

$$syntax\{P_1, \dots, P_n\}N ::= N_1$$

As a concrete example, assume the following:

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

By abuse of notation let

$$inj_{N_1,N}\{P_1, \dots, P_k\}(N_1) : N$$

be the parametric symbol corresponding to the generic injection production above. Note that for now inj is <u>not</u> parametric in the two sorts! It is defined like an ordinary symbol. For our example:

$$inj_{Map\{Int\{\},V\},MapInt\{V\}}\{V\}(Map\{Int\{\}, V\}) : MapInt\{V\}$$

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 $P_1, \dots, P_k$. Then any element of sort $\theta(N_1)$ is expected to also be found among the elements of sort $\theta(N)$, possibly renamed. Moreover, such injections are expected to be consistent.

1. For example, assume another production:

$$syntax\{Q_1, \dots, Q_L\}\ M ::= M_1$$

with corresponding injection:

$$inj_{M_1,M}\{Q_1, \dots, Q_L\}(M_1) : M$$

such that there is some instance ρ of the parameters $Q_1, \dots, Q_L$ such that $\rho(M_1) = \theta(N1)$ and $\rho(M) = \theta(N)$. Then we certainly want the following to hold:

$$inj_{N_1,N}\{\theta(\bar{P})\}(\phi : \theta(N_1)) = inj_{M_1,N}\{\rho(\bar{Q})\}(\phi)$$

2. As another example, assume productions:

$$\begin{array}{ll} syntax\{A_1, \dots, A_a\}\ X ::= X_1 & \alpha \\ syntax\{B_1, \dots, B_b\}\ Y_1 ::= Y_2 & \beta \\ syntax\{C_1, \dots, C_c\}\ Z ::= Z_2 & \gamma \end{array}$$

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

$$inj_{X_1,X}\{\alpha(\bar{A})\}(inj_{Y_2,Y_1}\{\beta(\bar{B})\}(\phi : \beta(Y_2))) = inj_{Z_2,Z}\{\gamma(\bar{C})\}(\phi)$$

We also want parametric sorts to lift subsorts.

3. That is, if

$$inj_{N_1,N}\{P_1, \dots, P_k\}(N_1) : N$$

is an injection parametric symbol corresponding to production $syntax\{P_1, \dots, P_k\}\ N ::= N_1$, then we also want to have parametric symbols

$$inj_{sort\{N_1\},sort\{N\}}\{P_1, \dots, P_k\}(sort\{N_1, \dots\}) : sort\{N, \dots\}$$

for any other parametric sort $sort\{\_, \dots\}$.

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:

$$\begin{array}{l} syntax\ Exp ::= Int \\ syntax\{S\}\ List\{S\} ::= cons(S, List\{S\}) \end{array}$$

or in KORE:

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

Then we need to add the following axiom:

$$\begin{aligned}&\mathit{inj}_{\mathit{List}\{\mathit{Int}\},\mathit{List}\{\mathit{Exp}\}}(\mathit{cons}\{\mathit{Int}\}(\phi:\mathit{Int},\psi:\mathit{List}\{\mathit{Int}\}))\\&\quad=\mathit{cons}\{\mathit{Exp}\}(\mathit{inj}_{\mathit{Int},\mathit{Exp}}(\phi),\mathit{inj}_{\mathit{List}\{\mathit{Int}\},\mathit{List}\{\mathit{Exp}\}}(\psi))\end{aligned}$$

5. But what if the parametric symbol does not depend on the parameter, that is, say:

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

Then we need to add an axiom as follows:

$$\mathit{length}\{\mathit{Int}\}(\phi:\mathit{List}\{\mathit{Int}\})=\mathit{length}\{\mathit{Exp}\}(\mathit{inj}_{\mathit{List}\{\mathit{Int}\},\mathit{List}\{\mathit{Exp}\}}(\phi))$$

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:

$$\mathit{symbol}\ \mathit{cast}\{P_1,P_2\}(P_1):P_2$$

while it makes sense to add axioms like:

$$\mathit{inj}_{\mathit{Int},\mathit{Exp}}(\mathit{cast}\{\mathit{Int},\mathit{Int}\}(\phi:\mathit{Int}))=\mathit{cast}\{\mathit{Int},\mathit{Exp}\}(\phi)$$

you cannot instantiate P_2 with everything, eg., you cannot add:

$$\mathit{inj}_{\mathit{Int},\mathit{String}}(\mathit{cast}\{\mathit{Int},\mathit{Int}\}(\phi:\mathit{Int}))=\mathit{cast}\{\mathit{Int},\mathit{String}\}(\phi)$$

So things are really tricky. What I propose is to add the following axioms for now, and <u>in parallel</u> 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!

1.1 Proposal for injections

1. Define/assume one parametric symbol

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

This is what we do now too. Note that this is <u>different</u> from having one *inj* symbol for each subsorting. In particular, for the subsorting discussed at the beginning of this writing

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

instead of the injection label we had there, namely:

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

we refer to this subsorting using the generic injection:

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

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":

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

I don't think that we want to axiomatize that inj is an injective function, because we may want to inject sets of larger cardinals (e.g. identifiers) into sets of smaller cardinals (e.g. $Bool(x\ or\ y)\ and\ x = x$).

3. Axiomatize that inj is reflexive:

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

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

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

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:

$$\begin{aligned} axiom\{P_1, \ldots, P_k, P'_1, \ldots, P'_k\}\ & inj\{S, S'\}(\sigma\{P_1, \ldots, P_k\}(\phi_1 : S_1, \ldots, \phi_n : S_n)) \\ &= \sigma\{P'_1, \ldots, P'_n\}(inj\{S1, S'_1\}(\phi_1), \ldots, inj\{S_k, S'_k\}(\phi_k)) \end{aligned}$$

where $S'_1, \ldots, S'_n$ are $S_1[P'_1/P_1, \ldots, P'_k/P_k], \ldots, S_n[P'_1/P_1, \ldots, P'_k/P_k]$, and $S[P'_1/P_1, \ldots, P'_k/P_k]$.

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 above in 6 above. And I would agree, but I do believe that we should either disallow symbols like in 6 above, or otherwise give a serious warning. Note that inj itself is such a symbol :-)

What makes me believe that axiom 4 above is OK is that it corresponds to the main requirement of order-sorted algebra, namely regularity. Let me elaborate.

1.2 Order-sorted regularity (or pre-regularity?)

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

Def: (S, Σ) 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 subsort relation constructively through parametric sorts starting with a base subsort relation, I dare to claim 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 $\sigma\{P_1,\ldots,P_n\}(S_1,\ldots,S_n) : S$ the sorts $S_1,\ldots,S_n$ already refer to all parameters $P_1,\ldots,P_n$, 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 $S \leq S'$
- transitive and reflexive
- if $S_1 \leq S'_1,\ldots,S_k \leq S'_k$ and $Sort\{P_1,\ldots,P_k\}$ is a parametric sort, then $Sort\{S_1,\ldots,S_k\} \leq Sort\{S'_1,\ldots,S'_k\}$.