First Order Predicate Logic

Pragmatics

Wolfgang Schreiner and Wolfgang Windsteiger
Wolfgang.(Schreiner|Windsteiger)@risc.jku.at

Research Institute for Symbolic Computation (RISC)
Johannes Kepler University (JKU), Linz, Austria
http://www.risc.jku.at

Pragmatics

We will now investigate the practical use of logic in two contexts.

- Defining Models
  - Introducing new domains and operations.
  - Unique characterizations of their meaning.
- Specifying Problems
  - Describing expectations for computations.
  - Assumptions on the inputs and guarantees for the outputs.

Highly relevant for computer science and mathematics.

The Standard Models $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$

Each model consists of a domain (a set of values) and constants, functions, predicates on that domain.

- The Natural Numbers
  - $\mathbb{N}$: the set of all natural numbers $0, 1, 2, \ldots$.
  - $\mathbb{N}_n$: the first n natural numbers $0, 1, \ldots, n-1$.
  - $\mathbb{N}_{>0}$: the natural numbers $1, 2, \ldots$ without 0.
- The Integer Numbers
  - $\mathbb{Z}$: the set of all integers $\ldots, -2, -1, 0, 1, 2, \ldots$.
- The Real Numbers
  - $\mathbb{R}$: the set of all real numbers.
  - $\mathbb{R}_{\geq 0}$: the set of all non-negative real numbers.
  - $\mathbb{R}_{>0}$: the set of all positive real numbers.

Example

- $n \in \mathbb{N}_8$: n is a natural number in the range $0, \ldots, 7$.

We assume the usual arithmetic operations.

The Standard Model “Set”

- Domain $\mathcal{P}(T)$
  - The set of all sets whose elements are from set T.
- Membership predicate: $e \in S$
  - Read: “element e is in set S”
- Set builder quantifier: $\{t \mid x \in S \wedge \ldots \wedge F\}$
  - Read: “the set of all values of term t where the variables $x, \ldots$ run over all elements of sets $S, \ldots$ that satisfy formula F”
  - Term t, terms $S, \ldots$ (denoting sets), formula F.

Example

- $S \in \mathcal{P}(\mathbb{N}_8)$: S is a set whose elements are natural numbers in $0, \ldots, 7$.
- $S = \{2 \cdot x \mid x \in \mathbb{N} \wedge x > 0\}$: S is the set of all positive even numbers.

Sets model “unordered collections”.

The Standard Model "Product"

- Domain $T_1 \times \ldots \times T_n$
  - The set of all tuples with n components that are from sets $T_1, \ldots, T_n$, respectively.
- Tuple constructor $(c_1, \ldots, c_n)$
  - Read: "the tuple with components $c_1, \ldots, c_n$"
- Tuple selector $t.i$
  - Read: "component i of tuple t".
  - Tuple index $i = 1, \ldots, n$.

Example

- $t \in \mathbb{N}_2 \times \mathbb{Z}$: t is a tuple with two components; its first component $t.1$ is a bit (0 or 1) and its second component $t.2$ is an integer.

Tuples model "records" or "structures".

The Standard Model "Sequence"

- Sequence Domains
  - T^*: the set of all finite sequences of values from set T.
  - T^ω: the set of all infinite sequences of values from set T.
- Sequence length $length(s)$
  - Read: "the length of sequence s".
  - Only if $s \in T^*$, i.e., s is finite.
- Sequence selector $s(i)$
  - Read: "element i of sequence s".
  - $s \in T^*$: $i \in \mathbb{N}_{length(s)}$
  - $s \in T^\omega$: $i \in \mathbb{N}$

Example

- $s \in \mathbb{Z}^*$: s is a finite sequence of integers; if $length(s) = 4$, it has elements $s(0), s(1), s(2), s(3)$.

Finite sequences model "arrays".

Domain Definitions

From the standard domains, we may build new domains.

- A domain definition

$$T := \ldots$$

  defines a new domain T from previously introduced domains using domain constructors and/or set builders.

