Albert-Ludwigs-Universität
Institut für Informatik
Prof. Dr. S. Albers
M. Schmidt, S. Schmelzer

Freiburg i. Br., den 30. Mai 2005

Software-Praktikum SS 05
Implementation von Datenkompressionsalgorithmen
Aufgabenblatt 4

Bearbeitung bis: 22.06.05

In der Arbeit „A Technique for High-Performance Data Compression“ von Terry A. Welch wird ein Kompressionsverfahren beschrieben, welches sich an die zu komprimierenden Daten anpasst. Dabei wird mit einem Initialwörterbuch gestartet, in dem alle 256 Byte enthalten sind. Sei c ein bisher gelesenes Wort von Bytes, welches im Wörterbuch enthalten ist. Wir lesen das nächste Byte b. Ist $c + b$ im Wörterbuch enthalten, so setzen wir $c = c + b$ und fahren mit dem Lesen der Datei fort. Andernfalls geben wir den Code für c aus, erzeugen einen neuen Codebucheintrag für $c + b$ und setzen $c = b$. Jedes Wort c wird dabei mit einer festen Anzahl von Bits codiert. Dadurch ergibt sich eine maximale Anzahl von möglichen Wörterbucheinträgen. Ist das Wörterbuch voll, so werden keine neuen Einträge erzeugt. Durch die feste Codewortlänge benötigt man keine Trennzeichen zwischen den einzelnen Codewörtern.

Erweitern Sie Ihr Programm derart, dass bei Auswahl des Eintrags `Komprimieren` des Menüs `LZW` ein Benutzerdialog erscheint, in dem die zu komprimierende Datei ausgewählt werden kann. Der Name der komprimierten Datei ergibt sich dabei aus dem Namen der ursprünglichen Datei und der Endung `.lzw`. Das Initialwörterbuch beinhaltet die 256 Byte in kanonischer Reihenfolge, d.h. bei einem 14-stelligen Code wird das Zeichen `00000000` durch die Bitfolge `00000000000000` codiert. Während der Kompression soll wieder eine Fortschrittsanzeige aktualisiert werden. Anschließend sind die entsprechenden Felder mit den Namen der Dateien, der Kompressionsrate, usw. zu belegen.

Die Dekompression soll analog erfolgen. Im Dialog zur Dateiauswahl werden nur Dateien mit der Endung `.lzw` angezeigt.

Die Kompression soll mit `12`, `14` und `16` Bit Codewortlänge möglich sein. Modifizieren Sie dazu das Fenster `Einstellungen` aus dem Menü `LZW`. Bei der Dekompression soll automatisch die verwendete Codewortlänge erkannt werden. Hierzu verwenden wir einen Header von 1 Byte, welcher am Anfang jeder komprimierten Datei steht. Dieser ist wie folgt aufgebaut:

- die ersten 3 Bit werden noch nicht verwendet, sie werden mit der Bitfolge 100 gefüllt und beim Lesen der Datei nicht betrachtet
- Bit 4 bis 8 enthalten die Codewortlänge

Beispiel: 10010000: Codewortlänge 16 Bit

Die verwendete Codewortlänge soll nach Kompression und Dekompression ebenfalls ausgegeben werden.

Ergänzen Sie Ihr UML-Klassendiagramm um die durch die LZW-Komprimierung hinzugekommenen Komponenten.

Hinweis: Auf unserer Praktikumsseite gibt es eine `tar`-Datei mit Beispielinstanzen. Diese umfasst folgende Dateien.

- `text1.txt`
- `text1_12bit.txt.lzw`
- `text2.txt`
- `text3_12bit.txt.lzw`
- `text3_14bit.txt.lzw`
- `text3_16bit.txt.lzw`

Die Datei `text1.txt` ist mit einer Codewortlänge von 12 Bit in `text1_12bit.txt.lzw` komprimiert. In `text2.txt` ist ein größerer Beispieltext in deutscher Sprache enthalten. Die letzten drei Dateien liefern bei Dekompression denselben Text. Die Kompression erfolgte dabei mit 12, 14 bzw. 16 Bit.