Defining New Notions

Wolfgang Schreiner
Research Institute for Symbolic Computation (RISC-Linz)
Johannes Kepler University, Linz, Austria

Wolfgang.Schreiner@risc.uni-linz.ac.at
http://www.risc.uni-linz.ac.at/people/schreine

Mastering Complexity

Problem: Formulas become very large.

Complex objects and properties.

Solution: Introduce new function- and predicate-constants.

Capture under single name an object respectively property whose description with the help of the basic constants is very large.

- **Abstraction**
  - Refer to complex notion just by a name.
  - Not necessary to repeat its description over and over again.
- **Parameterization**
  - Refer to common core of similar notions by a name.
  - Use parameters to distinguish between different instances.

Theory

Definition: Theory

- Function- and predicate-constants.
- Formulas using these constants (axioms).

Formulas considered true in the domain described by this theory.

Example: Real Numbers

- $0, 1, +, *, \leq$
- $\forall x, y : x + y = y + x$

  $\ldots$

Formulas in basic theory become large and difficult to understand.

Extending Theories

Definition: definition

- A statement that introduces into a theory a new constant and additional axioms.
- Restriction: no formula can be derived in new theory that could not be derived from orginal theory (consistency).

Example: Real Numbers

- Unary function constant $|\ |$.
- $\forall x : 0 \leq x \Rightarrow |x| = x$
  $\forall x : x \leq 0 \Rightarrow |x| = -x$

Explicit Predicate Definitions

$$p(x_0, \ldots, x_{n-1}) :\Leftrightarrow F$$

- Introduction of new n-ary predicate constant p (definiendum).
- p must not occur within formula F (definiens).
- Free variables of F must be among $x_0, \ldots, x_{n-1}$.
- New axiom

$$\forall x_0, \ldots, x_{n-1} : p(x_0, \ldots, x_{n-1}) \Leftrightarrow F$$

New predicate is uniquely characterized.

Natural Language

A predicate definition in natural language is usually expressed as

x is (called) a P, if $\ldots$,

introducing the unary predicate $P(x)$, or

x and y are P, if $\ldots$

introducing the binary predicate $P(x, y)$, or

x is a P of y, if $\ldots$

which also introduces $P(x, y)$.

Not necessarily formal syntax.

Examples

- The statement

  p is prime, if its only divisors are 1 and p.

  introduces the unary predicate

  p $\underline{\text{is prime}} :\Leftrightarrow (\forall x : x \text{ is a divisor of } p \Rightarrow (x = 1 \lor x = p))$.

- The statement

  x is a divisor of y, if $x * z = y$, for some z.

  introduces the binary predicate

  x $\underline{\text{is a divisor of}}$ $y :\Leftrightarrow \exists z : x * z = y$.

Examples

- The statement

  x is prime $:\Leftrightarrow y \not| \, x$

  is not a definition, because free y does not occur on left side.

- The statement

  x is a divisor of $a * y$, if $x * z = a * y$, for some z.

  is not a definition, because on the left hand side of the formula

  x is a divisor of $a * y \Leftrightarrow \exists z : x * z = a * y$.

  a general term $a * y$ appears where only a variable is allowed.

Beware of bogus definitions!

Constrained Predicate Definitions

Global conditions:

- Let x be such that P. Then x is a Q if F holds.
- A P is a Q if F holds.

Translation:

$$Q(x) :\Leftrightarrow P(x) \wedge F$$

Implicit conditions have to be added to definition.

Example

The definition

Let f be a binary relation. f is a partial function if every element of its domain is in relation to at most one element of its range.

as well as the definition

A partial function is a binary relation where every element of the domain is in relation to at most one element of the range.

are to be read as the definition of a unary predicate

$$
\begin{aligned}
& f \text{ is a partial function} :\Leftrightarrow \\
& \quad f \text{ is a binary relation} \wedge \\
& \quad \forall x, y_0, y_1 : (\langle x, y_0 \rangle \in f \wedge \langle x, y_1 \rangle \in f) \Rightarrow y_0 = y_1.
\end{aligned}
$$

Explicit Function Definitions

$$f(x_0, \ldots, x_{n-1}) :\!= T$$

- Introduction of new n-ary function constant f (definiendum).
- f must not occur within term T (definiens).
- Free variables of T must be among $x_0, \ldots, x_{n-1}$.
- New axiom

$$\forall x_0, \ldots, x_{n-1} : f(x_0, \ldots, x_{n-1}) = T$$

New function is uniquely characterized.

Example

- The statement
$$\mathrm{add}(x, y) := x + y$$
defines a binary function add.
- The statement
$$\mathrm{add}(x) := x + y$$
is not a definition because of free variable y.
- The statement
$$\mathrm{add}(x * y) := x + y$$
is not a definition because $x * y$ is not a variable.

