Algorithmentheorie

„Amortisierung“

Stefan Edelkamp

Analyse von Algorithmen

- Best Case
- Worst Case
- Average Case
- Amortisierte Worst Case

Was sind die durchschnittlichen Kosten
einer schlechtest möglichen Folge
von Operationen?

Definition

- Unter den amortisierten worst-case Kosten einer Folge $o_1, ..., o_n$ von Operationen, die auf einer gegebenen Struktur ausgeführt werden, versteht man die durchschnittlichen Kosten pro Operation für eine schlechteste mögliche Wahl der Folge $o_1, ..., o_n$.
- Genauer als simple, oft zu pessimistische worst-case-Analyse
- Mögliche Vorgehensweisen bei der amortisierten worst-case-Analyse:
- Gesamtkosten der Operationsfolge berechnen und Durchschnitt bilden (Aggregat Methode)
- Bankkonto Paradigma: Bezahle für die „billigen“ Operationen (freiwillig) etwas mehr und bezahle die anderen von den Ersparnissen
- Potentialmethode: a_i = t_i + pot_i – pot_{i-1}

Amortisierung

Idee:

- Zahle für billige Operation etwas mehr
- Verwende Erspartes um für teure Operationen zu zahlen

Drei Methoden:

1. Aggregatmethode
2. Bankkonto – Methode
3. Potentialfunktion – Methode

1.Aggregatmethode: Dualzähler

Operation	Zählerstand	Kosten
0	00000	1
1	00001	2
2	00010	1
3	00011	3
4	00100	1
5	00101	2
6	00110	
7	00111	
8	01000	
9	01001	
10	01010	
11	01011	
12	01100	
13	01101	

2. Bankkonto – Methode

Idee:

Bezahle zwei KE für das Verwandeln einer 0 in eine 1

→ jede 1 hat eine KE auf dem Konto

Beobachtung:

In jedem Schritt wird genau eine 0 in eine 1 verwandelt

Bankkonto – Methode

Operation	Zählerstand
0	0 0 0 0 0
1	0 0 0 0 1
2	0 0 0 1 0
3	0 0 0 1 1
4	0 0 1 0 0
5	0 0 1 0 1
6	0 0 1 1 0
7	0 0 1 1 1
8	0 1 0 0 0
9	0 1 0 0 1
10	0 1 0 1 0

3. Potentialfunktion

Potentialfunktion ϕ

Datenstruktur $D \rightarrow \phi(D)$

t_l = wirkliche Kosten der l-ten Operation

ϕ_l = Potential nach Ausführung der l-ten Operation (= $\phi(D_l)$)

a_l = amortisierte Kosten der l-ten Operation

Definition:

$$a_l = t_l + \phi_l - \phi_{l-1}$$

Beispiel: Dualzähler

D_i = Stand der i-ten Operation

$\phi_i = \phi(D_i)$ = # von Einsen in D_i

i–te Operation	# von Einsen
D_{i-1}: .....0/1.....01.....1	B_{i-1}
⬇	
D_i : .....0/1.....10.....0	$B_i = B_{i-1} - b_i + 1$

t_i = **wirkliche Bitwechselkosten von Operation i**

$= b_i + 1$

Dualzähler

$$
\begin{aligned}
a_i &= (b_i + 1) + (B_{i-1} - b_i + 1) - B_{i-1} \\
&= 2 \\
&\Rightarrow \sum t_i \le 2n
\end{aligned}
$$

t_i = wirkliche Bitwechselkosten von Operation i

a_i = amortisierte Bitwechselkosten von Operation i