Software Verification and Testing

Lecture Notes: Z I

Motivation

so far: we have seen that properties of software systems can be specified using first-order logic, set theory and the relational calculus

tasks:

- develop a specific notation for software specifications
- provide an environment for declaring and combining these specifications
- integrate them into a formal method to refine specifications to executable code

slogan: speficiations are (possibly) non-executable programs

The Z Notation

remark: as a formal language, Z needs a formal unambiguous notation

observation:

- most of the concepts have already be defined using "standard" mathematical notation
- the Z notation is somewhat different

task: we will now review the Z notation and introduce some additional concepts

The Z Notation

first-order logic: the following notation is used in Z

- $\neg$, $\wedge$, $\vee$ are used for negation, conjunction and disjunction
- $\Rightarrow$ and $\Leftrightarrow$ are used for implication and bi-implication (also called equivalence)
- $\exists\, x \bullet \phi$ denotes "there is an x such that ϕ"
- $\exists_1$ stands for "there is precisely one. . . "
  this allows for definite descriptions of entities

The Z Notation

sets: the following notation is used in Z

- sets can be defined by extension: $\{a, b, c\}$
- comprehension is written as $\{x \mid \phi\}$
- set comprehension can be used for pattern definition:

$$\{x \mid drivesCar(x) \bullet address(x)\}$$

  yields the set of addresses of car drivers
- the power set of set A is denoted by $\mathbb{P}\,A$.

The Z Notation

types: Z uses special sets called types for specifying entities

analogy: types in programming languages

intuition: each value x in a specification is associated with precisely one type which is the largest set s such that $x \in s$

here: we identify sets and types, but write $x : T$ when x has type T

type construction:

- Z has only $\mathbb{Z}$ as a basic built-in type
- types can be constructed from given types for instance by taking cartesian products and powersets
- further constructs will appear later

The Z Notation

definitions: Z provides special notation for definitions

- we now consider declarations, abrreviations and axioms
- further kinds of definition will come later

declaration:

- type declaration: $[Type]$ introduces new basis type $Type$.
- variable declaration: $x : A$ declares variable x with type A

example:

- $[Guest, Room]$ declares basic types for a hotel booking system
- $x : Guest$ declares a guest x of the hotel

notation: pairs $x : A$ are called signatures

The Z Notation

abbreviations:

- notation: $symbol == term$
- condition: $symbol$ may not occur in $term$

example:

- $MarxBrothers == \{Chico, Harpo, Groucho, Gummo, Zeppo\}$
- $English == \{p : Person \mid drinksTea(p) \land putsMilk(p)\}$

remark: abbreviations can be parametrised ($\varnothing[S] == \{x : S \mid x \neq x\}$)

The Z Notation

axiomatic definitions: the definition of an object is constrained by conditions

notation: (separation of declaration and conditions/predicates)

$$\begin{array}{|l} \textit{declaration} \\ \hline \textit{predicate} \end{array}$$

example:

$$\begin{array}{|l} \mathbb{N} : \mathbb{P}\,\mathbb{Z} \\ \hline \forall\, z : \mathbb{Z} \bullet z \in \mathbb{N} \Leftrightarrow z \geq 0 \end{array}$$

variant: if a condition is true it can be omitted

The Z Notation

axiomatic definitions: can be parametric

notation: (from example)

$$
\begin{array}{|l}
\hline
[X] \\
\_ \subseteq \_ : \mathbb{P}\, X \leftrightarrow \mathbb{P}\, X \\
\hline
\forall\, s, t : \mathbb{P}\, X \bullet s \subseteq t \Leftrightarrow \forall\, x : X \bullet x \in s \Rightarrow x \in t \\
\hline
\end{array}
$$

remark: $\subseteq$ is defined as an infix operator on sets of arbitrary type

The Z Notation

relations:

- many constructs defined as previously
- maplet notation: $x \mapsto y$ as alternative to (x, y)
- $R(\!| A |\!)$ denotes relational image of A under R, i.e.,

$$R(\!| A |\!) = \mathrm{dom}(R \rhd A)$$

- converse of R is denoted by $R^{\sim}$ or R^{-1}
- identity on set X is denoted by $\mathrm{id}\, X$
- relational composition is denoted $R \mathbin{;} S$

