1.1 Predicates and Quantifiers in Z

- **Predicates**
  - 0-ary predicates can be regarded as propositions (sentences) because they are simply statements of facts independent of any individual variables.
  - Unary predicates are simply properties of objects (individuals).
  - Binary predicates are relations between pairs of objects (individuals), and in general n-ary predicates express relations among n-tuple of objects (individuals).
- **Quantifiers**
  - Universal quantifier, Existential quantifier.
  - Unique quantified, Counting quantifier.

1.2 Predicates and Quantifiers in Z

- **Universal quantifier**
  - The universal quantifier is written as ‘ $\forall$ ’ and its pronounced ‘for all’ (it looks like an upside-down ‘A’ in for All). It is used in the form:

    $\forall$ declaration | constraint • predicate

    which state that for the variable given in the declaration, restricted to certain values (constraint, ‘ | constraint ’ may be omitted), the predicate holds for all values.

- **Existential quantifier**
  - The existential quantifier is written as ‘ $\exists$ ’ and is pronounced ‘’ (it looks like a backwards ‘E’ in there Exists). It is used in the form:

    $\exists$ declaration | constraint • predicate

    which state that for the variable given in the declaration, restricted to certain values (constraint, ‘ | constraint ’ may be omitted), the predicate holds for some value(s).

1.3 Predicates and Quantifiers in Z

- **Unique quantifier**
  - The unique quantifier is written as ‘ $\exists_1$ ’ and is similar to the existential quantifier except that is states that there exists only one value for which the predicate is true.
- **Counting quantifier**
  - The counting quantifier counts for how many values of the variable the predicate holds.
  - In Z this is not needed; we use a set comprehension to construct the set of values for which the predicate holds, and the finds the size of the set.

1.4 Predicates and Quantifiers in Z

- **Set comprehension**
  - It constructs (defines) a set by giving a condition (predicate) which must hold for all members of the set.
  - The general form of set comprehension is

    {declaration | constraint • expression}

    where the declaration is for a typical element and it gives the element's type; the constraint (' | constraint' may be the omitted) restricts the possible values of the typical element and it is a logical expression which must be true for that value of the typical element to be included; the expression is an expression indicating the value to be included in the set.
- **Examples**
  - {x: *Z* | Even(x) • x * x } the set of the squares of the even integers
  - {x: *Z* • x * x } the set of the squares of the integers

2.1 Relations in Z

Declaring a relation

- A relation relates elements of a set called the source or from-set to elements of a set called the target or to-set.
- Relation R relates typed set X to typed set Y:

  $$R: X \leftrightarrow Y \; (X \leftrightarrow Y == \mathbb{P}(X \times Y))$$

- Ex: [COUNTRY] the set of all countries

  [LANGUAGE] the set of all languages

  speaks: COUNTRY ↔ LANGUAGE

2.2 Relations in Z

- **Maplets**
  - A maplet $x \rightarrow y == (x, y)$ pronounced 'x is related to y' or 'x map to y'.
  - $x\ R\ y == x \rightarrow y \in R == (x, y) \in R$
  - Ex: GB $\rightarrow$ English $\in$ speaks == (GB, English) $\in$ speaks

2.3 Relations in Z

- **Domain and Range of a relation**
  - Relation R relates types set X to typed set Y:

    $$R: X \leftrightarrow Y \ (X \leftrightarrow Y == \mathcal{P}(X \times Y))$$

  - dom R == $\{a \mid (\exists b)((a, b) \in R)\}$, dom R $\subseteq$ X
  - ran R == $\{b \mid (\exists a)((a, b) \in R)\}$, ran R $\subseteq$ Y
- **Relational image**
  - For S $\subseteq$ dom R, $R(|S|) == \{b \mid (\exists a)((a, b) \in R)\}$,

    $R(|S|) \subseteq$ ran R
  - R (|S|) is pronounced ‘the image of S under R’ or ‘the relational image of S in R’.

2.4 Relations in Z

- **Infix relations**
  - _R_: $X \leftrightarrow Y$
  - Ex: _speaks_: COUNTRY $\leftrightarrow$ LANGAUGE, GB speaks English
- **Inverse of a relation**
  - Relation R relates types set X to typed set Y:

$$R: X \leftrightarrow Y \ (X \leftrightarrow Y == \mathbb{P}(X \times Y))$$

  - The inverse relation of the relation R, written as $R^{\sim}$, relates typed set Y to typed set X such that if x R y then y $R^{\sim}$ x.