Conditional Function Definitions

Result defined in multiple ways:

$$
\begin{aligned}
f(x_0, \ldots, x_{n-1}) := \; & T_0, \ \textbf{if } F \\
& T_1, \ \textbf{otherwise}
\end{aligned}
$$

Translation:

$$f(x_0, \ldots, x_{n-1}) := \textbf{if } F \textbf{ then } T_0 \textbf{ else } T_1$$

Introduction of a new kind of terms.

Conditional Terms and Formulas

Definition: conditional term

$$(\textbf{if } F \textbf{ then } T_0 \textbf{ else } T_1)$$

- Formula F, terms T_0 and T_1
- Result is T_0, if F holds, and T_1, otherwise.

Definition: conditional formula

$$(\textbf{if } F \textbf{ then } F_0 \textbf{ else } F_1)$$

- Formulas F, F_0, F_1.
- Result is F_0, if F holds, and F_1, otherwise.

Example

The definition

$$|x| := \begin{array}{l} -x, \ \textbf{if}\ x < 0 \\ x, \ \textbf{else} \end{array}$$

is another form of writing

$$|x| := \textbf{if}\ x < 0\ \textbf{then}\ -x\ \textbf{else}\ x$$

which introduces the absolute value of x.

Constrained Function Definitions

Restrict Arguments (needs set theory):

$$
\begin{array}{l}
f : A_0 \times \ldots \times A_{n-1} \to B \\
f(x_0, \ldots, x_{n-1}) := T
\end{array}
$$

New axiom:

$$
\begin{array}{l}
\forall x_0, \ldots, x_{n-1} : \\
\quad \langle x_0, \ldots, x_{n-1} \rangle \in A \;\Rightarrow \\
\qquad f(x_0, \ldots, x_{n-1}) = T.
\end{array}
$$

If arguments violate condition, function result is undefined.

Example

We define

$$\begin{aligned}
&\mathrm{mult} : \mathbb{R}_{\neq 0} \times \mathbb{R}_{\neq 0} \to \mathbb{R}_{\neq 0} \\
&\mathrm{mult}(x, y) := x * y
\end{aligned}$$

For all real numbers $x \neq 0, y \neq 0$, we have

$$\mathrm{mult}(x, y)/y = x.$$

However, we do not know the value of

$$\mathrm{mult}(0, y)$$

therefore we must not conclude $\mathrm{mult}(0, y)/y = 0$.

Implicit Function Definitions

Result defined by a formula F:

$$\text{Let } f(x_0, \ldots, x_{n-1}) \text{ be a } y \text{ such that } F.$$

Translation:

$$f(x_0, \ldots, x_{n-1}) := \textbf{such } y : F$$

Introduction of a new kind of terms.

Such Quantifier

Definition: term (**such** x: F)

- Takes variable x and formula F.
- Constructs term where x is bound.
- Value is some x such that F holds (if such an x exists).

New Axiom:

$$(\exists x : F) \Rightarrow (\forall x : x = (\textbf{such } x : F) \Rightarrow F),$$

If an object with property F exists, then the value of the “such term” satisfies F as well.

Example

$$x/y := \textbf{such } q : q * y = x.$$

- $12/4 = 3$ because
  - Existence: $\exists q : q * 4 = 12$ (because $3 * 4 = 12$);
  - Unicity: $\forall q : q * 4 = 12 \Rightarrow q = 3$
- We must not assume that $(1/0) * 0 = 1$, because we cannot prove

$$\exists q : q * 0 = 1.$$

Take care when reasoning about implicitly defined functions.

Example

$$\text{Let } f(x) \text{ be a } y \text{ such that } y|x.$$

i.e.,

$$f(x) := \textbf{such } y : y|x$$

- We know $\exists x : x|12$ (because $6|12$).
- We know $f(12)|12$.
- We know $f(12) \in \{1, 2, 3, 4, 6, 12\}$.
- We do not know $f(12) = 6$.

Value is not necessarily unique.

Unique Implicit Function Definitions

Result defined by a formula:

$$\text{Let } f(x_0, \ldots, x_{n-1}) \text{ be } \underline{\text{the}}\ y \text{ such that } F.$$

Translation:

$$f(x_0, \ldots, x_{n-1}) := \textbf{such } y : F$$

Plus claim (to be proved):

$$(\forall x_0, \ldots, x_{n-1}, y : F \Rightarrow y = f(x_0, \ldots, x_{n-1})).$$

Unicity has to be proved separately!

Example

- It is wrong to write

  Let $f(x)$ be the y such that $y|x$.

  because from $6|12$ we cannot conclude $f(12) = 6$.

- We may define

  Let x/y be the q such that $q * y = x$.

  because we can prove unicity:

$$(\forall x, y, q : q * y = x \Rightarrow (\forall q' : q' * y = x \Rightarrow q = q'))$$

  Since $3 * 4 = 12$, we may thus conclude $12/4 = 3$.