The Z Notation

functions: Z uses the following notation

- $\fatsemi$ and $\circ$ for forward and backward composition, i.e.,

$$(f \fatsemi g)(x) = g(f(x)) \qquad \text{(contravariant)}$$

$$(f \circ g)(x) = f(g(x)) \qquad \text{(covariant)}$$

- $\pfun$ and $\rightarrow$ for partial and total functions
- $\pinj$ and $\rightarrowtail$ for partial and total injections
- $\psurj$ and $\twoheadrightarrow$ for partial and total surjections
- $\bij$ for bijections (injective and surjective functions)

The Z Notation

conventions:

- one usually writes $f\ x$ instead of $f(x)$
- infix notation is used, e.g., for arithmetic functions

$$3+4 \qquad 5*8 \qquad 4/2 \qquad 7-5$$

The Z Notation

λ-**notation:** definition of anonymous functions

- in mathematics $\lambda\, x.3 + x$ stands for function $x \mapsto 3 + x$ which for every x yields $3 + x$
- example
  - $double$ without λ

$$\begin{array}{|l}
double : \mathbb{N} \to \mathbb{N} \\
\hline
\forall\, m, n \in \mathbb{N} \bullet m \mapsto n \in \text{double} \Leftrightarrow m + m = n
\end{array}$$

  - $double$ with λ

$$\begin{array}{|l}
double : \mathbb{N} \to \mathbb{N} \\
\hline
double = \lambda\, n : \mathbb{N} \bullet n + n
\end{array}$$

The Z Notation

general syntax of λ-notation

$$\lambda\ declaration \mid constraint \bullet result$$

example:

$$pair : ((\mathbb{N} \pfun \mathbb{N}) \times (\mathbb{N} \pfun \mathbb{N})) \fun (\mathbb{N} \pfun (\mathbb{N} \times \mathbb{N}))$$

$$pair = \lambda f, g : \mathbb{N} \pfun \mathbb{N} \bullet (\lambda n : \mathbb{N} \mid n \in \operatorname{dom} f \cap \operatorname{dom} g \bullet (f\ \ n, g\ \ n))$$

The Z Notation

example:

$$
\begin{aligned}
& pair\ (\lambda\, n : \mathbb{N} \bullet 2 * n, \lambda\, n : \mathbb{N} \bullet 3 * n)\ 4 \\
& \quad = \lambda\, m : \mathbb{N} \bullet ((\lambda\, n : \mathbb{N} \bullet 2 * n)\ m, (\lambda\, n : \mathbb{N} \bullet 3 * n)\ m)\ 4 \\
& \quad = \lambda\, m : \mathbb{N} \bullet (2 * m, 3 * m)\ 4 \\
& \quad = (2 * 4, 3 * 4) \\
& \quad = (8, 12)
\end{aligned}
$$

remark: this uses $(\lambda\, x \bullet f\ x)\ a = f\ a$ or $(\lambda\, x \bullet f)\ a = f[a/x]$
x, f and a must have appropriate types

The Z Notation

function overriding: (applicable to relations)

$$
\begin{array}{|l}
\hline
[X, Y] \\
\_ \oplus \_ : (X \leftrightarrow Y) \times (X \leftrightarrow Y) \rightarrow (X \leftrightarrow Y) \\
\hline
\forall f, g : X \leftrightarrow Y \bullet f \oplus g = (\mathrm{dom}\ g \lhd f) \cup g \\
\hline
\end{array}
$$

remark:

- outside the domain of g we keep f
- inside the domain of g we replace f by g

question: is the overriding of two functions a function?

The Z Notation

example: tracking persons

- types

$$Persons == \{ferdinand, leopold, maximilian\}$$
$$Locations == \{bed, office, beergarden\}$$

- functions

$$\begin{array}{|l} where\_is_0 : Persons \rightarrow Locations \\ \hline where\_is_0 = \{ferdinand \mapsto office, leopold \mapsto beergarden, \\ \qquad\qquad\quad maximilian \mapsto beergarden\} \end{array}$$

The Z Notation

example: tracking persons

- functions (continue:)

$$update_0 : Persons \pfun Locations$$

$$update_0 = \{ferdinand \mapsto beergarden\}$$

$$update_1 : Persons \pfun Locations$$

$$update_1 = \{maximilian \mapsto bed, leopold \mapsto bed\}$$

The Z Notation

example: tracking persons

- functions (continued):

$$\begin{array}{|l} where\_is_1 : Persons \rightarrow Locations \\ \hline where\_is_1 = where\_is_0 \oplus update_0 \end{array}$$

$$\begin{array}{|l} where\_is_2 : Persons \rightarrow Locations \\ \hline where\_is_2 = where\_is_1 \oplus update_1 \end{array}$$