Example

$$\begin{aligned}
\mathit{Nat} &:= \mathbb{N}_{2^{31}} \\
\mathit{Int} &:= \{i \mid i \in \mathbb{Z} \wedge -2^{31} \leq i \wedge i < 2^{31}\} \\
\mathit{IntArray} &:= \mathit{Int}^* \\
\mathit{IntStream} &:= \mathit{Int}^\omega \\
\mathit{Primes} &:= \{x \mid x \in \mathbb{N} \wedge x \geq 2 \wedge (\forall y \in \mathbb{N} : 1 < y \wedge y < x \rightarrow \neg(y|x))\}
\end{aligned}$$

Explicit Function Definitions

A new function my be introduced by describing its value.

- An explicit function definition

$$f : T_1 \times \ldots \times T_n \to T$$
$$f(x_1, \ldots, x_n) := t$$

  consists of
  - a new n-ary function constant f,
  - a type signature $T_1 \times \ldots \times T_n \to T$ withs sets $T_1, \ldots, T_n, T$,
  - a list of variables $x_1, \ldots, x_n$ (the parameters), and
  - a term t (the body) whose free variables occur in $x_1, \ldots, x_n$;
  - case $n = 0$: the definition of a value constant $f : T, f := t$.
- We have to show for the newly introduced function f

$$\forall x_1 \in T_1, \ldots, x_n \in T_n : t \in T$$

  and then know

$$\forall x_1 \in T_1, \ldots, x_n \in T_n : f(x_1, \ldots, x_n) = t$$

The body of an explicit function definition may only refer to *previously* defined functions (no recursion).

Examples

- Definition: Let x and y be natural numbers. Then the *square sum* of x and y is the sum of the squares of x and y.

$$\begin{aligned} &\mathit{squaresum} : \mathbb{N} \times \mathbb{N} \to \mathbb{N} \\ &\mathit{squaresum}(x, y) := x^2 + y^2 \end{aligned}$$

- Definition: Let x and y be natural numbers. Then the *squared sum* of x and y is the square of z where z is the sum of x and y.

$$\begin{aligned} &\mathit{sumsquared} : \mathbb{N} \times \mathbb{N} \to \mathbb{N} \\ &\mathit{sumsquared}(x, y) := \textbf{let } z = x + y \textbf{ in } z^2 \end{aligned}$$

- Definition: Let n be a natural number. Then the *square sum set* of n is the set of the square sums of all numbers x and y from 1 to n.

$$\begin{aligned} &\mathit{squaresumset} : \mathbb{N} \to \mathcal{P}(\mathbb{N}) \\ &\mathit{squaresumset}(n) := \{\mathit{squaresum}(x, y) \mid x, y \in \mathbb{N} \wedge 1 \leq x \leq n \wedge 1 \leq y \leq n\} \end{aligned}$$

Explicit Predicate Definitions

A new predicate my be introduced by describing its truth value.

- An explicit predicate definition

$$p \subseteq T_1 \times \ldots \times T_n$$
$$p(x_1, \ldots, x_n) :\Leftrightarrow F$$

  consists of

  - a new n-ary predicate constant p,
  - a type signature $T_1 \times \ldots \times T_n$ with sets $T_1, \ldots, T_n$
  - a list of variables $x_1, \ldots, x_n$ (the parameters), and
  - a formula F (the body) whose free variables occur in $x_1, \ldots, x_n$.
  - case $n = 0$: definition of a truth value constant $p :\Leftrightarrow F$.

- We then know for the newly introduced predicate p:

$$\forall x_1 \in T_1, \ldots, x_n \in T_n : p(x_1, \ldots, x_n) \leftrightarrow F$$

The body of an explicit predicate definition may only refer to *previously* defined predicates (no recursion).

