Script generated by TTT

Title: TÃ¤ubig: GAD (30.04.2013)

Date: Tue Apr 30 14:18:10 CEST 2013

Duration: 87:29 min

Pages: 41

Übersicht

Average Case Complexity

Uniforme Verteilung:
(alle Instanzen gleichwahrscheinlich)

$$t(n) = \frac{1}{|\mathcal{I}_n|} \sum_{i \in \mathcal{I}_n} T(i)$$

Tatsächliche Eingabeverteilung kann in der Praxis aber stark von uniformer Verteilung abweichen.

Dann

$$t(n) = \sum_{i \in \mathcal{I}_n} p_i \cdot T(i)$$

Aber: meist schwierig zu berechnen!

Beispiel: Binärzahl-Inkrementierung

increment(A)

Input : Array A mit Binärzahl in $A[0] \dots A[n-1]$,
in $A[n]$ steht eine 0

Output : inkrementierte Binärzahl in $A[0] \dots A[n]$

```
i = 0;
while (A[i] == 1) do
    A[i] = 0;
    i = i + 1;
A[i] = 1;
```

Durchschnittliche Laufzeit für Zahl mit n Bits?

Beispiel: Binärzahl-Inkrementierung

Analyse:

- Sei $\mathcal{I}_n$ die Menge der n-Bit-Instanzen.
- Für die Hälfte (1/2) der Zahlen $x_{n-1} \dots x_0 \in \mathcal{I}_n$ ist $x_0 = 0$

$\Rightarrow$ 1 Schleifendurchlauf

- Für die andere Hälfte gilt $x_0 = 1$.
  Bei diesen gilt wieder für die Hälfte (also insgesamt 1/4) der Zahlen $x_1 x_0 = 01$

$\Rightarrow$ 2 Schleifendurchläufe 11

- Für den Anteil $(1/2)^k$ der Zahlen gilt $x_{k-1}x_{k-2}\dots x_0 = 01\dots1$

$\Rightarrow$ k Schleifendurchläufe

Beispiel: Binärzahl-Inkrementierung

Durchschnittliche Laufzeit:

$$\begin{aligned} t(n) &= \frac{1}{|\mathcal{I}_n|} \overline{\sum_{i\in\mathcal{I}_n} T(i)} \\ &= \frac{1}{|\mathcal{I}_n|} \sum_{k=1}^{n} \frac{|\mathcal{I}_n|}{2^k} \cdot O(k) \\ &= \sum_{k=1}^{n} O\left(\frac{k}{2^k}\right) \\ &= O\left(\sum_{k=1}^{n} \frac{k}{2^k}\right) \stackrel{?}{=} O(1) \end{aligned}$$

$\frac{k}{2^k}$

Beispiel: Binärzahl-Inkrementierung

Lemma

$$\sum_{k=1}^{n} \frac{k}{2^k} \le 2 - \frac{n+2}{2^n}$$

Beispiel: Binärzahl-Inkrementierung

Lemma

$$\sum_{k=1}^{n} \frac{k}{2^k} \le 2 - \frac{n+2}{2^n}$$

Beweis

Induktionsanfang:

Für $n = 1$ gilt: $\sum_{k=1}^{1} \frac{k}{2^k} = \frac{1}{2} \le 2 - \frac{1+2}{2^1}$ ✓

Beispiel: Binärzahl-Inkrementierung

Lemma

$$\sum_{k=1}^{n} \frac{k}{2^k} \le 2 - \frac{n+2}{2^n}$$

Beweis

Induktionsanfang:

$$\text{Für } n = 1 \text{ gilt:} \quad \sum_{k=1}^{1} \frac{k}{2^k} = \frac{1}{2} \le 2 - \frac{1+2}{2^1} \quad \checkmark$$

Induktionsvoraussetzung:

$$\text{Für } n \text{ gilt:} \quad \sum_{k=1}^{n} \frac{k}{2^k} \le 2 - \frac{n+2}{2^n}$$

Beispiel: Binärzahl-Inkrementierung

Beweis.

