1 Plastic buffer

In this section, we describe briefly 1. *the specification method*, and 2. *the verification technique* we intend to build on top of the foundation developed in the present contribution.

1.1 Specification method

We envision a specification methodology where the rigid data types are built outside the hybrid specification. For example, a hybrid specification in HFOLR has a signature $(\mathsf{Nom}, \Lambda, \Sigma^{\mathtt{r}} \subseteq \Sigma)$, where $\Sigma^{\mathtt{r}} = (S^{\mathtt{r}}, F^{\mathtt{r}}, P^{\mathtt{r}})$ and $\Sigma = (S, F, P)$. Practitioners will start by specifying the rigid data types, i.e. a first-order specification with the signature $\Sigma^{\mathtt{r}}$. This is followed by the definition of (a) nominals, (b) accessibility relations between states, and (c) flexible data types, in such a way that no 'junk' and no 'confusion' are added to the rigid data types, i.e. the $\Sigma^{\mathtt{r}}$-models previously defined are preserved. For the sake of simplicity, in practice, a variable is identified only by its name; by a slight abuse of notation, for each inclusion $\chi\colon \Delta \hookrightarrow \Delta'$ and any Δ-sentence γ, we let γ denote $\chi(\gamma)$.

In this paper, only basic specifications SP are considered, that is $\mathsf{SP} = (\mathsf{Sig}(\mathsf{SP}), \mathsf{Sen}(\mathsf{SP}))$, where $\mathsf{Sig}(\mathsf{SP})$ is a signature and $\mathsf{Sen}(\mathsf{SP})$ is a set of sentences over $\mathsf{Sig}(\mathsf{SP})$.

Example 1. *We define the following specification of lists in* FOL*:*

> *spec* LIST
>
> *sorts* Elt List
>
> *op* $\mathtt{err}\colon\ \to \mathsf{Elt}$
>
> *op* $\mathtt{empty}\colon\ \to \mathsf{List}$
>
> *op* $\_\,{\scriptstyle ⨟}\,\_\colon \mathsf{List}\mathsf{Elt} \to \mathsf{List}$
>
> *vars* L $\mathsf{Q} : \mathsf{List}$
>
> *var* $\mathsf{F} : \mathsf{Elt}$
>
> *eq-1* $\forall \mathsf{L} \cdot \mathsf{L} ⨟ \mathtt{err} = \mathsf{L}$
>
> *eq-2* $\forall \mathsf{L} \cdot \mathsf{L} = \mathtt{empty} \vee \exists \mathsf{Q}, \mathsf{F} \cdot \mathsf{L} = \mathsf{Q} ⨟ \mathsf{F}$

□

The constant `err` can be regarded as an error element which can be used to describe partial functions on lists. According to the first sentence, `err` should not be regarded as an element of any list. By the second sentence, a list is either empty or it is obtained from another list by adding one element. The specification LIST provides the rigid data types for the hybrid specification presented next.

Example 2. *The hybrid specification* BUFFER *defined below consists of a buffer with two distinct operation modes: (a) 'lifo', where it behaves as a stack, and (b) 'fifo', where it behaves as a queue. The alternation of configurations is triggered by an event 'shift'.*

> *spec* $\mathsf{BUFFER}[\mathsf{LIST}]$
>
> *nominals* `lifo` `fifo`
>
> *modality* $\mathtt{shift} : 2$
>
> *op* $\mathtt{read} : \mathsf{List} \to \mathsf{Elt}$
>
> *op* $\mathtt{del} : \mathsf{List} \to \mathsf{List}$
>
> *vars* $\mathtt{E}$ $\mathsf{F} : \mathsf{Elt}$
>
> *var* $\mathsf{L} : \mathsf{List}$
>
> *rel-1* $@_{\mathtt{fifo}}\, \underline{\mathtt{shift}}(\mathtt{lifo})$
>
> *rel-2* $@_{\mathtt{lifo}}\, \underline{\mathtt{shift}}(\mathtt{fifo})$
>
> *eq-3* $\mathtt{read}(\mathtt{empty}) = \mathtt{err}$
>
> *eq-4* $\forall \mathsf{L}, \mathtt{E} \cdot (@_{\mathtt{lifo}}\mathtt{read})(\mathsf{L} ⨟ \mathtt{E}) = \mathtt{E}$

