1 Modeling

Formal specification of a buffer with an infinite number of states.

1.1 Rigid data types

Example 1. *Specification of lists of arbitrary elements:*

```
spec! LISTS
pr BOOL
sorts Elt List .
op empty : -> List .
op _⨾_ : Elt List -> List .
op _in_ : Elt List -> Bool .
vars E E' : Elt .
var L : List .
```

(1) $\forall \mathtt{E} \cdot \mathtt{E\ in\ empty} = \mathtt{false}$

(2) $\forall \mathtt{E}, \mathtt{E}' \cdot (\mathtt{E\ in\ E}' \fatsemi \mathtt{L})\ \ \mathtt{if}\ \mathtt{E} = \mathtt{E}'$

(3) $\forall \mathtt{E}, \mathtt{E}' \cdot \mathtt{E\ in\ E}' \fatsemi \mathtt{L} = \mathtt{E\ in\ L}\ \mathtt{if}\ \neg(\mathtt{E} = \mathtt{E}')$

1.2 Nominals

Example 2. *Specification of nominals:*

```
spec! NOMINAL
sort Nominal .
op init : -> Nominal .
op next : Nominal -> Nominal .
```

1.3 Flexible data types

Example 3. *Specification of the attributes* `read` *and* `del`*:*

```
spec BUFFER[LISTS,NOMINAL]
op read : List -> [Elt] .
op del : List -> List .
var E : Elt .
var L : List .
var Z : Nominal .
```

(4) $\forall \mathtt{Z} \cdot (@_{\mathtt{Z}}\, \mathtt{del})(\mathtt{empty}) = \mathtt{empty}$

(5) $\forall \mathtt{E}, \mathtt{L} \cdot (@_{\mathtt{init}}\, \mathtt{read})(\mathtt{E} \fatsemi \mathtt{L}) = \mathtt{E}$

(6) $\forall \mathtt{E}, \mathtt{L} \cdot (@_{\mathtt{init}}\, \mathtt{del})(\mathtt{E} \fatsemi \mathtt{L}) = \mathtt{L}$

(7) $\forall \mathtt{Z}, \mathtt{E}, \mathtt{L} \cdot (@_{\mathtt{next(Z)}}\, \mathtt{read})(\mathtt{E} \fatsemi \mathtt{L}) = (@_{\mathtt{Z}}\, \mathtt{read})(\mathtt{L})$

(8) $\forall \mathtt{Z}, \mathtt{E}, \mathtt{L} \cdot (@_{\mathtt{next(Z)}}\, \mathtt{del})(\mathtt{E} \fatsemi \mathtt{L}) = \mathtt{E} \fatsemi (@_{\mathtt{Z}}\, \mathtt{del})(\mathtt{L})$

2 Formal verification

The property we are interested in proving formally is $\Gamma \vdash_{\Sigma} \forall \mathtt{L}, \mathtt{E} \cdot \exists \mathtt{Z} \cdot (@_Z\, \mathtt{read})(\mathtt{L}) = \mathtt{E}\ \mathtt{if}\ (\mathtt{E}\ \mathtt{in}\ \mathtt{L}) = \mathtt{true}$, where $\Sigma = \mathsf{Sig}(\mathsf{BUFFER})$ and $\Gamma = \mathsf{Sen}(\mathsf{BUFFER})$. We proceed by induction on the structure of $\mathtt{L}$.

- ind. base: $\Gamma \vdash \forall \mathtt{E} \cdot \exists \mathtt{Z} \cdot (@_{\mathtt{Z}}\, \mathtt{read})(\mathtt{empty}) = \mathtt{E}\ \mathtt{if}\ (\mathtt{E}\ \mathtt{in}\ \mathtt{empty}) = \mathtt{true}$, which is true since $\mathtt{E}\ \mathtt{in}\ \mathtt{empty} = \mathtt{false}$
- ind. step: $\Gamma \cup \{\forall \mathtt{E} \cdot \exists \mathtt{Z} \cdot \mathtt{read}(\mathtt{Z}, \mathtt{l})\ \mathtt{if}\ (\mathtt{E}\ \mathtt{in}\ \mathtt{l}) = \mathtt{true}\} \vdash_{\mathsf{Sig}[\mathtt{l},\mathtt{e}]} \forall \mathtt{E} \cdot \exists \mathtt{Z} \cdot \mathtt{read}(\mathtt{Z}, \mathtt{l} ⨟ \mathtt{e}) = \mathtt{E}\ \mathtt{if}\ (\mathtt{E}\ \mathtt{in}\ \mathtt{l} ⨟ \mathtt{e}) = \mathtt{true}$, where $\mathtt{e} \colon \to \mathsf{Elt}$ and $\mathtt{l} \colon \to \mathsf{List}$