Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Peter Lammich

SS 2017
26. 5. 2017

Functional Data Structures

Exercise Sheet 5

- Import *Complex_Main* instead of *Main*!
- For this exercise sheet (and homework!), you are not allowed to use sledgehammer! Proofs using the *smt*, *metis*, *meson*, *or moura* methods are forbidden!

Exercise 5.1 Estimating power-of-two by factorial

Prove that, for all natural numbers $n > 3$, we have $2^n < n!$. We have already prepared the proof skeleton for you.

lemma *exp_fact_estimate*: *"n>3 ⟹ (2::nat)ˆn < fact n"*
proof (*induction n*)
  case *0* **then show** *?case* **by** *auto*
next
  case (*Suc n*)
  assume *IH*: *"3 < n ⟹ (2::nat) ˆ n < fact n"*
  assume *PREM*: *"3 < Suc n"*
  show *"(2::nat) ˆ Suc n < fact (Suc n)"*

Fill in a proof here. Hint: Start with a case distinction whether *n>3* or *n=3*.

qed

Warning! Make sure that your numerals have the right type, otherwise proofs will not work! To check the type of a numeral, hover the mouse over it with pressed CTRL (Mac: CMD) key. Example:

lemma *"2ˆn ≤ 2ˆSuc n"*
  apply *auto* **oops** — Leaves the subgoal *2 ˆ n ≤ 2 ∗ 2 ˆ n*

You will find out that the numeral *2* has type *'a*, for which you do not have any ordering laws. So you have to manually restrict the numeral's type to, e.g., *nat*.

lemma *"(2::nat)ˆn ≤ 2ˆSuc n"* **by** *simp* — Note: Type inference will infer *nat* for the unannotated numeral, too. Use CTRL+hover to double check!

Exercise 5.2 Sum Squared is Sum of Cubes

- Define a function sumto $f\ n = \sum_{i=0\ldots n} f(i)$.
- Show that $(\sum_{i=0\ldots n} i)^2 = \sum_{i=0\ldots n} i^3$.

fun *sumto* :: *"(nat ⇒ nat) ⇒ nat ⇒ nat"*

lemma *"sumto (λx. x) n ^ 2 = sumto (λx. x^3) n"*
proof (*induct n*)
  case *0* **show** *?case* **by** *simp*
next
  case (*Suc n*)
  assume *IH*: "(*sumto* (λ*x*. *x*) *n*)^2 = *sumto* (λ*x*. *x* ^ *3*) *n*"
  note [*simp*] = *algebra_simps* — Extend the simpset only in this block
  show "(*sumto* (λ*x*. *x*) (*Suc n*))^2 = *sumto* (λ*x*. *x* ^ *3*) (*Suc n*)"

Insert a proof here

qed

Exercise 5.3 Paths in Graphs

A graph is described by its adjacency matrix, i.e., *G* :: *'a ⇒ 'a ⇒ bool.*

Define a predicate *path G u p v* that is true if *p* is a path from *u* to *v*, i.e., *p* is a list of nodes, not including *u*, such that the nodes on the path are connected with edges. In other words, *path G u* $(p_1 \ldots p_n)$ *v*, iff *G u* p_1, *G* p_i p_{i+1}, and $p_n = v$. For the empty path (*n=0*), we have *u=v*.

fun *path* :: *"('a ⇒ 'a ⇒ bool) ⇒ 'a ⇒ 'a list ⇒ 'a ⇒ bool"*

Test cases

definition *"nat_graph x y ⟷ y=Suc x"*
value ‹*path nat_graph 2 [] 2*›
value ‹*path nat_graph 2 [3,4,5] 5*›
value ‹¬ *path nat_graph 3 [3,4,5] 6*›
value ‹¬ *path nat_graph 2 [3,4,5] 6*›

Show the following lemma, that decomposes paths. Register it as simp-lemma.

lemma *path_append*[*simp*]: *"path G u (p1@p2) v ⟷ (∃ w. path G u p1 w ∧ path G w p2 v)"*

Show that, for a non-distinct path from *u* to *v*, we find a longer non-distinct path from *u* to *v*. Note: This can be seen as a simple pumping-lemma, allowing to pump the length of the path.

Hint: Theorem *not_distinct_decomp.*

lemma *pump_nondistinct_path*:
  assumes *P*: *"path G u p v"*
  assumes *ND*: *"¬distinct p"*

shows "$\exists\, p'.$ *length* $p' >$ *length* $p \wedge \neg$*distinct* $p' \wedge$ *path* $G\ u\ p'\ v$"

Homework 5 Estimate for Number of Leafs

Submission until Friday, June 2, 11:59am. Recall: Use Isar, proofs using *metis*, *smt*, *meson*, *or moura* (as generated by sledgehammer) are forbidden!

Define a function to count the number of leafs in a binary tree:

fun *num_leafs* :: "*'a tree* $\Rightarrow$ *nat*"

Start by showing the following auxiliary lemma:

lemma *auxl*:
  assumes *IHl*: "*num_leafs* $l \leq 2$ ^ *height* l" **and** *IHr*: "*num_leafs* $r \leq 2$ ^ *height* r"
    and *lr*: "*height* $l \leq$ *height* r"
  shows "*num_leafs*(*Node* $l\ x\ r$) ≤ 2 ^ *height*(*Node* $l\ x\ r$)"

Also show the symmetric statement. Hint: Copy-paste-adjust!

lemma *auxr*:
  assumes *IHl*: "*num_leafs* $l \leq 2$ ^ *height* l" **and** *IHr*: "*num_leafs* $r \leq 2$ ^ *height* r"
    and *rl*: "$\neg$ *height* $l \leq$ *height* r"
  shows "*num_leafs*(*Node* $l\ x\ r$) ≤ 2 ^ *height*(*Node* $l\ x\ r$)"

Finally, show that we can estimate the number of leafs in a tree as follows:

lemma "*num_leafs* $t \leq 2$^*height* t"
proof (*induction* t)
  case *Leaf* **show** *?case* **by** *auto*
next
  case (*Node* $l\ x\ r$)
  assume *IHl*: "*num_leafs* $l \leq 2$ ^ *height* l"
  assume *IHr*: "*num_leafs* $r \leq 2$ ^ *height* r"
  show "*num_leafs* $\langle l,\ x,\ r\rangle \leq 2$ ^ *height* $\langle l,\ x,\ r\rangle$"

Fill in your proof here

qed

Homework 5.1 Simple Paths

Submission until Friday, June 2, 11:59am. This homework is worth 5 bonus points.

A simple path is a path without loops, or, in other words, a path where no node occurs twice. (Note that the first node of the path is not included, such that there may be a simple path from u to u.)

Show that for every path, there is a corresponding simple path:

lemma *exists_simple_path*:
  assumes *"path G u p v"*
  shows "$\exists\, p'$. *path G u p' v* $\wedge$ *distinct p'*"
  using *assms*
proof (*induction p rule*: *length_induct*)
  case (*1 p*)
  assume *IH*: "$\forall\, pp$. *length pp* $<$ *length p* $\longrightarrow$ *path G u pp v*
    $\longrightarrow$ ($\exists\, p'$. *path G u p' v* $\wedge$ *distinct p'*)"
  assume *PREM*: *"path G u p v"*
  show "$\exists\, p'$. *path G u p' v* $\wedge$ *distinct p'*"

Fill in your proof here.

qed