Examples

- Definition: Let x, y be natural numbers. Then x *divides* y (written as $x|y$) if $x \cdot z = y$ for some natural number z.

$$| \subseteq \mathbb{N} \times \mathbb{N}$$
$$x|y :\Leftrightarrow \exists z \in \mathbb{N} : x \cdot z = y$$

- Definition: Let x be a natural number. Then x *is prime* if x is at least two and the only divisors of x are one and x itself.

$$\mathit{isprime} \subseteq \mathbb{N}$$
$$\mathit{isprime}(x) :\Leftrightarrow x \geq 2 \wedge \forall y \in \mathbb{N} : y|x \rightarrow y = 1 \vee y = x$$

- Definition: Let p, n be a natural numbers. Then p is a *prime factor* of n, if p is prime and divides n.

$$\mathit{isprimefactor} \subseteq \mathbb{N} \times \mathbb{N}$$
$$\mathit{isprimefactor}(p, n) :\Leftrightarrow \mathit{isprime}(p) \wedge p|n$$

Implicit Function Definitions

A new function may be introduced by a condition for its value.

- An implicit function definition

$$f : T_1 \times \ldots \times T_n \to T$$
$$f(x_1, \ldots, x_n) := \textbf{such } y : F \text{ (or: } \textbf{the } y : F)$$

  consists of
  - a new n-ary function constant f,
  - a type signature $T_1 \times \ldots \times T_n \to T$ withs sets $T_1, \ldots, T_n, T$,
  - a list of variables $x_1, \ldots, x_n$ (the parameters),
  - a variable y (the result variable),
  - a formula F (the result condition) whose free variables occur in $x_1, \ldots, x_n, y$.
- We then know for the newly introduced function f

$$\forall x_1 \in T_1, \ldots, x_n \in T_n :$$
$$(\exists y \in T : F) \to (\exists y \in T : F \wedge y = f(x_1, \ldots, x_n))$$

  - If there is some value that satisfies the result condition, the function result is one such value (otherwise, it is undefined).
  - With **the** we claim that the value of f always exists and is unique.

The definition of a function by a formula (rather than a term).

Examples

- Definition: Let x be a real number. *A root* of x is a real number y such that the square of y is x (if such a y exists).

$$\begin{aligned} &aRoot : \mathbb{R} \to \mathbb{R} \\ &aRoot(x) := \textbf{such } y : y^2 = x \end{aligned}$$

- Definition: Let x be a non-negative real number. *The root* of x is that real number y such that the square of y is x and $y \geq 0$.

$$\begin{aligned} &theRoot : \mathbb{R}_{\geq 0} \to \mathbb{R} \\ &theRoot(x) := \textbf{the } y : y^2 = x \wedge y \geq 0 \end{aligned}$$

- Definition: Let $m, n \in \mathbb{N}$ with n positive. Then the (truncated) quotient $q \in \mathbb{N}$ of m and n is such that $m = n \cdot q + r$ for some $r \in \mathbb{N}$ with $r < n$.

$$\begin{aligned} &quotient : \mathbb{N} \times \mathbb{N}_{>0} \to \mathbb{N} \\ &quotient(m, n) := \textbf{the } q : \exists r \in \mathbb{N} : m = n \cdot q + r \wedge r < n \end{aligned}$$

- Definition: Let x, y be positive natural numbers. Then $gcd(x, y)$ denotes the greatest such number that divides both x and y.

$$\begin{aligned} &gcd : \mathbb{N}_{>0} \times \mathbb{N}_{>0} \to \mathbb{N}_{>0} \\ &gcd(x, y) := \textbf{the } z : z|x \wedge z|y \wedge \forall z' \in \mathbb{N}_{>0} : z'|x \wedge z'|y \to z' \leq z \end{aligned}$$

The result of an implicitly specified function is not necessarily uniquely defined (and may be also completely undefined).

Predicates versus Functions

A predicate gives rise to functions in two ways.

- A predicate:

$$\mathit{isprimefactor} \subseteq \mathbb{N} \times \mathbb{N}$$
$$\mathit{isprimefactor}(p, n) :\Leftrightarrow \mathit{isprime}(p) \wedge p|n$$

- An implicitly defined function:

$$\mathit{someprimefactor} : \mathbb{N} \to \mathbb{N}$$
$$\mathit{someprimefactor}(n) := \textbf{such } p : \mathit{isprime}(p) \wedge p|n$$

- An explicitly defined function whose result is a set:

$$\mathit{allprimefactors} : \mathbb{N} \to \mathcal{P}(\mathbb{N})$$
$$\mathit{allprimefactors}(n) := \{p \in \mathbb{N} \mid \mathit{isprime}(p) \wedge p|n\}$$

The preferred style of definition is a matter of taste and purpose.