Induktionsschritt: $n \to n+1$

$$\begin{aligned}
\sum_{k=1}^{n+1} \frac{k}{2^k} &= \left(\sum_{k=1}^{n} \frac{k}{2^k}\right) + \frac{n+1}{2^{n+1}} \\
&\le 2 - \frac{n+2}{2^n} + \frac{n+1}{2^{n+1}} \qquad \text{(laut Ind.vor.)} \\
&= 2 - \frac{2(n+2)}{2^{n+1}} + \frac{n+1}{2^{n+1}} = 2 - \frac{\cancel{2}n + \overset{3}{\cancel{4}} - n - \cancel{1}}{2^{n+1}} \\
&= 2 - \frac{n+3}{2^{n+1}} \\
&= 2 - \frac{(n+1)+2}{2^{n+1}}
\end{aligned}$$

$$\sum_{k=1}^{n} \frac{k}{2^k} = O(1)$$

□

Übersicht

Zufallsvariable

Definition

Für einen Wahrscheinlichkeitsraum mit Ergebnismenge Ω nennt man eine Abbildung $X : \Omega \mapsto \mathbb{R}$ (numerische) Zufallsvariable.

Zufallsvariable

Definition

Für einen Wahrscheinlichkeitsraum mit Ergebnismenge Ω nennt man eine Abbildung $X : \Omega \mapsto \mathbb{R}$ (numerische) Zufallsvariable.

Eine Zufallsvariable über einer endlichen oder abzählbar unendlichen Ergebnismenge heißt diskret.

Zufallsvariable

Definition

Für einen Wahrscheinlichkeitsraum mit Ergebnismenge Ω nennt man eine Abbildung $X : \Omega \mapsto \mathbb{R}$ (numerische) Zufallsvariable.

Eine Zufallsvariable über einer endlichen oder abzählbar unendlichen Ergebnismenge heißt diskret.

Der Wertebereich diskreter Zufallsvariablen

{1, −0,5}

$$W_X := X(\Omega) = \{x \in \mathbb{R} \mid \exists \omega \in \Omega \text{ mit } X(\omega) = x\}$$

ist ebenfalls endlich bzw. abzählbar unendlich.

Zufallsvariable

Definition

Für einen Wahrscheinlichkeitsraum mit Ergebnismenge Ω nennt man eine Abbildung $X : \Omega \mapsto \mathbb{R}$ (numerische) Zufallsvariable.

Eine Zufallsvariable über einer endlichen oder abzählbar unendlichen Ergebnismenge heißt diskret.

Der Wertebereich diskreter Zufallsvariablen

$$W_X := X(\Omega) = \{x \in \mathbb{R} \mid \exists \omega \in \Omega \text{ mit } X(\omega) = x\}$$

ist ebenfalls endlich bzw. abzählbar unendlich.

Schreibweise: $\Pr[X = x] := \Pr\left[X^{-1}(x)\right] = \sum_{\omega \in \Omega \mid X(\omega) = x} \Pr[\omega]$

Zufallsvariable

Beispiel

Wir ziehen aus einem Poker-Kartenspiel mit 52 Karten (13 von jeder Farbe) eine Karte.

Wir bekommen bzw. bezahlen einen bestimmten Betrag, je nachdem welche Farbe die Karte hat, z.B. 4 Euro für Herz, 7 Euro für Karo, −5 Euro für Kreuz und −3 Euro für Pik.

Wenn wir ein As ziehen, bekommen wir zusätzlich 1 Euro.

Zufallsvariable

Beispiel

Wir ziehen aus einem Poker-Kartenspiel mit 52 Karten (13 von jeder Farbe) eine Karte.

Wir bekommen bzw. bezahlen einen bestimmten Betrag, je nachdem welche Farbe die Karte hat, z.B. 4 Euro für Herz, 7 Euro für Karo, −5 Euro für Kreuz und −3 Euro für Pik.

Wenn wir ein As ziehen, bekommen wir zusätzlich 1 Euro.

$\Omega = \{\heartsuit A, \heartsuit K, \ldots, \heartsuit 2, \diamondsuit A, \diamondsuit K, \ldots, \diamondsuit 2, \clubsuit A, \clubsuit K, \ldots, \clubsuit 2, \spadesuit A, \spadesuit K, \ldots, \spadesuit 2\}$.

X sei der Geldbetrag den wir bekommen bzw. bezahlen.

Effizienz | Erwartete Laufzeit

Zufallsvariable

Beispiel

Wir ziehen aus einem Poker-Kartenspiel mit 52 Karten (13 von jeder Farbe) eine Karte.

