Aufgaben zum Tut am 09.07.2007

Thomas Pajor

9. Juli 2007

Aufgabe 1.

Gegeben sei folgende Nachricht.

$$\alpha := rokokokostuem$$

(a) Kodieren Sie die Nachricht mittels LZW. Neue Wörterbucheinträge können ab dem Index 256 erzeugt werden.

(b) Dekodieren Sie die aus Aufgabe (a) erhaltene Nachricht mit dem LZW-Verfahren.

Lösung.

(a) Anwendung des Lempel-Ziv-Welch Algorithmus auf die Zeichenkette $\alpha = rokokokostuem$ führt zunächst zu der Tabelle 1. Die Ausgabe ist damit

$$\beta = rok\$257\$259stuem$$

und damit immerhin um drei Zeichen kürzer als die Eingabe. Man sieht hier auch schon den Nachteil der LZW-Kompression: Nämlich, dass wir ziemlich lange nichts gewinnen, bis sich manche Zeichenketten mehrfach wiederholen. Jedoch wie auch bei der Burrows-Wheeler-Transformation zeigt sich erst bei langen Zeichenketten mit häufig wiederholenden Wörtern (wie beispielsweise einem Text) die volle Wirkung des LZW-Verfahrens. Das Wörterbuch, das wir „im Speicher“ aufgebaut haben, braucht nicht mitgeschickt zu werden, da es bei der Dekompression automatisch rekonstruiert werden kann.

(b) Das Kodieren in LZW ist ziemlich straight-forward. Der interessantere Teil ist sicherlich die Dekodierung und das gleichzeitige Aufbauen des Wörterbuches. Da der Wörterbucheintrag beim Kodieren immer aus der Ausgabe a und dem *nächsten* Zeichen – sagen wir α – besteht, können wir bei der Dekompression die Wörterbucheinträge erst einen Schritt später erzeugen. Denn ist zum Beispiel in unserem Eingabestrom folgende Situation $\cdots a\$423\cdots$,

Eingabe	Ausgabe	Wörterbuch
r	*r*	256 = *ro*
o	*o*	257 = *ok*
k	*k*	258 = *ko*
o	\$257	259 = *oko*
k		
o	\$259	260 = *okos*
k		
o		
s	*s*	261 = *st*
t	*t*	262 = *tu*
u	*u*	263 = *ue*
e	*e*	264 = *em*
m	*m*	

Tabelle 1: LZW-Kodierung der Zeichenkette *rokokokostuem*

so wissen wir zum Zeitpunkt, zu dem wir das *a* lesen noch nicht wie das Zeichen, das *nach* dem *a* kommt. Schließlich hängt das ja von dem Inhalt in \$423 ab.

Tabelle 2 zeigt den Anfang der Dekodierung bis zum Auftreten des $K\Omega K$-Problems. Wir müssen den Wörterbucheintrag 259 aufsuchen, den es aber noch garnicht gibt. Das heißt beim Erstellen der LZW-Kodierung wurde der Eintrag 259 direkt im folgenden Schritt nach der Erstellung benutzt. Da wir jetzt einen Schritt später das Wörterbuch aufbauen würde der Eintrag in diesem Schritt erstellt werden. Das letzte Zeichen von \$259 entspricht ja dem Zeichen, was *nach* der Ausgabe des vorhergehenden Codewortes (in unserem Fall also *ok*) steht. Und da wir den Eintrag 259 bei der Kodierung in dem Schritt erstellt haben, als wir *ok* kodiert haben, und es danach gleich benutzt haben, muss dieses Zeichen dem ersten Zeichen von *ok* entsprechen. Der gesuchte Eintrag ist als 259 = *oko*. Nun können wir die Tabelle vervollständigen, und erhalten Tabelle 3.

Die Ausgabe ist also wieder

$$\alpha = rokokokostuem.$$

Eingabe	Ausgabe	Wörterbuch
r	*r*	
o	*o*	256 = *ro*
k	*k*	257 = *ok*
$257	*ok*	258 = *ko*
$259		
s		
t		
u		
e		
m		

Tabelle 2: LZW-Dekodierung der Zeichenkette β = *rok*$257$259*stuem* bis zum Auftreten des $K\Omega K$-Problems.

Eingabe	Ausgabe	Wörterbuch
r	*r*	
o	*o*	256 = *ro*
k	*k*	257 = *ok*
$257	*ok*	258 = *ko*
$259	*oko*	259 = *oko*
s	*s*	260 = *oko s*
t	*t*	261 = *st*
u	*u*	262 = *tu*
e	*e*	263 = *ue*
m	*m*	264 = *em*

Tabelle 3: Komplette Dekodierung der Zeichenkette.