Specifying Problems

An important role of logic in computer science is to specify problems.

- The specification of a (computational) problem

$$\textbf{Input: } x_1 \in T_1, \ldots, x_n \in T_n \quad \textbf{where } I$$
$$\textbf{Output: } y_1 \in U_1, \ldots, y_m \in U_m \ \textbf{where } O$$

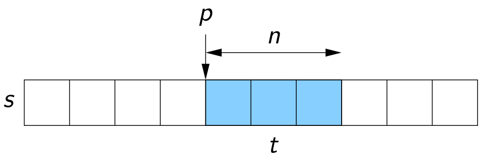
  consists of

  - a list of input variables $x_1, \ldots, x_n$ with types $T_1, \ldots, T_n$,
  - a formula I (the input condition or precondition) whose free variables occur in $x_1, \ldots, x_n$,
  - a list of output variables $y_1, \ldots, y_m$ with types $U_1, \ldots, U_m$, and
  - a formula O (the output condition or postcondition) whose free variables occur in $x_1, \ldots, x_n, y_1, \ldots, y_m$.

The specification is expressed with the help of functions and predicates that have been previously defined to describe the problem domain.

Example

Extract from a finite sequence s of natural numbers a subsequence of length n starting at position p.



Input: $s \in \mathbb{N}^*, n \in \mathbb{N}, p \in \mathbb{N}$ **where**

$$n + p \leq length(s)$$

Output: $t \in \mathbb{N}^*$ **where**

$$length(t) = n \;\wedge$$
$$\forall i \in \mathbb{N}_n : t(i) = s(i+p)$$

The resulting sequence must have appropriate length and content.

The Adequacy of Specifications

Given a specification

Input: x **where** $P(x)$ **Output:** y **where** $Q(x, y)$

we may ask the following questions:

- Is precondition satisfiable? $(\exists x : P(x))$

  Otherwise no input is allowed.

- Is precondition not trivial? $(\exists x : \neg P(x))$

  Otherwise every input is allowed, why then the precondition?

- Is postcondition always satisfiable? $(\forall x : P(x) \rightarrow \exists y : Q(x, y))$

  Otherwise no implementation is legal.

- Is postcondition not always trivial? $(\exists x, y : P(x) \wedge \neg Q(x, y))$

  Otherwise every implementation is legal.

- Is result unique? $(\forall x, y_1, y_2 : P(x) \wedge Q(x, y_1) \wedge Q(x, y_2) \rightarrow y_1 = y_2)$

  Whether this is required, depends on our expectations.

Example: The Problem of Integer Division

Input: $m \in \mathbb{N}, n \in \mathbb{N}$

Output: $q \in \mathbb{N}, r \in \mathbb{N}$ **where** $m = n \cdot q + r$

- The postcondition is always satisfiable but not trivial.
  - For $m = 13, n = 5$, e.g. $q = 2, r = 3$ is legal but $q = 2, r = 4$ is not.
- But the result is not unique.
  - For $m = 13, n = 5$, both $q = 2, r = 3$ and $q = 1, r = 8$ are legal.

Input: $m \in \mathbb{N}, n \in \mathbb{N}$

Output: $q \in \mathbb{N}, r \in \mathbb{N}$ **where** $m = n \cdot q + r \wedge r < n$

- Now the postcondition is not always satisfiable.
  - For $m = 13, n = 0$, no output is legal.

Input: $m \in \mathbb{N}, n \in \mathbb{N}$ **where** $n \neq 0$

Output: $q \in \mathbb{N}, r \in \mathbb{N}$ **where** $m = n \cdot q + r \wedge r < n$

- The precondition is not trival but satisfiable.
  - $m = 13, n = 0$ is not legal but $m = 13, n = 5$ is.
- The postcondition is always satisfiable and result is unique.
  - For $m = 13, n = 5$, only $q = 2, r = 3$ is legal.