Wir bekommen bzw. bezahlen einen bestimmten Betrag, je nachdem welche Farbe die Karte hat, z.B. 4 Euro für Herz, 7 Euro für Karo, −5 Euro für Kreuz und −3 Euro für Pik.

Wenn wir ein As ziehen, bekommen wir zusätzlich 1 Euro.

$\Omega = \{\heartsuit A, \heartsuit K, \ldots, \heartsuit 2, \diamondsuit A, \diamondsuit K, \ldots, \diamondsuit 2, \clubsuit A, \clubsuit K, \ldots, \clubsuit 2, \spadesuit A, \spadesuit K, \ldots, \spadesuit 2\}$.

X sei der Geldbetrag den wir bekommen bzw. bezahlen.

$W_X = \{-5, -4, -3, -2, 4, 5, 7, 8\}$

$\Pr[X = -3] = \Pr[\spadesuit K] + \ldots + \Pr[\spadesuit 2] = 12/52 = 3/13$

Erwartungswert

Definition

Für eine diskrete Zufallsvariable X ist der Erwartungswert definiert als

$$\mathbb{E}[X] := \sum_{x \in W_X} x \cdot \Pr[X = x] = \sum_{\omega \in \Omega} X(\omega) \cdot \Pr[\omega]$$

sofern $\sum_{x \in W_X} |x| \cdot \Pr[X = x]$ konvergiert (absolute Konvergenz).

Effizienz | Erwartete Laufzeit

Erwartungswert

Definition

Für eine diskrete Zufallsvariable X ist der Erwartungswert definiert als

$$\mathbb{E}[X] := \sum_{x \in W_X} x \cdot \Pr[X = x] = \sum_{\omega \in \Omega} X(\omega) \cdot \Pr[\omega]$$

sofern $\sum_{x \in W_X} |x| \cdot \Pr[X = x]$ konvergiert (absolute Konvergenz).

Bei endlicher Ereignismenge und gleichwahrscheinlichen Ereignissen entspricht der Erwartungswert dem Durchschnitt:

$$\mathbb{E}[X] = \sum_{x \in W_X} x \cdot \Pr[X = x] = \sum_{\omega \in \Omega} X(\omega) \cdot \frac{1}{|\Omega|} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} X(\omega)$$

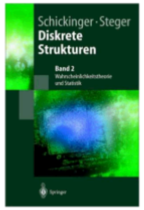
Erwartungswert

Beispiel

(Beispiel wie zuvor)

$$\mathbb{E}[X] = 4\cdot\frac{12}{52} + 5\cdot\frac{1}{52} + 7\cdot\frac{12}{52} + 8\cdot\frac{1}{52} + (-5)\cdot\frac{12}{52} + (-4)\cdot\frac{1}{52} + (-3)\cdot\frac{12}{52} + (-2)\cdot\frac{1}{52} = \frac{43}{52}$$

Wir bekommen also im Erwartungswert $\frac{43}{52}$ Euro pro gezogener Karte.



Grundlagen zu diskreter Wahrscheinlichkeitstheorie findet man z.B. in folgendem Buch:

Th. Schickinger, A. Steger
Diskrete Strukturen – Band 2
(Wahrscheinlichkeitstheorie und Statistik)
Springer-Verlag, 2001.

Erwartungswert

Beispiel

Münze werfen, bis sie zum ersten Mal Kopf zeigt

Zufallsvariable k: Anzahl der Versuche

$\Pr[k=x] = \left(\frac{1}{2}\right)^x$

k ungerade: Spieler bezahlt etwas an die Bank
k gerade: Spieler bekommt etwas von der Bank

Zufallsvariable X: Gewinnbetrag der Bank

Erwartungswert

Beispiel

Münze werfen, bis sie zum ersten Mal Kopf zeigt

Zufallsvariable k: Anzahl der Versuche

$\sum_{k=1}^{\infty} \frac{k}{2^k}$

k ungerade: Spieler bezahlt etwas an die Bank
k gerade: Spieler bekommt etwas von der Bank

Zufallsvariable X: Gewinnbetrag der Bank

Variante 1: Spieler bezahlt / bekommt k Euro
$\mathbb{E}[X]$ existiert (absolute Konvergenz)

$1\cdot\left(\frac{1}{2}\right)^1 + 2\left(\frac{1}{2}\right)^2 + 3\left(\frac{1}{2}\right)^3 + 4\left(\ldots\right.$

