CODIERUNGSTHEORIE - PRAKTIKA 1

Staubach Jan
Wombacher Sascha

29. November 2015

AGENDA

1. Generierung des Huffman-Baums

2. Zeichencodierung

3. Rekunstruktion des Huffman-Baums

4. Lesen des codierten Strings

GENERIERUNG DES HUFFMAN-BAUMS

Generierung des Huffman-Baums - Wahrscheinlichkeit eines Zeichen

1. für jedes Zeichen *z* der Eingabezeichenkette wird die Häufigkeit (Anzahl) errechnet
2. Anschließend wird diese Anzahl durch die Eingabelänge Normiert (Intervall zwischen (0, 1])

Generierung des Huffman-Baums - Baumgenerierung

1. Erstelle je ein Node pro Zeichen
2. Sortiere alle Nodes aufsteigend anhand ihrer Wahrscheinlichkeit
3. Verbinde die zwei Nodes mit den geriengsten Wahrscheinlichkeiten
4. Füge das entstehende Node in die Zeichenliste ein (Wahrscheinlichkeit = Summe der einzel Nodes)
5. Stelle die Sortierung wieder her (*insertion sort* - Ansatz)
6. Wiederhole Schritt 3-5 bis nur noch ein Node existiert (=> Tree-Root)

Generierung des Huffman-Baums - Speicherung

für die Weiterverarbeitung ist das Speichern des Huffman-Baums ein wichtiger Bestandteil

1. Ersten 32Bit beschreiben die Anzahl der Zeichen im Baum (Blätter)
2. Die Folgenden 8Bit beschreiben die Länge eines Zeichens (Bsp.: char = 1, int32 = 4)
3. Nehme ein Zeichen aus Huffman-Baum
4. Schreibe die Gesamtlänge des Zeichen + Codierung in die Folgenden 8Bit
5. Schreibe das Zeichen in die Nächsten Bits
6. Schreibe die Codierung des Zeichen in die folge Bits
7. Wiederhole Schritt 3-6 für jedes Zeichen im Huffman-Baum

Generierung des Huffman-Baums - Speicherung - Programmcode

```
1  BitWriter<> writer(output);
2  writer.add(leaveCount);
3
4  for (const _Leave<T>* ptr = this->m_FirstLeave; ptr; ptr
       = ptr->nextLeave){
5  writer.addByte(8 * (1 + (char)sizeof(T)) + (char)ptr->
       m_Coding.size());
6  const char* tmpPtr = reinterpret_cast<const char*>(&ptr
       ->data.first);
7  for (int i = 0; i < sizeof(T); ++i)
8  writer.addByte(tmpPtr[i]);
9  writer.addBits(ptr->m_Coding);
10 }
11 writer.flush();
12
```

ZEICHENCODIERUNG

ZEICHENCODIERUNG

1. Generiere pro Zeichen die jeweilige Codierung
   (von Root gesehen: leftNode = 1, rightNode = 0)
2. Lese ein Zeichen *z* des Input-Strings
3. Finde die Codierung für *z*
4. Füge die Codierung dem BitWriter hinzu
5. Wiederhole Schritte 2-4 für alle Zeichen des Strings
6. Flush für den BitWriter

REKUNSTRUKTION DES HUFFMAN-BAUMS

Rekunstruktion des Huffman-Baums

1. Öffne die erstellte Header-Datei, siehe Folie *6*
2. Lese *invertert* wie zuvor beschrieben

LESEN DES CODIERTEN STRINGS

LESEN DES CODIERTEN STRINGS

1. Setze Pointer auf Root
2. Lese ein Bit des Inputstreams (BitReader)
3. Verfolge Pointer anhand des Bits
   (1: Pointer = Pointer->left, 0: Pointer = Pointer->right)
4. Zeigt der Pointer auf ein Blatt?
   - Ja : Schreibe Zeichen, setze Pointer auf Root
5. Wiederhole Schritt 2 bis 4 für jedes Bit des Inputstreams

VIELEN DANK FÜR DIE AUFMERKSAMKEIT!