Example: The Problem of Linear Search

Given a finite integer sequence a and an integer x, determine the smallest position p at which x occurs in a ($p = -1$, if x does not occur in a).

Example: $a = [2, 3, 5, 7, 5, 11], x = 5 \rightsquigarrow p = 2$

Input: $a \in \mathbb{Z}^*, x \in \mathbb{Z}$
Output: $p \in \mathbb{N} \cup \{-1\}$ **where**
  let $n = \mathit{length}(a)$ **in**
  if $\exists p \in \mathbb{N}_n : a(p) = x$
    then $p \in \mathbb{N}_n \wedge a(p) = x \wedge (\forall q \in \mathbb{N}_n : a(q) = x \rightarrow p \leq q)$
    else $p = -1$

All inputs are legal; the result always exists and is uniquely determined.

Example: The Problem of Binary Search

Given a finite integer sequence a that is sorted in ascending order and an integer x, determine some position p at which x occurs in a ($p = -1$, if x does not occur in a).

Example: $a = [2,3,5,5,5,7,11], x = 5 \rightsquigarrow p \in \{2,3,4\}$

Input: $a \in \mathbb{Z}^*, x \in \mathbb{Z}$ **where**

  let $n = \mathit{length}(a)$ **in**

  $\forall k \in \mathbb{N}_{n-1} : a(k) \leq a(k+1)$ // a is sorted

Output: $p \in \mathbb{N} \cup \{-1\}$ **where**

  let $n = \mathit{length}(a)$ **in**

  if $\exists p \in \mathbb{N}_n : a(p) = x$

   then $p \in \mathbb{N}_n \wedge a(p) = x$

   else $p = -1$

Not all inputs are legal; for every legal input, the result exists but is not uniquely determined.

Example: The Problem of Sorting

Given a finite integer sequence a, determine that permutation b of a that is sorted in ascending order.

Example: $a = [5,3,7,2,3] \rightsquigarrow b = [2,3,3,5,7]$

Input: $a \in \mathbb{Z}^*$

Output: $b \in \mathbb{N}^*$ **where**

$$
\begin{array}{l}
\textbf{let } n = length(a) \textbf{ in} \\
length(b) = n \land \\
(\forall k \in \mathbb{N}_{n-1} : b(k) \leq b(k+1)) \land \text{ // } b \text{ is sorted} \\
\exists p \in \mathbb{N}_n{}^* : \text{ // } b \text{ is a permutation of } a \\
\quad (\forall k1 \in \mathbb{N}_n, k2 \in \mathbb{N}_n : k1 \neq k2 \rightarrow p(k1) \neq p(k2)) \land \\
\quad (\forall k \in \mathbb{N}_n : a(k) = b(p(k)))
\end{array}
$$

All inputs are legal; the result always exists and is uniquely determined.

Implementing Problem Specifications

The ultimate goal of computer science is to implement specifications.

- The specifications demands the definition of a function $f : T_1 \times \ldots \times T_n \to U_1 \times \ldots \times U_m$ such that

$$\forall x_1 \in T_1, \ldots, x_n \in T_n : I \to \quad \textbf{let } (y_1, \ldots, y_m) = f(x_1, \ldots, x_n) \textbf{ in } O$$

  - For all arguments $x_1, \ldots, x_n$ that satisfy the input condition,
  - the result $(y_1, \ldots, y_m)$ of f satisfies the output condition.
- The specification itself already implicitly defines such a function:

$$f(x_1, \ldots, x_n) := \textbf{such } y_1, \ldots, y_m : I \to O$$

- However, the specification is actually implemented only by an explicitly defined function (computer program).

  The correctness of the implementation with respect to the specification has to be verified (e.g. by a formal proof).

A core goal of CS is to adequately specify problems, to implement the specifications, and to verify the correctness of the implementations.