Variante 2: Spieler bezahlt / bekommt 2^k Euro
$\mathbb{E}[X]$ existiert nicht (keine Konvergenz)

$2^1\cdot\left(\frac{1}{2}\right)^1 - 2^2\cdot\left(\frac{1}{2}\right)^2 + 2^3\cdot\left(\frac{1}{2}\right)^3 - \ldots$

$1 - 1 + 1 - 1 + \ldots$

Variante 3: Spieler bezahlt / bekommt $\boxed{\frac{2^k}{k}}$ Euro
$\mathbb{E}[X]$ existiert nicht (Konvergenz, aber keine absolute)

Erwartungswert zusammengesetzter Zufallsvariablen

Satz (Linearität des Erwartungswerts)

Für Zufallsvariablen $\boxed{X_1, \ldots, X_n}$ *und*

$$\boxed{X} := a_1 X_1 + \ldots + a_n X_n$$

mit $a_1, \ldots, a_n \in \mathbb{R}$ *gilt*

$$\mathbb{E}[X] = a_1\mathbb{E}[X_1] + \ldots + a_n\mathbb{E}[X_n].$$

Interessant ist für uns vor allem der einfache Fall:

X_1

$$X := X_1 + \ldots + X_n$$

$X := aX_1 + b$

mit

$$\mathbb{E}[X] = \mathbb{E}[X_1] + \ldots + \mathbb{E}[X_n].$$

$\mathbb{E}[X] = a\mathbb{E}[X_1] + b$

Beispiel: Suche in statischer Liste

$\longrightarrow \boxed{3} \longrightarrow \boxed{5} \longrightarrow \boxed{2} \longrightarrow \boxed{1} \longrightarrow \boxed{4} \longrightarrow$

- gegeben: Liste mit Elementen $1, \ldots, m$
- search(i): lineare Suche nach Element i ab Listenanfang

  s_i Position von Element i in der Liste ($1 \hat{=}$ Anfang)

  p_i Wahrscheinlichkeit für Zugriff auf Element i

Effizienz | Erwartete Laufzeit

Beispiel: Suche in statischer Liste

$\longrightarrow \boxed{3} \longrightarrow \boxed{5} \longrightarrow \boxed{2} \longrightarrow \boxed{1} \longrightarrow \boxed{4} \longrightarrow$

- gegeben: Liste mit Elementen $1, \ldots, m$
- search(i): lineare Suche nach Element i ab Listenanfang

  s_i Position von Element i in der Liste ($1 \hat{=}$ Anfang)

  p_i Wahrscheinlichkeit für Zugriff auf Element i

Erwartete Laufzeit der Operation search(i) mit zufälligem i:

$$\mathbb{E}[T(\text{search}(i))] = O\left(\sum_i p_i s_i\right)$$

Effizienz | Erwartete Laufzeit

Beispiel: Suche in statischer Liste

$\longrightarrow \boxed{3} \longrightarrow \boxed{5} \longrightarrow \boxed{2} \longrightarrow \boxed{1} \longrightarrow \boxed{4} \longrightarrow$

- gegeben: Liste mit Elementen $1, \ldots, m$
- search(i): lineare Suche nach Element i ab Listenanfang

  s_i Position von Element i in der Liste ($1 \hat{=}$ Anfang)

  p_i Wahrscheinlichkeit für Zugriff auf Element i

Erwartete Laufzeit der Operation search(i) mit zufälligem i:

$$\mathbb{E}[T(\text{search}(i))] = O\left(\sum_i p_i s_i\right)$$

Erwartete Laufzeit $t(n)$ für n Zugriffe bei statischer Liste:

$$t(n) = \mathbb{E}[T(n \times \text{search}(i))] = n \cdot \mathbb{E}[T(\text{search}(i))] = O\left(n \sum_i p_i s_i\right)$$

Effizienz | Erwartete Laufzeit

Beispiel: Suche in statischer Liste

Optimale Anordnung?