eq-5 $\forall \mathtt{E} \cdot (@_{\mathtt{fifo}}\mathtt{read})(\mathtt{empty} \,⨾\, \mathtt{E}) = \mathtt{E}$

eq-6 $\forall \mathsf{L}, \mathtt{E}, \mathsf{F} \cdot (@_{\mathtt{fifo}}\mathtt{read})(\mathsf{L} \,⨾\, \mathtt{E} \,⨾\, \mathsf{F}) = (@_{\mathtt{fifo}}\mathtt{read})(\mathsf{L} \,⨾\, \mathtt{E})$

eq-7 $\mathtt{del}(\mathtt{empty}) = \mathtt{empty}$

eq-8 $\forall \mathsf{L}, \mathtt{E} \cdot (@_{\mathtt{lifo}}\mathtt{del})(\mathsf{L} \,⨾\, \mathtt{E}) = \mathsf{L}$

eq-9 $\forall \mathtt{E} \cdot (@_{\mathtt{fifo}}\mathtt{del})(\mathtt{empty} \,⨾\, \mathtt{E}) = \mathtt{empty}$

eq-10 $\forall \mathsf{L}, \mathtt{E}, \mathsf{F} \cdot (@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L} \,⨾\, \mathtt{E} \,⨾\, \mathsf{F}) = (@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L} \,⨾\, \mathtt{E}) \,⨾\, \mathsf{F}$

□

The REL component of the hybrid signature consists of two nominals `fifo` and `lifo` and one binary modality `shift`. The signature of rigid symbols is the signature of LIST. There are two flexible operation symbols, `del` and `read`.

The system has two operation modes, `lifo`, when it behaves like a stack, and `fifo`, when it behaves like a queue. The binary modality `shift` makes the transition between `lifo` and `fifo` modes according to the nominal relations $@_{\mathtt{fifo}}\, \underline{\mathtt{shift}}(\mathtt{lifo})$ and $@_{\mathtt{lifo}}\, \underline{\mathtt{shift}}(\mathtt{fifo})$. The function symbol `read` denotes the operation which returns the top/front of a stack/queue, and `del` denotes the operation 'pop'. Notice that `read` and `del` play different roles in each operation mode.

The models of BUFFER consists of all Kripke structures (W, M) over the signature of BUFFER which (a) have a rigid LIST structure, that is $M_{\mathtt{lifo}} \upharpoonright_{\mathsf{Sig(LIST)}} = M_{\mathtt{fifo}} \upharpoonright_{\mathsf{Sig(LIST)}}$ is a model of LIST, and (b) satisfy the axioms defined in Example 2. This construction is particularly useful for structured specifications which are obtained from basic specifications by applying specification building operators such as union, translation, hiding or freeness. As for Example 2, notice that any Kripke structure over Sig(BUFFER) which satisfies Sen(BUFFER) (i.e. all sentences defined in Example 1 and Example 2) is a model of BUFFER.

1.2 Formal verification

This section is dedicated to proving that BUFFER satisfies the following property:

$$\forall \mathsf{L} \cdot (@_{\mathtt{lifo}}\mathtt{del})((@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L})) = (@_{\mathtt{fifo}}\mathtt{del})((@_{\mathtt{lifo}}\mathtt{del})(\mathsf{L}))$$

This means that the order of deleting the front and the top element from a list is irrelevant w.r.t. the final result. In order to implement efficient proof strategies, one often needs to derive new proof rules from the original ones.

$$(\mathit{Ref})\ \frac{}{\Gamma \vdash \forall X \cdot t = t} \qquad (\mathit{Sym})\ \frac{\Gamma \vdash \forall X \cdot t_1 = t_2}{\Gamma \vdash \forall X \cdot t_2 = t_1}$$

$$(\mathit{Trans})\ \frac{\Gamma \vdash \forall X \cdot t_1 = t_2 \quad \Gamma \vdash \forall X \cdot t_2 = t_3}{\Gamma \vdash \forall X \cdot t_1 = t_3} \qquad (\mathit{Rew})\ \frac{\Gamma \vdash \forall X \cdot t_1 = t_2}{\Gamma \vdash \forall Y \cdot t[\theta(t_1)_p] = t[\theta(t_2)]_p}\ [\ \theta\colon X \to T_{@\Sigma}(Y)\ ]$$

Table 1. Derived proof rules for HFOLR

Since the present contribution is not dedicated to the presentation of a formal method, minimally, we define a system of proof rules in Table 1, which allows one to avoid complex formal proofs for obvious properties. Notice that $e[t_1 \leftarrow t_2]$ is the sentence obtained from e by substituting t_2 for t_1, while $t|_p$ is the subterm of t at position p and $t[\theta(t_i)]_p$ is the term obtained from t by substituting $\theta(t_i)$ for $t|_p$ at position p.

