Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики

Факультет информационных технологий и программирования
Кафедра компьютерных технологий

Рыбак Андрей Викторович

Представление структур данных индуктивными семействами и доказательства их свойств

Научный руководитель: ассистент кафедры ТП Я. М. Малаховски
Санкт-Петербург

2014

Содержание

Введение **4**

Глава 1. Обзор предметной области **5**

1.1 Структуры данных 5
1.1.1 Функциональные структуры данных 5
1.2 Индуктивные семейства и зависимые типы 5
1.3 Agda 6
1.4 Выводы по главе 1 7

Глава 2. Описание реализованной структуры данных **9**

2.1 Постановка задачи 9
2.2 Структура данных «двоичная куча» 9
2.2.1 Заполнение дерева 10
2.3 Тип данных для двоичной кучи 10
2.4 Вставка элементов 10
2.4.1 Вставка элементов в заполненное дерево 10
2.4.2 Вставка элементов в незаполненное дерево 10
2.5 todo3 10
2.5.1 subsection 10
2.5.2 10
2.6 10
2.7 Выводы по главе 2 10

Заключение **11**

Список литературы **12**

Введение

Структуры данных используются в программировании повсеместно для упрощения хранения и обработки данных. Свойства структур данных происходят из инвариантов, которые эта структура данных соблюдает.

Практика показывает, что тривиальные структуры и их инварианты данных хорошо выражаются в форме индуктивных семейств. Мы хотим узнать насколько хорошо эта практика работает и для более сложных структур.

В данной работе рассматривается представление в форме индуктивных семейств структуры данных приоритетная очередь типа «двоичная куча».

Глава 1. Обзор предметной области

В <...> структуры данных позволяют хранить и обрабатывать множество однотипных и/или логически связанных данных в вычислительной технике. Задача (?) структур данных — облегчить написание программ для программистов и ускорить обработку данных.

1.1. Структуры данных

Структуры данных используются в программировании для абстрагирования обработки связанных и однородных данных.

Часто используемые структуры данных ... включаются в стандартные библиотеки языков программирования. Существует несколько различных .... Основные из них:

- foobar

1.1.1. Функциональные структуры данных

... В отличие от ...

1.2. Индуктивные семейства и зависимые типы

Индуктивные семейства — это типы данных, которые могут зависеть от типов и значений. Например, тип векторов индексированных длиной

```
module VecSample where
data ℕ : Set where
  zero : ℕ
  succ : ℕ → ℕ
data Vec A : ℕ → Set where
  nil  : Vec A zero
```

```
cons : ∀ {n} → A → Vec A n → Vec A (succ n)
```

Такое определение позволяет нам описать функцию head для такого списка, которая не может бросить исключение:

```
head : ∀ {A} {n} → Vec A (succ n) → A
```

У аргумента функции head тип Vec A (succ n), то есть вектор, в котором есть хотя бы один элемент. Это позволяет произвести сопоставление с образцом только по конструктору cons:

```
head (cons a as) = a
```

Одной из областей применения индуктивных семейств являются системы интерактивного доказательства теорем.

Индуктивные семейства позволяют формализовать математические структуры, кодируя утверждения о структурах в них самих, тем самым перенося сложность из доказательств в определения.

1.3. Agda

Agda [1] — чистый функциональный язык программирования с зависимыми типами.

В Agda есть поддержка модулей.

```
module AgdaDescription where
```

В коде на Agda широко используются символы Unicode. Пример — определение типа натуральных чисел.

```
data ℕ : Set where
  zero : ℕ
  succ : ℕ → ℕ
```

В Agda функции можно определять как mixfix операторы. Пример — сложение натуральных чисел:

```
_+_ : ℕ → ℕ → ℕ
zero + b = b
succ a + b = succ (a + b)
```

Символы подчеркивания обозначают места для аргументов.