$\Rightarrow$ wenn für alle Elemente i, j mit $p_i > p_j$ gilt, dass $s_i < s_j$, d.h. die Elemente nach Zugriffswahrscheinlichkeit sortiert sind

o.B.d.A. seien die Indizes so, dass $p_1 \geq p_2 \geq \ldots \geq p_m$

- Optimale Anordnung: $s_i = i$
- Optimale erwartete Laufzeit: $\text{opt} = \sum_i p_i \boxed{\cdot\, i}$

Einfach: wenn die Zugriffswahrscheinlichkeiten bekannt sind
$\Rightarrow$ optimale erwartete Laufzeit durch absteigende Sortierung nach p_i

Problem: was wenn die Wahrscheinlichkeiten p_i unbekannt sind?

Beispiel: Suche in selbstorganisierender Liste

$$\longrightarrow \boxed{16} \longrightarrow \boxed{9} \longrightarrow \boxed{4} \longrightarrow \boxed{23} \longrightarrow \boxed{18} \longrightarrow$$

Move-to-Front Rule:

Verschiebe nach jeder erfolgreichen Suche das gefundene Element an den Listenanfang

Bsp.: Ausführung von search(4) ergibt

$$\longrightarrow \boxed{4} \longrightarrow \boxed{16} \longrightarrow \boxed{9} \longrightarrow \boxed{23} \longrightarrow \boxed{18} \longrightarrow$$

Beispiel: Suche in selbstorganisierender Liste

Erwartete Laufzeit $t(n)$ bei dynamischer Liste:

$$\mathbb{E}[T(\text{search}(i))] = O\left(\sum_i p_i \cdot \mathbb{E}[s_i]\right)$$

Satz

Ab dem Zeitpunkt, wo auf jedes Element mindestens einmal zugegriffen wurde, ist die erwartete Laufzeit der search-Operation unter Verwendung der Move-to-Front Rule höchstens $2 \cdot \text{opt}$.

Beispiel: Suche in selbstorganisierender Liste

Beweis.

Betrachte zwei feste Elemente i und j

t_0 Zeitpunkt der letzten Suchoperation auf i oder j

- bedingte Wahrscheinlichkeit: $\Pr[A \mid B] = \dfrac{\Pr[A \wedge B]}{\Pr[B]}$
- $\Pr[C \mid (C \vee D)] = \dfrac{\Pr[C \wedge (C \vee D)]}{\Pr[C \vee D]} = \dfrac{\Pr[C]}{\Pr[C \vee D]}$
- $\Pr[\text{search}(j) \text{ bei } t_0 \mid \text{search}(i \vee j) \text{ bei } t_0] = \dfrac{p_j}{p_i + p_j}$
- mit Wsk. $\dfrac{p_i}{p_i + p_j}$ steht i vor j und mit Wsk. $\dfrac{p_j}{p_i + p_j}$ steht j vor i

Beispiel: Suche in selbstorganisierender Liste

Beweis.

Betrachte nun nur ein festes Element i

- Definiere Zufallsvariablen $X_j \in \{0, 1\}$ für $j \neq i$:

$$X_j = 1 \quad \Leftrightarrow \quad j \text{ vor } i \text{ in der Liste}$$

- Erwartungswert:

$$\begin{aligned} \mathbb{E}[X_j] &= 0 \cdot \Pr[X_j = 0] + 1 \cdot \Pr[X_j = 1] \\ &= \cancel{\Pr[\text{letzte Suche}} \text{ nach } i/j \text{ war nach } j] \\ &= \frac{p_j}{p_i + p_j} \end{aligned}$$

Beispiel: Suche in selbstorganisierender Liste

Beweis.

Betrachte nun nur ein festes Element i

- Definiere Zufallsvariablen $X_j \in \{0, 1\}$ für $j \neq i$:

$$X_j = 1 \quad \Leftrightarrow \quad j \text{ vor } i \text{ in der Liste}$$

- Erwartungswert:

$$\begin{aligned} \mathbb{E}[X_j] &= 0 \cdot \Pr[X_j = 0] + 1 \cdot \Pr[X_j = 1] \\ &= \Pr[\text{letzte Suche nach } i/j \text{ war nach } j] \\ &= \frac{p_j}{p_i + p_j} \end{aligned}$$

Beispiel: Suche in selbstorganisierender Liste

Beweis.

