Theoretische Informatik Serie 2

Benjamin Simmonds, Dario Näpfer, Fabian Bösiger

Aufgabe 4

Sowohl (a) als auch (b) haben eine geringere Kolmogorov-Komplexität als (c), da Programm (c) auf eine Binärdarstellung von n^2 im Programmcode angewiesen ist, (a) und (b) dagegen nur auf n angewiesen sind, und n^2 im Programm selbst ausgerechnet wird. Somit kommt (c) nicht als Lösung in Frage.

Programm (a) speichert n nur ein Mal, (b) dagegen drei Mal, (a) hat deshalb eine geringere Komplexität als b.

Die folgende Tabelle zeigt die Kolmogorov-Komplexitäten von (a), (b) und (c), wobei c die Anzahl Bits für die Darstellung des Programmcodes beschreibt.

Programm	Obere Schranke für Kolmogorov-Komplexität
(a)	$\log_2(n+1)+c$
(b)	$3 * \log_2(n+1)+c$
(c)	$\log_2(n^2+1)+\log_2(n+1)+c$

Somit liefert (a) die beste obere Schranke für die Kolmogorov-Komplexität.

Aufgabe 5

(a)

Wir betrachten folgendes Programm, wobei n der dynamische Teil des Programms bildet.

```
begin
    p = pow(2, n)
    for i = 1 to p do
        write("01")
    end
end
```

Zu beachten ist, dass n^2 während der Ausführung des Programms berechnet wird und somit nicht direkt im Programm gespeichert werden muss. Die Kolmogorov-Komplexität dieses Programms ist $K(n) \leq \log_2(n+1)+c$, wobei c die Grösse der Codierung des Rests des Programms darstellt und konstant ist.

(b)

Wir können ein Programm erstellen, welches für uns $2^{(i+1)^2}$ nur aus dem dynamischen Teil i berechnet.

```
begin
    p = pow(2, pow(i + 1, 2))
    write(p)
end
```

Somit hat y_i eine Kolmogorov-Komplexität von $K(y_i) \leq log_2(i+1) + c$, wobei c die Grösse der Codierung des Rests des Programms darstellt und konstant ist.

Wir wählen $y_i = 2^{(i+1)^2}$. Wenn wir y_i in der gegebenen Gleichung $K(y_i) \leq \frac{log_2 log_2 y_i}{2} + c$ einsetzen, erhalten wir:

$$K(y_i) \leq \frac{log_2 log_2(2^{(i+1)^2})}{2} + c$$

$$K(y_i) \leq \frac{log_2((i+1)^2)}{2} + c$$

$$K(y_i) \leq \frac{2log_2(i+1)}{2} + c$$

$$K(y_i) \leq log_2(i+1) + c$$

Diese Lösung stimmt mit der vorherigen Berechnung der Kolmogorov-Komplexität von y_i überein, und somit ist $y_i = 2^{(i+1)^2}$ unsere gesuchte Folge von natürlichen Zahlen.

Aufgbe 6

Wir stellen uns die Zahlen von 1 bis $f(n)$ im Binärformat mit der Länge $\log_2(f(n)+1)$ vor und unterteilen diese in vier Gruppen, basierend auf den beiden grössten Ziffern.

Wir stellen dabei fest, dass wir bei der Gruppe a die ersten beiden Bits der Binärdarstellung weglassen können, da diese beide 0 sind und somit keine Informationen liefern. Zudem können wir bei der Gruppe b das erste Bit weglassen. Die Schranken der Kolmogorov-Komplexität sehen dann folgendermassen aus:

Gruppennummer	Format	Untere Schranke der Kolmogorov-Komplexität
a	00??? ... ???	$\log_2(f(n)+1) - 2$
b	01??? ... ???	$\log_2(f(n)+1) - 1$
c	10??? ... ???	$\log_2(f(n)+1)$
d	11??? ... ???	$\log_2(f(n)+1)$

Wenn wir $f(n) = 2^{(2n+1)}$ betrachten, stellen wir fest, dass die Komplexität der Gruppen folgendermassen aussieht:

Gruppennummer	Format	Untere Schranke der Kolmogorov-Komplexität
a	00??? ... ???	$(2n+1)-2=2n-1$
b	01??? ... ???	$(2n+1)-1=2n$
c	10??? ... ???	$2n+1$
d	11??? ... ???	$2n+1$

Da drei Viertel der natürlichen Zahlen von 1 bis $f(n)$ in die Gruppen b, c und d fallen, und diese eine Kolmogorov-Komplexität von $K(n) \geq 2n$ haben, ist $f(n) = 2^{(2n+1)}$ unsere gesuchte Lösung.