Зависимые типы позволяют определять типы, зависящие (или индексированные) от значений других типов. Пример — список, индексированный длиной:

```
data Vec (A : Set) : ℕ → Set where
  nil  : Vec A zero
  cons : ∀ {n} → A → Vec A n → Vec A (succ n)
```

В фигурные скобки заключаются неявные аргументы.

Такое определение позволяет нам описать функцию head для такого списка, которая не может бросить исключение:

```
head : ∀ {A} {n} → Vec A (succ n) → A
```

У аргумента функции head тип Vec A (succ n), то есть вектор, в котором есть хотя бы один элемент. Это позволяет произвести сопоставление с образцом только по конструктору cons:

```
head (cons a as) = a
```

1.4. Выводы по главе 1

Рассмотрены некоторые существующие подходы к .... Описаны различные .... Кратко описана ... . Кратко описаны особенности языка про-

граммирования *Agda*.

Глава 2. Описание реализованной структуры данных

В данной главе описывается разработанная функциональная структура данных приоритетная очередь типа «двоичная куча».

2.1. Постановка задачи

Целью данной работы является разработка типов данных для представления структуры данных и инвариантов.

Требования к данной работе:

- Разработать типы данных для представления структуры данных
- Реализовать функции по работе со структурой данных
- Используя разработанные типы данных доказать выполнение инвариантов.

2.2. Структура данных «двоичная куча»

Определение 2.1. Двоичная куча или пирамида [2] — такое двоичное подвешенное дерево, для которого выполнены следующие три условия:

- Значение в любой вершине не больше (если куча для минимума), чем значения её потомков.
- На i-ом слое 2^i вершин, кроме последнего. Слои нумеруются с нуля.
- Последний слой заполнен слева направо (как показано на рисунке 2.1).

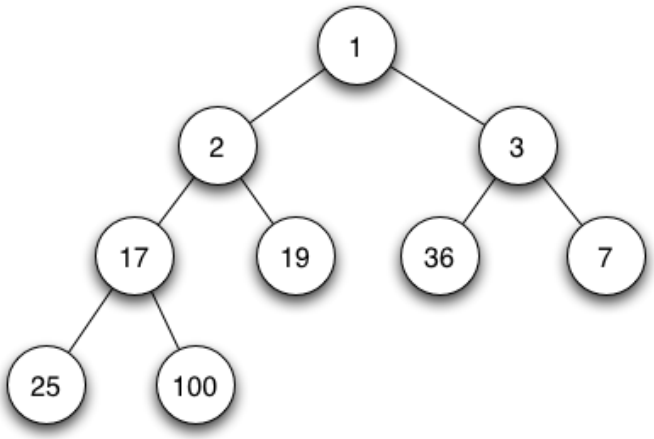

Рис. 2.1: Пример заполненной кучи для минимума

2.2.1. Заполнение дерева

2.3. Тип данных для двоичной кучи

2.4. Вставка элементов

2.4.1. Вставка элементов в заполненное дерево

2.4.2. Вставка элементов в незаполненное дерево

2.5. todo3

2.5.1. subsection

2.5.2.

2.6.

2.7. Выводы по главе 2

Разработаны типы данных для представления структуры данных двоичная куча.

Заключение

В данной работе реализована структура данных «двоичная куча». Разработаны типы данных для представления структуры данных и инвариантов индуктивными семействами. Реализованы функции для работы со структурой данных. Было доказано соблюдение инвариантов с использованием вспомогательных типов данных.

Таким образом, данная работа удовлетворяет поставленным требованиям.

Список литературы

1. Agda language. http://wiki.portal.chalmers.se/agda/pmwiki.php.
2. *Cormen T. H.*, *Leiserson C. E.*, *Rivest R. L.*, *Stein C.* Introduction to Algorithms, Second Edition. The MIT Press и McGraw-Hill Book Company, 2001. ISBN: 0-262-03293-7, 0-07-013151-1.