For the sake of simplicity, we denote by Δ the signature Sig(BUFFER), by Γ the set of sentences Sen(BUFFER), and by PR(L) the formula $(@_{\mathtt{lifo}}\mathtt{del})((@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L})) = (@_{\mathtt{fifo}}\mathtt{del})((@_{\mathtt{lifo}}\mathtt{del})(\mathsf{L}))$.

Lemma 1. *We assume that the variable* L *is of sort* List*, and the variables* E *and* F *are of sort* Elt.

1. $\Gamma \vdash \mathsf{PR}(\mathtt{empty})$;
2. $\Gamma \vdash \forall \mathtt{E} \cdot \mathsf{PR}(\mathtt{empty} \,⨾\, \mathtt{E})$;
3. $\Gamma \vdash \forall \mathsf{L}, \mathtt{E}, \mathsf{F} \cdot \mathsf{PR}(\mathsf{L} \,⨾\, \mathtt{E} \,⨾\, \mathsf{F})$;
4. $\Gamma \vdash \forall \mathsf{L}, \mathtt{E} \cdot \mathsf{PR}(\mathsf{L} \,⨾\, \mathtt{E})$;

5. $\Gamma \vdash \forall \mathsf{L} \cdot \mathsf{PR}(\mathsf{L})$.

Proof. The first two assertions are straightforward to prove. We start with the third assertion.

1	$\Gamma \vdash \forall \mathsf{L}, \mathtt{E}, \mathsf{F} \cdot (@_{\mathtt{lifo}}\mathtt{del})((@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E} \fatsemi \mathsf{F})) = (@_{\mathtt{lifo}}\mathtt{del})((@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E}) \fatsemi \mathsf{F})$	by (*Rew*) from eq-10
2	$\Gamma \vdash \forall \mathsf{L}, \mathtt{E}, \mathsf{F} \cdot (@_{\mathtt{lifo}}\mathtt{del})((@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E}) \fatsemi \mathsf{F}) = (@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E})$	by (*Rew*) from eq-8
3	$\Gamma \vdash \forall \mathsf{L}, \mathtt{E}, \mathsf{F} \cdot (@_{\mathtt{fifo}}\mathtt{del})((@_{\mathtt{lifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E} \fatsemi \mathsf{F})) = (@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E})$	by (*Rew*) from eq-8
4	$\Gamma \vdash \forall \mathsf{L}, \mathtt{E}, \mathsf{F} \cdot (@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E}) = (@_{\mathtt{fifo}}\mathtt{del})((@_{\mathtt{lifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E} \fatsemi \mathsf{F}))$	by (*Sym*) from 3
5	$\Gamma \vdash \forall \mathsf{L}, \mathtt{E}, \mathsf{F} \cdot (@_{\mathtt{lifo}}\mathtt{del})((@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E} \fatsemi \mathsf{F})) = (@_{\mathtt{fifo}}\mathtt{del})((@_{\mathtt{lifo}}\mathtt{del})(\mathsf{L} \fatsemi \mathtt{E} \fatsemi \mathsf{F}))$	by (*Trans*) from 1, 2 and 4

We prove the fourth assertion:

1	$\Gamma \vdash_{\Delta[\mathsf{L},\mathtt{E}]} \mathsf{L} = \mathtt{empty} \vee \exists \mathsf{Q}, \mathsf{F} \cdot \mathsf{L} = \mathsf{Q} \fatsemi \mathsf{F}$	from eq-2, since eq-2 is a rigid sentence
2	$\Gamma \cup \{\mathsf{L} = \mathtt{empty}\} \vdash_{\Delta[\mathsf{L},\mathtt{E}]} \mathsf{PR}[\mathsf{L} \fatsemi \mathtt{E}]$	
2.1	$\Gamma \vdash_{\Delta[\mathsf{L},\mathtt{E}]} \mathsf{PR}[\mathtt{empty} \fatsemi \mathtt{E}]$	by (*Rew*) from the second assertion
2.2	$\Gamma \cup \{\mathsf{L} = \mathtt{empty}\} \vdash_{\Delta[\mathsf{L},\mathtt{E}]} \mathsf{PR}[\mathtt{empty}; \mathtt{E}]$	by *(Transitivity)* and *(Monotonicity)*
2.3	$\Gamma \cup \{\mathsf{L} = \mathtt{empty}\} \vdash_{\Delta[\mathsf{L},\mathtt{E}]} \mathsf{PR}[\mathsf{L} \fatsemi \mathtt{E}]$	by (*Rew*) and (*Trans*) from 2.2
3	$\Gamma \cup \{\exists \mathsf{Q}, \mathsf{F} \cdot \mathsf{L} = \mathsf{Q} \fatsemi \mathsf{F}\} \vdash_{\Delta[\mathsf{L},\mathtt{E}]} \mathsf{PR}[\mathsf{L} \fatsemi \mathtt{E}]$	
3.1	$\Gamma \cup \{\mathsf{L} = \mathsf{Q} \fatsemi \mathsf{F}\} \vdash_{\Delta[\mathsf{L},\mathsf{Q},\mathtt{E},\mathsf{F}]} \mathsf{PR}[\mathsf{Q} \fatsemi \mathsf{F} \fatsemi \mathtt{E}]$	by (*Rew*) from the third assertion
3.2	$\Gamma \cup \{\mathsf{L} = \mathsf{Q} \fatsemi \mathsf{F}\} \vdash_{\Delta[\mathsf{L},\mathsf{Q},\mathtt{E},\mathsf{F}]} \mathsf{PR}[\mathsf{L} \fatsemi \mathtt{E}]$	by (*Rew*) and (*Trans*) from 3.1
3.3	$\Gamma \cup \{\exists \mathsf{Q}, \mathsf{F} \cdot \mathsf{L} = \mathsf{Q} \fatsemi \mathsf{F}\} \vdash_{\Delta[\mathsf{L},\mathtt{E}]} \mathsf{PR}[\mathsf{L} \fatsemi \mathtt{E}]$	by (*Quant*$_I$) from 3.2
4	$\Gamma \vdash_{\Delta[\mathsf{L},\mathtt{E}]} \mathsf{PR}(\mathsf{L} \fatsemi \mathtt{E})$	by (*Disj*$_E$) from 1, 2 and 3
5	$\Gamma \vdash_{\Delta} \forall \mathsf{L}, \mathtt{E} \cdot \mathsf{PR}(\mathsf{L} \fatsemi \mathtt{E})$	from 4, $\forall \mathsf{L}, \mathtt{E} \cdot \mathsf{PR}[\mathsf{L} \fatsemi \mathtt{E}]$ is a rigid sentence

The proof of the fifth assertion resembles the proof of the fourth assertion. □

It is worth noting that the expressivity of the basic layer of HFOLR allows a simple description of the property to prove $\forall \mathsf{L} \cdot (@_{\mathtt{lifo}}\mathtt{del})((@_{\mathtt{fifo}}\mathtt{del})(\mathsf{L})) = (@_{\mathtt{fifo}}\mathtt{del})((@_{\mathtt{lifo}}\mathtt{del})(\mathsf{L}))$. The same property can be expressed in HFOLS as follows: $\forall \mathsf{L} \cdot \exists x_1, x_2, y_1, y_2 \cdot \ x_1 = y_1 \wedge @_{\mathtt{lifo}}(x_1 = \mathtt{del}(x_2))) \wedge @_{\mathtt{fifo}}(x_2 = \mathtt{del}(\mathsf{L})) \wedge @_{\mathtt{fifo}}(y_1 = \mathtt{del}(y_2))) \wedge @_{\mathtt{lifo}}(y_2 = \mathtt{del}(\mathsf{L}))$, where x_i, y_i are variables of sort List. It is not difficult to see that the expressivity of HFOLR_b has deep ramifications in formal verification; for example, the proofs become much simpler.