Adv. Data Structures: Functional queues

Anders Ingemann (20052979)
Peter E. Hvidgaard (20062546)

December 16, 2011

Project

In this project we perform an experimental study of functional data structures. The implementations are done in Haskell, which is a lazy language. Due to this we have to make sure that our results are actually used, so the evaluations are not postponed.

We implement a queue using

1. A standard Haskell list.
2. A pair of lists, with amotized $O(1)$ gurantee if the same queue will never be argument to repeated queue operations.
3. A $O(1)$ list, with a worst-case gurantee of $O(1)$ per queue operation (if executed strictly).
4. A pair of lists, exploiting the properties of lazy evaluation of list concatenation to guranteed amortized $O(1)$ per queue operation.

We then design and perform experiements comparing the different implementations, where we cover the worst-case scenario for every queue

Remarks

IF ANY

Queue implementations

Teoretiske overvejelser og overfladisk analyse af deres running times

A standard Haskell list

A pair of lists

A $O(1)$ list

A pair of lists, exploiting laziness

0.1 Test cases forcing worst-case behaviour

Dette skal være noget om hvilket input det vil få de forskellige data struckturer til at kører deres worst-case

A standard Haskell list

A pair of lists

A $O(1)$ list

A pair of lists, exploiting laziness

0.2 Experiments with worst-case

A standard Haskell list

A pair of lists

A $O(1)$ list

A pair of lists, exploiting laziness

0.3　5

0.4 6