Explicit Definitions

$$p(x_0, \ldots, x_{n-1}) :\Leftrightarrow F$$
$$f(x_0, \ldots, x_{n-1}) := T$$

- Definiendum (p, f) must not occur in definiens (F, T).
- Restriction necessary to avoid contradictions:
  - Invalid: $p(x) :\Leftrightarrow \neg p(x)$.
  - Invalid: $f(x) := 1 + f(x)$.

Restriction may be lifted in certain cases.

Recursive Definitions

$$
\begin{aligned}
p(x_0,\ldots,x_{n-1}) &:\Leftrightarrow \textbf{if } C \textbf{ then } F_0 \textbf{ else } F_1 \\
f(x_0,\ldots,x_{n-1}) &:= \textbf{if } C \textbf{ then } T_0 \textbf{ else } T_1
\end{aligned}
$$

- p/f may appear in one branch of the conditional.
- Restriction: we can construct a termination function t:
  - $\forall x_0,\ldots,x_{n-1} : t(x_0,\ldots,x_{n-1}) \in \mathbb{N}$.
  - For any occurence of $p(T_0,\ldots,x_{n-1})$ respectively $f(T_0,\ldots,x_{n-1})$ in the definiens, we have

$$t(T_0,\ldots,T_{n-1}) < t(x_0,\ldots,x_{n-1})$$

Special case of a more general rule.

Example

Recursive function definition

$$
\begin{aligned}
& * : \mathbb{N} \times \mathbb{N} \to \mathbb{N} \\
& x * y := \\
& \quad \textbf{if } y = 0 \\
& \qquad \textbf{then } 0 \\
& \qquad \textbf{else } x + x * (y - 1)
\end{aligned}
$$

with termination function $t(x, y) := y$.

Reduction sequence:

$$
\begin{aligned}
& \underline{2 * 3} = 2 + (\underline{2 * 2}) = 2 + (2 + \underline{2 * 1}) = \\
& \quad 2 + (2 + (2 + (\underline{2 * 0}))) = 2 + (2 + (2 + 0)) = 6.
\end{aligned}
$$

Termination after a finite number of unfoldings.

Example

Recursive predicate definition

$$
\begin{array}{l}
\text{iseven} \subseteq \mathbb{N} \\
\text{iseven}(x) :\Leftrightarrow \\
\quad \textbf{if } x = 0 \\
\qquad \textbf{then } \mathrm{T} \\
\qquad \textbf{else } \neg\text{iseven}(x-1)
\end{array}
$$

with termination function $t(x) := x$.

Reduction sequence:

$$
\begin{array}{l}
\text{iseven}(3) = \neg\text{iseven}(2) = \neg\neg\text{iseven}(1) = \neg\neg\neg\text{iseven}(0) = \\
\quad \neg\neg\neg\mathrm{T} = \neg\neg F = \neg T = \mathrm{F}.
\end{array}
$$

Termination after a finite number of unfoldings.

Example

Bogus recursive definition:

$$
\begin{aligned}
&* : \mathbb{N} \times \mathbb{N} \to \mathbb{N} \\
&x * y := \\
&\quad \textbf{if } y = 0 \\
&\quad\quad \textbf{then } 0 \\
&\quad\quad \textbf{else } x + x * (y + 1)
\end{aligned}
$$

Reduction sequence:

$$
\underline{2 * 3} = 2 + (\underline{2 * 4}) = 2 + (2 + \underline{2 * 5}) = \\
2 + (2 + (2 + (\underline{2 * 6}))) = \ldots
$$

No termination function can be given.

Logic Evaluator

Predicate and function definitions:

pred p(x_0, ..., x_{n-1}) <=> F;
fun f(x_0, ..., x_{n-1}) = T;
fun f(x) = such(x in S: F, x);
fun f(x_0, ..., x_{n-1}) recursive R = T;

See lecture notes for examples.

Logic Evaluator

```
pred divides(x, y) <=>
  exists(z in nat(1, y): =(*(x, z), y));
> predicate divides/2.
fun divisor(x) =
  such(y in nat(2, x): divides(y, x), y);
> function divisor/1.
term divisor(6);
> 2.
term divisor(49);
> 7.
fun div(x, y) = such(z in nat(1, x): =(*(y, z), x), z);
> function div/2.
term div(70, 14);
> 5.
term div(70, 15);
> ERROR: no such value.
fun divquot(n) = such(x in nat(2, -(n, 1)), y in nat(1, x):
  =(n, *(x, y)), tuple(x, y));
> function divquot/1.
term divquot(85);
> <17, 5>.
```

```
term divquot(17);
```

Summary

- Definitions.
  - Theories and their extension.
- Explicit predicate and function definitions.
  - Constrained definitions.
  - Conditional definitions.
- Implicit function definitions.
  - Unicity of definitions.
- Recursive predicate and function definitions.
  - Termination functions.