- Listenposition von Element i: $\quad 1 + \sum_{j \neq i} X_j$
- Erwartungswert der Listenposition von Element i:

$$\begin{aligned} \mathbb{E}[s_i] &= \mathbb{E}\left[1 + \sum_{j \neq i} X_j\right] \\ &= 1 + \mathbb{E}\left[\sum_{j \neq i} X_j\right] = 1 + \sum_{j \neq i} \mathbb{E}[X_j] \\ \mathbb{E}[s_{i,\mathrm{MTF}}] &= 1 + \sum_{j \neq i} \frac{p_j}{p_i + p_j} \end{aligned}$$

Beispiel: Suche in selbstorganisierender Liste

Beweis.

Erwartete Laufzeit der search-Operation:

$$\begin{aligned} \mathbb{E}[T_{\mathrm{MTF}}] &= \sum_i p_i \left(1 + \sum_{j \neq i} \frac{p_j}{p_i + p_j}\right) \\ &= \sum_i \left(p_i + \sum_{j \neq i} \frac{p_i p_j}{p_i + p_j}\right) = \sum_i \left(p_i + 2 \sum_{j<i} \frac{p_i p_j}{p_i + p_j}\right) \\ &= \sum_i p_i \left(1 + 2 \sum_{j<i} \frac{p_j}{p_i + p_j}\right) \leq \sum_i p_i \left(1 + \overbrace{2 \sum_{j<i} 1}^{2(i-1)}\right) \\ &\leq \sum_i p_i \cdot (2i \cancel{-1}) < \sum_i p_i \cdot 2i = 2 \cdot \mathrm{opt} \end{aligned}$$

□

Übersicht

Sequenzen

Sequenz: lineare Struktur

$$s = \langle e_0, \ldots, e_{n-1} \rangle$$

(Gegensatz: verzweigte Struktur in Graphen, fehlende Struktur in Hashtab.)

Klassische Repräsentation:

- (Statisches) Feld / Array:
  direkter Zugriff über $s[i]$
  - Vorteil: Zugriff über Index, homogen im Speicher
  - Nachteil: dynamische Größenänderung schwierig
- Liste:
  indirekter Zugriff über Nachfolger / Vorgänger
  - Vorteil: Einfügen / Löschen von Teilsequenzen
  - Nachteil: kein Zugriff per Index, Elemente über Speicher verteilt

Sequenzen

Operationen:

- $\langle e_0, \ldots, e_{n-1} \rangle[i]$ liefert Referenz auf e_i
- $\langle e_0, \ldots, e_{n-1} \rangle.\text{get}(i) = e_i$
- $\langle e_0, \ldots, e_{i-1}, e_i, \ldots, e_{n-1} \rangle.\text{set}(i, e) = \langle e_0, \ldots, e_{i-1}, e, \ldots, e_{n-1} \rangle$
- $\langle e_0, \ldots, e_{n-1} \rangle.\text{pushBack}(e) = \langle e_0, \ldots, e_{n-1}, e \rangle$
- $\langle e_0, \ldots, e_{n-1} \rangle.\underline{\text{popBack}}() = \langle e_0, \ldots, e_{n-2} \rangle$
- $\langle e_0, \ldots, e_{n-1} \rangle.\underline{\text{size}}() = n$

Übersicht

Sequenz als Feld

Problem:

- Beim Anlegen des Felds ist nicht bekannt, wieviele Elemente es enthalten wird
- Nur Anlegen von statischen Feldern möglich
  (s = new ElementTyp[w])

Lösung: Datenstruktur für dynamisches Feld

Dynamisches Feld

Erste Idee:

- Immer dann, wenn Feld s nicht mehr ausreicht: generiere neues Feld der Größe $w + c$ für ein festes c

$s[0]$	$s[1]$	$s[2]$	...	$s[w-1]$	andere Daten

$\Downarrow$ Kopieren in neues größeres Feld

$s[0]$	$s[1]$	$s[2]$	...	$s[w-1]$	$s[w]$	...	$s[w+c-1]$

Dynamisches Feld

Zeitaufwand für Erweiterung: $O(w + c) = O(w)$

$s[0]$	$s[1]$	$s[2]$	...	$s[w-1]$	andere Daten

$\Downarrow$ Kopieren in neues größeres Feld

$s[0]$	$s[1]$	$s[2]$	...	$s[w-1]$	$s[w]$	...	$s[w+c-1]$

Zeitaufwand für n pushBack Operationen:

- Aufwand von $O(w)$ nach jeweils c Operationen
- Gesamtaufwand:

$$O\left(\sum_{i=1}^{n/c} c \cdot i\right) = O\left(n^2\right)$$