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

Freiburg i. Br., den 20. Juni 2005

Software-Praktikum SS 05
Implementation von Datenkompressionsalgorithmen
Aufgabenblatt 5

Bearbeitung bis: 06.07.05

In der Arbeit „A Technique for High-Performance Data Compression“ von Terry A. Welch werden weitere Varianten des Kompressionsverfahrens vorgestellt. Wir wollen drei Varianten implementieren.

a) **adaptive Codewortlänge**
Zu Beginn des Verfahrens sind nur die ersten 256 Einträge des Wörterbuchs belegt. Die übertragenen Codewörter können zu diesem Zeitpunkt mit 8 Bit codiert werden. Mit dem nächsten Eintrag ins Wörterbuch entsteht das Codewort 256, welches mindestens 9 Bit zur Codierung benötigt. Ein analoger Sprung erfolgt bei Aufnahme des Codewortes 512 ins Wörterbuch. Zu Beginn können also deutlich weniger als 12 (14, 16) Bit zur Codierung der Codewörter benutzt werden. Wir beginnen mit einer Codewortlänge von 9 Bit und erhöhen diese, bis wir eine vom Benutzer vorgegebene maximale Codewortlänge erreichen.

b) **leeres Anfangswörterbuch**
Es ist möglich mit einem leeren Wörterbuch zu starten. Wird ein neues Zeichen entdeckt, so muss der Code des neuen Zeichens und das Zeichen im Klartext übertragen werden. Sei c das bisher gelesene Wort von Bytes. Wir lesen das nächste Zeichen b. Ist b noch nicht im Codebuch enthalten, so geben wir den Code von c aus, erzeugen einen neuen Eintrag im Wörterbuch für b und geben anschließend den Code von b, eine binäre 1 und b im Klartext aus. Die binäre 1 dient zur Unterscheidung vom Sonderfall, in dem beim Dekomprimieren ein unbekanntes Codewort gelesen wird, welches aber aus den bisherigen Informationen konstruiert werden kann. Dieser Sonderfall muss beim Komprimieren auch erkannt und nach dem Code eine binäre 0 übertragen werden. Alle anderen Fälle für b werden analog zum Standardverfahren behandelt. Der letzte Eintrag im Wörterbuch wird nicht genutzt. Er dient zur Übertragung unbekannter Zeichen bei vollem Wörterbuch, d.h. es werden nur Codewörter von 0 bis $2^l - 2$ mit `l= Codewortlänge` vergeben. Sei b ein neues Zeichen. Bei einem vollen Wörterbuch könnte das Zeichen nicht übertragen werden. Durch die Beschränkung des Wörterbuchs steht uns aber noch ein Codewort mit l Bits zur Verfügung. Wir übertragen dieses Codewort gefolgt vom neuen Zeichen in der bereits beschriebenen Art. Auch beim Dekomprimieren darf für dieses Zeichen kein neuer Eintrag im Wörterbuch erzeugt werden.

c) **mehrere Dateien**
Bisher ist es nur möglich eine Datei zu komprimieren. Implementieren Sie das Verfahren nach Welch, so dass mehrere Dateien in eine gemeinsame Datei gepackt werden können.
Lösungsvorschlag: Nach dem Header ist in der BSTW-Codierung der Dateiname und die Originalgröße der ersten Datei codiert. Anschließend folgt die erste komprimierte Datei, gefolgt von Name und Größe der zweiten Datei usw.

Implementieren Sie a) und b). Die beiden Varianten müssen nicht gemeinsam anwendbar sein. Ändern Sie das Fenster Einstellungen entsprechend ab. Die ersten 3 Bit des Headers sind wie folgt aufgebaut

- Bit 1:
  1: Anfangswörterbuch
  0: leeres Anfangswörterbuch
- Bit 2:
  1: adaptive Codewortlänge
  0: feste Codewortlänge
- Bit 3:
  1: mehrere Dateien gepackt
  0: eine Datei gepackt

Variante c) ist eine Zusatzaufgabe. Hier können Sie gerne eigene Ideen verwirklichen, z.B. bei der Codierung der Dateinamen, der Reihenfolge, usw.

Ergänzen Sie Ihr UML-Klassendiagramm um die hinzugekommenen Komponenten.

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

- `text1.txt`
- `text1.txt.lzw`
- `text2.txt`
- `text2.txt.lzw`
- `book1`
- `pic`

Die Datei `text1.txt` wurde mit Codewortlänge 12-Bit und leerem Anfangswörterbuch komprimiert. Die Datei `text2.txt` wurde mit maximaler Codewortlänge 12-Bit und adaptiver Codewortlänge komprimiert. Die beiden Dateien `book1` und `pic` sind aus dem `Calgary Corpus` und sollen zum Testen aller Verfahren an größeren Beispielen genutzt werden.