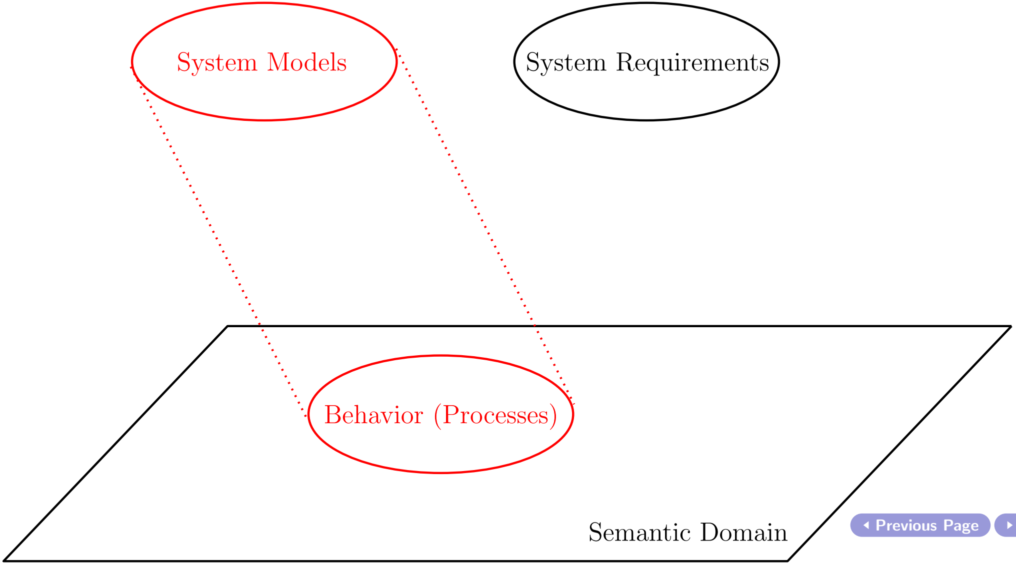
System Validation: Reasoning about Abstract Data Types

Mohammad Mousavi and Jeroen Keiren

General Overview



Example
Euro Sort (recap)

```
sort Euro;
cons zero, fifty_cents,
     one_euro, more: Euro;
     % constants: constructors with no parameter
map  eq: Euro × Euro → Bool;
     plus: Euro × Euro → Euro;
var  e:Euro;
eqn  eq(e, e)= true;                    (1)
     eq(zero, one_euro)= false;         (2)
     eq(one_euro, zero)= false;         (3)
     ...
```

Example
Euro Sort

Theorem. $\text{zero} \neq \text{one\_euro}$

Example
Euro Sort

Theorem. $\mathsf{zero} \neq \mathsf{one\_euro}$

Proof technique: proof by contradiction.

Proof of zero $\neq$ one_euro

Assume towards contradiction zero $=$ one_euro. Then, we have:

$$\text{true} = (1)$$

$$\begin{aligned} \text{eq}(e, e) &= \text{true}; & (1) \\ \text{eq}(\text{zero}, \text{one\_euro}) &= \text{false}; & (2) \\ \text{eq}(\text{one\_euro}, \text{zero}) &= \text{false}; & (3) \end{aligned}$$

Proof of zero $\neq$ one_euro

Assume towards contradiction
zero $=$ one_euro. Then, we have:

$$
\begin{aligned}
\text{true} &= && (1)\\
\text{eq(zero, zero)} &= && \text{(assump.)}
\end{aligned}
$$

$$
\begin{aligned}
\text{eq(e, e)} &= \text{true}; && (1)\\
\text{eq(zero, one\_euro)} &= \text{false}; && (2)\\
\text{eq(one\_euro, zero)} &= \text{false}; && (3)
\end{aligned}
$$

Proof of zero $\neq$ one_euro

Assume towards contradiction zero $=$ one_euro. Then, we have:

$$
\begin{aligned}
\text{true} &= && (1)\\
\text{eq(zero, zero)} &= && \text{(assump.)}\\
\text{eq(zero, one\_euro)} &= && (2)
\end{aligned}
$$

$$
\begin{aligned}
\text{eq(e, e)} &= \text{true}; && (1)\\
\text{eq(zero, one\_euro)} &= \text{false}; && (2)\\
\text{eq(one\_euro, zero)} &= \text{false}; && (3)
\end{aligned}
$$

