Induktionsbeweise mit BOBJ

Listen–Spezifikation

1. Auf dem Handout-Server findet sich die Spezifikation einer Liste ganzer Zahlen im File `list.bob` mit den Operationen `cons`, `length`, `head` und `append`. Leere Listen sind durch die Konstante `nil` definiert.

   (a) Laden Sie die Listen-Spezifikation und studieren Sie sie bitte. Sehen Sie sich die Spezifikation der Operation `head` genau an. Wie geht sie mit leeren Listen um?

   (b) Ergänzen sie die Spezifikation um die Operation `tail`, die den Rest einer Liste (ohne das vorderste Element) bestimmt. Wie sieht die Signatur aus? Durch welche Gleichungen wird ihr Verhalten bestimmt. Wie geht man mit der leeren Liste um?

   (c) Wie würde sich die Spezifikation ändern, wenn man `cons` als Infix-Operator `_::_` definieren würde?

Beweis von Eigenschaften von Operationen

2. Eine wichtige Eigenschaft von Operationen ist die Assoziativität, denn sie ermöglicht eine nebenläufige Implementierung der jeweiligen Operation.

   Die in der Listen–Spezifikation definierte Operation `append` ist assoziativ, also

   $$\forall L_1, L_2, L_3 : \texttt{append}(L_1, \texttt{append}(L_2, L_3)) = \texttt{append}(\texttt{append}(L_1, L_2), L_3)$$

   .

   Glauben wir nur diese Eigenschaft oder können wir sie beweisen? Ein möglicher Beweis führt eine Induktion über die Länge der Liste L_1 durch.

   Beweisen Sie bitte mit Hilfe von BOBJ die Assoziativität von `append`:

   (a) Wie sieht die Formulierung der Behauptung in BOBJ aus? (Das Gleichheitsprädikat ist `==`). Wie werden L_1, L_2, L_3 allquantifiziert?

   (b) Beweisen Sie den Induktionsanfang (L_1 ist die leere Liste), in dem sie einen geeigneten Term reduzieren.

   (c) Nehmen Sie die Induktionsvoraussetzung als Gleichung in das BOBJ-System auf. Wie sieht sie aus?

   (d) Beweisen Sie, dass die Behauptung auch für eine um ein Element verlängerte Liste L_1 gilt. (Wiederum Reduzieren eines geeigneten Ausdrucks).

   (e) Warum lässt sich die ursprüngliche Behauptung nun auf triviale Weise reduzieren?