

## Kurzanleitung zu PVS Teil 5

### Subtypen

Alle in PVS definierten Funktionen müssen total sein. Eine Möglichkeit auch partielle Funktionen (wie z. B. die Division) in PVS zu behandeln ist die Funktion durch eine Einschränkung des Definitionsbereichs mit einem Subtyp zu „totalisieren“.

Beispiel: Subtyp der reellen Zahlen ungleich 0.

```
nzreal : NONEMPTY_TYPE = {r: real | r /= 0} CONTAINING 1
/: [real, nzreal -> real]
```

Die Anwendung einer Funktion, deren Argumente auf einen Subtyp eingeschränkt wurden, führt bei der Typüberprüfung meist zu Typkorrektheitsbedingungen.

**Prädikate und Subtypen:** Für ein Prädikat P über einem Elementtyp T bezeichnet (P) den zugehörigen Subtyp von T all derjenigen Elemente, die P erfüllen:

```
P : pred[nat] = LAMBDA (n:nat): n >= 100000
BigNumbers : TYPE = (P) % -- Typ der natürlichen Zahlen ab 100000
```

### PVS Beweisregeln (Fortsetzung)

**Beweisregeln:** (TYPEPRED) und (SKOSIMP\* :PREDS? T)

(TYPEPRED) führt die Subtypeinschränkungen für die Ausdrücke in EXPRS als Antezedentformeln ein.

```
(TYPEPRED &REST EXPRS):
  Extract subtype constraints for EXPRS and add as antecedents.
  Note that subtype constraints are also automatically recorded by
  the decision procedures.
```

Dieser Befehl ist nützlich, um Typbedingungen von Skolemvariablen sichtbar zu machen:

```

|-----
{1}  FORALL (x: {y: nat | y > 2}): P(x)

Rule? (skosimp*)
|-----
{1}  P(x!1)

Rule? (typepred "x!1")
{-1}  x!1 > 2
|-----
[1]  P(x!1)
    
```

Um Typinformation von quantifizierten Variablen direkt bei der Skolemisierung einzuführen, kann bei (SKOSIMP\*) die Schlüsselwortoption :PREDS? T benutzt werden.

```

|-----
{1}  FORALL (x: {y: nat | y > 2}): P(x)

Rule? (skosimp* :preds? T)
{-1}  x!1 > 2
|-----
[1]  P(x!1)
    
```