Proof of zero $\neq$ one_euro

Assume towards contradiction zero $=$ one_euro. Then, we have:

$$\begin{aligned} \text{true} &= && (1) \\ \text{eq(zero, zero)} &= && \text{(assump.)} \\ \text{eq(zero, one\_euro)} &= && (2) \\ \text{false} \end{aligned}$$

$$\text{eq(e, e)} = \text{true}; \quad (1)$$
$$\text{eq(zero, one\_euro)} = \text{false}; \quad (2)$$
$$\text{eq(one\_euro, zero)} = \text{false}; \quad (3)$$

Example

Natural

```
sort Natural;
cons zero: Natural;
     succ: Natural → Natural;
map  eq: Natural × Natural → Bool;
var  i, j: Natural;
eqn  eq(i, i)= true;                          (1)
     eq(zero, succ(i))= false;                (2)
     eq(succ(i), zero)= false;                (3)
     eq(succ(i), succ(j))= eq(i,j);           (4)
```

Another theorem

Theorem. $\text{zero} \neq \text{succ}(i)$, for each Natural i.

Another theorem

Theorem. $\text{zero} \neq \text{succ}(i)$, for each Natural i.

Proof by contradiction.

Proof of zero $\neq$ succ(i)

Assume towards contradiction that for some Natural n, zero $=$ succ(n). Then, we have:

$$
\begin{aligned}
&\text{eq(i, i)} = \text{true;} && (1)\\
&\text{eq(zero, succ(i))} = \text{false;} && (2)\\
&\text{eq(succ(i), zero)} = \text{false;} && (3)\\
&\text{eq(succ(i), succ(j))} = \text{eq(i,j);} && (4)
\end{aligned}
$$

Proof of zero $\neq$ succ(i)

Assume towards contradiction that for some Natural n, zero $=$ succ(n). Then, we have:

$$\begin{aligned} \text{true} &= (1) \\ \text{eq(zero, zero)} & \end{aligned}$$

eq(i, i)= true; (1)
eq(zero, succ(i))= false; (2)
eq(succ(i), zero)= false; (3)
eq(succ(i), succ(j))= eq(i,j); (4)

Proof of zero $\neq$ succ(i)

Assume towards contradiction that for some Natural n, zero $=$ succ(n). Then, we have:

$$\begin{array}{lcl} \text{true} & = & (1) \\ \text{eq(zero, zero)} & = & \text{(assump.)} \\ \text{eq(zero, succ(n))} & & \end{array}$$

$$\begin{array}{ll} \text{eq(i, i)= true;} & (1) \\ \text{eq(zero, succ(i))= false;} & (2) \\ \text{eq(succ(i), zero)= false;} & (3) \\ \text{eq(succ(i), succ(j))= eq(i,j);} & (4) \end{array}$$

Proof of zero $\neq$ succ(i)

Assume towards contradiction that for some Natural n, zero = succ(n). Then, we have:

$$
\begin{aligned}
\text{true} &= (1) \\
\text{eq(zero, zero)} &= (\text{assump.}) \\
\text{eq(zero, succ(n))} &= (2) \\
\text{false}
\end{aligned}
$$

eq(i, i)= true;	(1)
eq(zero, succ(i))= false;	(2)
eq(succ(i), zero)= false;	(3)
eq(succ(i), succ(j))= eq(i,j);	(4)

Induction

Proof Rule

Thesis: $P(s)$ for each s of a given sort S.
Rule:

- prove $P(c)$ for each constant c of sort S.
- assuming that $P(x_i)$ holds (induction hypothesis, for each $0 \leq i < n$), prove $P(f(x_0, \ldots, x_{n-1}))$ for each n-ary constructor of sort S.

Example
Natural

$$
\begin{array}{lll}
\texttt{sort} & \text{Natural;} & \\
\texttt{cons} & \text{zero: Natural;} & \\
 & \text{succ: Natural} \to \text{Natural;} & \\
\texttt{map} & \text{eq: Natural} \times \text{Natural} \to \text{Bool;} & \\
\texttt{var} & \text{i, j: Natural;} & \\
\texttt{eqn} & \text{eq(i, i)} = \text{true;} & (1) \\
 & \text{eq(zero, succ(i))} = \text{false;} & (2) \\
 & \text{eq(succ(i), zero)} = \text{false;} & (3) \\
 & \text{eq(succ(i), succ(j))} = \text{eq(i,j);} & (4)
\end{array}
$$