2.5 Example Using Relations

[COUNTRY] the set of all countries of the world

[DATE] the dates of a given year

Hols

holidays: COUNTRY $\leftrightarrow$ DATE

REPLY ::= yes | no

Enquire

Ξ Hols

c?: COUNTRY

d?: DATE

rep!: REPLY

$((c?, d?) \in holidays \wedge rep! = yes) \vee$

$((c?, d?) \notin holidays \wedge rep! = no)$

2.5 Example Using Relations

$$\begin{array}{|l}
\textbf{Decree} \\
\Delta \text{ Hols} \\
c?: \text{COUNTRY} \\
d?: \text{DATE} \\
\hline
holidays' = holidays \cup \{(c?, d?)\} \\
\hline
\end{array}$$

$$\begin{array}{|l}
\textbf{Abolish} \\
\Delta \text{ Hols} \\
c?: \text{COUNTRY} \\
d?: \text{DATE} \\
\hline
holidays' = holidays \setminus \{(c?, d?)\} \\
\hline
\end{array}$$

$$\begin{array}{|l}
\textbf{Dates} \\
\Xi \text{ Hols} \\
c?: \text{COUNTRY} \\
ds!: \mathbb{P}\,\text{DATE} \\
\hline
ds! = holidays\, (\!|\{c?\}|\!) \\
\hline
\end{array}$$

2.6 Relations in Z

- **Domain restriction**
  - The domain restriction operator (‘ $\lhd$ ’) restricts a relation R to that part where the domain of that relation R is constrained in a particular set S:

    $S \lhd R$

    pronounced ‘the relation R domain restricted to S’.
- **Range restriction**
  - The range restriction operator (‘ $\rhd$ ’) restricts a relation R to that part where the range of that relation R is contained in a particular set S:

    $R \rhd S$

    pronounced ‘the relation R range restricted to S’.

2.7 Relations in Z

Domain subtraction

- The domain subtraction operator (‘ <+ ’) restricts a relation R to that part where the domain of that relation R is not constrained in a particular set S:

  S <+ R

  pronounced ‘the relation R domain subtracted to S’.

Range subtraction

- The range subtraction operator (‘ +> ’) restricts a relation R to that part where the range of that relation R is not contained in a particular set S:

  R +> S

  pronounced ‘the relation R range subtracted to S’.

2.8 Example Using Relations

[COUNTRY] the set of all countries of the world

[DATE] the dates of a given year

Hols

holidays: COUNTRY $\leftrightarrow$ DATE

EU: $\mathbb{P}$COUNTRY

EU $\lhd$ holidays (mapping only EU countries to their holidays)

EU $⩤$ holidays (mapping non-EU countries to their holidays)

Summer: $\mathbb{P}$DATE

holidays $\rhd$ summer (countries to holidays in the summer)

holidays $⩥$ summer (countries to holidays not in the summer)

2.9 Relations in Z

- **Forward composition**
  - The relation formed by the relation R, then the relation Q, is called the forward composition of R with Q:

    $R: X \leftrightarrow Y,\ Q: Y \leftrightarrow Z,$

    $R;\ Q: X \leftrightarrow Z$
- **Backward composition**
  - The backward composition of Q with R is the same as the forward composition of R with Q:

    $Q \bullet R == R\ ;\ Q$
- **Repeated composition of homogeneous relation**

    $R: X \leftrightarrow X,\ R: X \leftrightarrow X,$

    $R\ ;\ R : X \leftrightarrow X$

2.10 Relations in Z

- **Transitive closure**
  - The transitive closure R^+, as used in:

    $x \ R^+ \ y$

    means that there is a repeated composition of R which relates x to y.

- **Identity relation**
  - The identity relation

    id X

    is the relation which maps all x's on to themselves:

    $\text{id } X == \{x : X \bullet x \rightarrow x\}$

2.11 Relations in Z

- **Reflexive Transitive closure**
  - The repeated composition

    $$= R^{+} \cup \text{ id } X$$

    includes the identity relation. The reflexive transitive closure is similar to the transitive closure except that it includes the identity relation.

    $$x \; R^{*} \; x$$

    this is always true, even is x R x is false.

2.12 Example: Using Relations to define Family Relationships

[PERSON] the set of all possible uniquely identified persons

father, mother: PERSON $\leftrightarrow$ PERSON

parent: PERSON $\leftrightarrow$ PERSON

parent = father $\cup$ mother

sibling: PERSON $\leftrightarrow$ PERSON

sibling = (parent ; parent $^{\sim}$) \ id PERSON

(Note: the relation parent composed with its own inverse is the set of persons with the same parents)

ancestor: PERSON $\leftrightarrow$ PERSON

ancestor = parent $^{+}$