UMEÅ UNIVERSITET
Institutionen för Datavetenskap
Laborationsrapport

February 24, 2020

5DV086 - LAB 2

Programspråk 7.5 hp
VT 2020

Huffmankodning

Namn Emil Söderlind (id15esd@cs.umu.se)

Kursansvarig
Jan Erik Moström

Contents

1 Huffmankodning **1**

2 Användarhandledning **1**
2.1 Inlämningens innehåll 1
2.2 Programmets funktioner 1
2.2.1 Statistics 1
2.2.2 MakeTree 1
2.2.3 Encode 2
2.2.4 Decode 2
2.3 Kompilering och körning av programmet 2

3 Systembeskrivning **2**

4 Testkörningar **3**

1 Huffmankodning

Huffmankodning är ett sätt att utan dataförluster komprimera information. Genom att bygga upp ett binärt-träd där varje löv motsvarar ett tecken. Man låter varje tecken sedan motsvara koden som motsvarar vägen från roten av trädet till tecknets löv. Ena lövet motsvara 0 och andra 1. Kodningen av en hel sträng blir listan av 0:or och 1:or som motsvarar den vägen.

2 Användarhandledning

Nedan beskrivs hur programmet kan kompileras och köras samt exempelkörningar.

2.1 Inlämningens innehåll

Inlämningen består av Huffman.hs samt PDF:filen du just nu läser, ps_ou2_id15esd.pdf. Huffman.hs innehåller samtlig Haskell-kod för laborationen och PDF:en innehåller information om programmet och lösningen.

2.2 Programmets funktioner

Nedan beskrivs följande del-funktioner av programmet med exempelkörningar.

2.2.1 Statistics

Statistics returnerar antalet av varje tecken i en given sträng. Funktionen returnerar en lista av tuppler innehållandes varje tecken och antalet förekomster av tecknet i strängen.

Exempelkörning:

```
Huffman> :t statistics
statistics :: String -> [(Integer, Char)]

Huffman> statistics "Ostkakaa"

[(1,'O'),(3,'a'),(2,'k'),(1,'s'),(1,'t')]
```

2.2.2 MakeTree

MakeTree skapar ett huffman-träd utifrån resultatet från statistics. Se ovan 2.2.1.

```
Huffman> :t makeTree
maketree :: [(Integer, Char)] -> Htree

Huffman> makeTree "Ostkakaa"

Branch (Branch (Leaf 'O') (Leaf 'k'))
(Branch (Branch (Leaf 't') (Leaf 's')) (Leaf 'a'))
```

2.2.3 Encode

Encode tar en sträng och huffman-kodar den. Därefter returneras huffman-trädet samt strängens huffman-kodning.

```
Huffman> :t encode
encode :: String -> (Htree, [Integer])

Huffman> encode "Ostkakaa"

(Branch (Branch (Leaf 'O') (Leaf 'k'))
(Branch (Branch (Leaf 't') (Leaf 's')) (Leaf 'a')),
[0,0,1,0,1,1,0,0,0,1,1,1,0,1,1,1,1,1])
```

2.2.4 Decode

Decode tar ett huffman-träd och en huffman-kodning, avkodar kodningen och returnerar den avkodade strängen.

```
Huffman> :t decode
decode :: Htree -> [Integer] -> String

Huffman> tree = makeTree "Ostkakaa"
Huffman> decode tree [0,0,1,0,1,1,0,0,0,1,1,1,0,1,1,1,1,1]
"Ostkakaa"
```

2.3 Kompilering och körning av programmet

För att köra programmet ska vederbörande utföra nedan steg:

1. Se till att ghci är installerat på din maskin
2. Befinna dig i samma katalog som Huffman.hs med valfritt shell
3. Starta ghci genom att köra: "ghci"
4. Ladda Huffman.hs genom att köra: ":load Huffman.hs"
5. Därefter kan vederbörande köra de fyra olika funktionerna som exemplen ovan 2.2.

3 Systembeskrivning

Programmet består av följande fyra del-funktioner:

1. Programmet igenom den givna textsträngen och räknar antalet förekomster av varje tecken. Returnerar en lista av tupler med varje tecken och antalet förekomster av tecknet. Återfinns i funktionen **statistics**, se 2.2.1.
2. Utifrån resultatet från statistics skapas ett huffman-träd genom att först konstruera ett viktat träd och senare ta bort vikterna. Återfinns i funktionen **makeTree**, se 2.2.2.
3. Med hjälp av ovan två funktioner kan programmet i **encode**-funktionen utifrån en given sträng, bygga upp och returnera ett motsvarande huffman-träd och strängen huffmankodad. Se 2.2.3.
4. Den sista funktionen kan avkoda en huffmankodning givet kodningens huffmanträd. Programmet startar i trädets rot och "går nerför" trädet utifrån den givna kodningen och hittar i trädets löv de karaktärer som bygger upp meddelandet. Se **decode** under 2.2.4.

4 Testkörningar

För att säkerställa att programmet är korrekt har det tillhandahållna testet på labres[1] körts samt egna tester med påhittade strängar att koda och avkoda.

[1] webapps.cs.umu.se/labresults