Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Mohammad Abdulaziz, Ph.D.

SS 2020
24. 7. 2020

Functional Data Structures

Exercise Sheet 13

Presentation of Mini-Projects:

You are invited, on a voluntary basis, to give a short presentation (5-10 minutes) of your mini-projects in the tutorial on July 24.

If you are interested, please send me a recording via email until Wednesday. I will prepare a video containing these recordings and put it on the youtube channel.

Exam Proof Guidelines: We expect valid Isabelle proof scripts to be submitted as a solution to the questions of this exam. Major proof steps, especially inductions, need to be stated explicitly. The use of "sorry" may lead to the deduction of points but is preferable to spending a lot of time on individual proof steps.

Exercise 13.1 Amortized Complexity

A "stack with multipop" is a list with the following two interface functions:

fun *push* :: *"'a ⇒ 'a list ⇒ 'a list"* **where**
"push x xs = x # xs"

fun *pop* :: *"nat ⇒ 'a list ⇒ 'a list"* **where**
"pop n xs = drop n xs"

You may assume

definition *t_push* :: *"'a ⇒ 'a list ⇒ nat"* **where**
"t_push x xs = 1"

definition *t_pop* :: *"nat ⇒ 'a list ⇒ nat"* **where**
"t_pop n xs = min n (length xs)"

Use the potential method to show that the amortized complexity of *push* and *pop* is constant.

Homework 13.1 Amortised Complexity of Binomial Heap Insertion

Submission until Friday, 24. 7. 2020, 10:00am.

Recall the implementation of binomial heaps that is the theory file

"src/HOL/Data_Structures/Binomial_Heap.thy".

Use the potential method to show that the operation *insert* needs constant amortised running time. In particular, you will be required to find the constant and the potential function and prove the following theorem. Note: the running time of insert is modelled by the function *t_insert* in the theory file where binomial heaps are defined.

definition Φ **where**
abbreviation *c* **where**
lemma *amortised_insert*:
  assumes *"invar_bheap ts"*
  shows *"t_insert x ts* $+ \Phi$ *(insert x ts)* $- \Phi$ *ts* $=$ *c"*