Example
Natural

```
sort Natural;
cons zero: Natural;
     succ: Natural → Natural;
map  eq: Natural × Natural → Bool;
```

```
sort Natural;
cons zero: Natural;
     succ: Natural → Natural;
map  eq: Natural × Natural → Bool;
     plus: Natural × Natural → Natural;
var  i, j: Natural;
eqn  plus(zero, i)= i;                   (1)
     plus(i, zero)= i;                   (2)
     plus(i, succ(j))= succ(plus(i, j)); (3)
```

plus(succ(i), j) $=$ succ(plus(i,j))

Proof. By induction on j

$$
\begin{aligned}
\text{plus(zero, i)} &= \text{i}; \quad (1)\\
\text{plus(i, zero)} &= \text{i}; \quad (2)\\
\text{plus(i, succ(j))} &= \text{succ(plus(i, j))}; \quad (3)
\end{aligned}
$$

$\text{plus}(\text{succ}(i), j) = \text{succ}(\text{plus}(i,j))$

Proof. By induction on j

Induction basis: j = zero:

$\text{plus}(\text{succ}(i), \text{zero}) \quad =$

$$
\begin{aligned}
&\text{plus(zero, i)= i;} && (1)\\
&\text{plus(i, zero)= i;} && (2)\\
&\text{plus(i, succ(j))= succ(plus(i, j));} && (3)
\end{aligned}
$$

plus(succ(i), j) = succ(plus(i,j))

Proof. By induction on j

Induction basis: j = zero:

plus(succ(i), zero)	=	(2)
succ(i)	=	(2, from right to left)

plus(zero, i)= i;	(1)
plus(i, zero)= i;	(2)
plus(i, succ(j))= succ(plus(i, j));	(3)

plus(succ(i), j) $=$ succ(plus(i,j))

Proof. By induction on j

Induction basis: j = zero:

plus(succ(i), zero)	$=$	(2)
succ(i)	$=$	(2, from right to left)
succ(plus(i, zero))		

plus(zero, i)= i; (1)

plus(i, zero)= i; (2)

plus(i, succ(j))= succ(plus(i, j)); (3)

plus(succ(i), j) = succ(plus(i,j))

Proof. By induction on j

Induction basis: j = zero:

plus(succ(i), zero) = (2)

succ(i) = (2, from right to left)

succ(plus(i, zero))

Induction hypothesis, j = n:

assume that

plus(succ(i), n) = succ(plus(i,n));

plus(succ(i), succ(n)) = (3)

plus(zero, i)= i; (1)

plus(i, zero)= i; (2)

plus(i, succ(j))= succ(plus(i, j)); (3)

$\text{plus}(\text{succ}(i), j) = \text{succ}(\text{plus}(i,j))$

Proof. By induction on j

Induction hypothesis, j = n:

assume that

$\text{plus}(\text{succ}(i), n) = \text{succ}(\text{plus}(i,n))$;

Induction step, j = $succ(n)$:

prove that

$\text{plus}(\text{succ}(i), \text{succ}(n)) =$
$\quad \text{succ}(\text{plus}(i, \text{succ}(n)))$:

$\text{plus}(\text{succ}(i), \text{succ}(n))$	$=$	(3)
$\text{succ}(\text{plus}(\text{succ}(i), n))$	$=$	(ind. hyp.)

plus(zero, i)= i;	(1)
plus(i, zero)= i;	(2)
plus(i, succ(j))= succ(plus(i, j));	(3)

plus(succ(i), j) $=$ succ(plus(i,j))

Proof. By induction on j

Induction hypothesis, j = n:

assume that

plus(succ(i), n) $=$ succ(plus(i,n));

Induction step, j = $succ(n)$:

prove that

plus(succ(i), succ(n)) $=$
    succ(plus(i, succ(n))):

plus(zero, i)= i; (1)

plus(i, zero)= i; (2)

plus(i, succ(j))= succ(plus(i, j)); (3)

plus(succ(i), succ(n))	$=$	(3)
succ(plus(succ(i), n))	$=$	(ind. hyp.)
succ(succ(plus(i, n)))	$=$	(3, from right to left)

plus(succ(i), j) = succ(plus(i,j))

Proof. By induction on j

Induction hypothesis, j = n:
assume that
plus(succ(i), n) = succ(plus(i,n));
Induction step, j = $succ(n)$:
prove that
plus(succ(i), succ(n)) =
succ(plus(i, succ(n))):

plus(zero, i)= i; (1)
plus(i, zero)= i; (2)
plus(i, succ(j))= succ(plus(i, j)); (3)

plus(succ(i), succ(n))	=	(3)
succ(plus(succ(i), n))	=	(ind. hyp.)
succ(succ(plus(i, n)))	=	(3, from right to left)
succ(plus(i, succ(n)))		

Thank you very much.