The Z Notation

example: tracking persons

$$where\_is_0 \;\; leopold = beergarden$$
$$where\_is_1 \;\; leopold = beergarden$$
$$where\_is_2 \;\; leopold = bed$$

$$where\_is_1 = \{ferdinand \mapsto beergarden, leopold \mapsto beergarden, maximilian \mapsto beergarden\}$$
$$where\_is_2 = \{ferdinand \mapsto beergarden, leopold \mapsto bed, maximilian \mapsto bed\}$$

remark: this can be done more elegantly. . .

The Z Notation

theorem: if $\mathrm{dom}\, f \cap \mathrm{dom}\, g = \varnothing$, then $f \oplus g = f \cup g$

proof: if $\mathrm{dom}\, f \cap \mathrm{dom}\, g = \varnothing$, then $\mathrm{dom}\, f = \overline{\mathrm{dom}\, g} \cap \mathrm{dom}\, f$

$$\begin{aligned} f \oplus g &= (\mathrm{dom}\, g \mathbin{⩤} f) \cup g \\ &= (\mathrm{dom}\, g \mathbin{⩤} (\mathrm{dom}\, f \lhd f)) \cup g \\ &= (\overline{\mathrm{dom}\, g} \lhd (\mathrm{dom}\, f \lhd f)) \cup g \\ &= ((\overline{\mathrm{dom}\, g} \cap \mathrm{dom}\, f) \lhd f) \cup g \\ &= (\mathrm{dom}\, f \lhd f) \cup g \\ &= f \cup g \end{aligned}$$

you can visualise this using Venn diagrams. . .

Finite Sets

observation: finite sets are in bijective correspondence with subsets of natural numbers

intuition: when a set is finite, we can assign a unique natural number to each element

number range:

$$\begin{array}{|l} \_ \mathrel{..} \_ : \mathbb{N} \times \mathbb{N} \to \mathbb{P}\,\mathbb{N} \\ \hline \forall\, m, n \in \mathbb{N} \bullet m \mathrel{..} n = \{ i : \mathbb{N} \mid m \leq i \leq n \} \end{array}$$

finite sets: $\mathbb{F}\,X == \{ s : \mathbb{P}\,X \mid \exists\, n : \mathbb{N} \bullet \exists\, f : 1 \mathrel{..} n \rightarrowtail\!\!\!\!\twoheadrightarrow s \bullet \mathrm{true} \}$

Finite Sets

cardinality: the cardinality of a finite set is just its size. . .

$$
\begin{array}{|l}
\hline
[X] \\
\# : \mathbb{F}\, X \rightarrow \mathbb{N} \\
\hline
\forall\, s : \mathbb{F}\, X;\; n : \mathbb{N} \bullet n = \# s \Leftrightarrow \exists\, f : (1 \mathinner{..} n) \rightarrowtail\!\!\!\!\rightarrow s \bullet \mathrm{true} \\
\hline
\end{array}
$$

set of all finite functions: $A \mathrel{+\!\!\!\!\!+\!\!\!\!\rightarrow} B == \{ f : A \mathrel{+\!\!\!\!\rightarrow} B \mid \mathrm{dom}\, f \in \mathbb{F}\, A \}$

set of all finite injections: $A \mathrel{\rightarrowtail\!\!\!\!\!+\!\!\!\!+} B == A \mathrel{+\!\!\!\!\!+\!\!\!\!\rightarrow} B \cap A \mathrel{\rightarrowtail\!\!\!\!\!+} B$

remark: $A \mathrel{+\!\!\!\!\!+\!\!\!\!\rightarrow} B$ $(A \mathrel{\rightarrowtail\!\!\!\!\!+\!\!\!\!+} B)$ is the set of all finite (repetition-free) collections of elements of B, indexed by elements from A

Finite Sets

properties of cardinality: let s and t be finite sets

$$\#\varnothing = 0$$

$$\#s \leq \#(\{a\} \cup s) \leq 1 + \#s$$

$$max(\#s, \#t) \leq \#(s \cup t) \leq \#s + \#t$$

$$\varnothing \leq \#(s \cap t) \leq max(\#s, \#t)$$

when are these bounds sharp?