Willkommen

Algorithmen I Tutorium 19

Wer? Florian Tobias Schandinat

Wo? 50.34, Raum -118

Wann? jeden Donnerstag 15:45-17:15

Material online

http://github.com/schandinat/algorithmen1_ss11

1. Übungsblatt

$O(1) \not\subseteq O(0)$

1. Übungsblatt

$O(1) \not\subseteq O(0)$

Annahme: $\exists c \in \mathbb{R}_+ \exists n_0 \in \mathbb{N} : 1 \leq c \cdot 0 \qquad \forall n > n_0$

1. Übungsblatt

$O(1) \not\subseteq O(0)$

Annahme: $\exists c \in \mathbb{R}_+ \exists n_0 \in \mathbb{N} : 1 \leq c \cdot 0 \qquad \forall n > n_0$

da $1 \neq 0 = c \cdot 0 \qquad \forall c \in \mathbb{R} \forall n \in \mathbb{N}$

$\Longrightarrow O(1) \not\subseteq O(0)$

1. Übungsblatt

$O(1) \not\subseteq O(0)$

Annahme: $\exists c \in \mathbb{R}_+ \exists n_0 \in \mathbb{N} : 1 \leq c \cdot 0 \qquad \forall n > n_0$

da $1 \neq 0 = c \cdot 0 \qquad \forall c \in \mathbb{R} \forall n \in \mathbb{N}$

$\Longrightarrow O(1) \not\subseteq O(0)$

Anschaulich:

$O(0)$ der Codeabschnitt wird nicht ausgeführt

$O(1)$ der Codeabschnitt wird k-Mal ausgeführt, $k \in \mathbb{N}_0$

Erinnerung: Mastertheorem

Mastertheorem

$$T(n) = 1, n = 1$$

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n), n > 1$$

$$\begin{aligned} \theta(n^{log_b a}) &, \quad f(n) = O(n^{log_b(a)-\epsilon}), \epsilon > 0 \\ \theta(n^{log_b a} \cdot log(n)) &, \quad f(n) = \theta(n^{log_b a}) \\ \theta(f(n)) &, \quad f(n) = \Omega(n^{log_b(a)+\epsilon}), \epsilon > 0 \end{aligned}$$

Mastertheorem – Übung 1

binary search

PROCEDURE BinaereSuche(A, v, l, r)
if $l > r$ then
    return NIL
$m = \lfloor \frac{l+r}{2} \rfloor$
if $v == A[m]$ then
    return m
else if $v > A[m]$ then
    return BinaereSuche(A, v, m + 1, r)
else
    return BinaereSuche(A, v, l, m - 1)
end if

Mastertheorem – Übung 1

binary search

```
PROCEDURE BinaereSuche(A, v, l, r)
if l > r then
        return NIL
m = ⌊(l+r)/2⌋
if v == A[m] then
        return m
else if v > A[m] then
        return BinaereSuche(A, v, m + 1, r)
else
        return BinaereSuche(A, v, l, m - 1)
end if
```

$$T(n) = 1 \cdot T\left(\frac{n}{2}\right) + 1$$

Mastertheorem – Übung 1

binary search

```
PROCEDURE BinaereSuche(A, v, l, r)
if l > r then
        return NIL
m = ⌊(l+r)/2⌋
if v == A[m] then
        return m
else if v > A[m] then
        return BinaereSuche(A, v, m + 1, r)
else
        return BinaereSuche(A, v, l, m - 1)
end if
```

$$T(n) = 1 \cdot T\left(\frac{n}{2}\right) + 1$$
$$\Longrightarrow \theta(log(n))$$

Mastertheorem – Übung 2

binary tree traversal

```
PROCEDURE tree(n)
if n.left ≠ NIL
        tree(n.left)
if n.right ≠ NIL
        tree(n.right)
```

Mastertheorem – Übung 2

binary tree traversal

PROCEDURE tree(n)
if $n.left \neq NIL$
    tree(n.left)
if $n.right \neq NIL$
    tree(n.right)

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + 1$$

Mastertheorem – Übung 2

binary tree traversal

PROCEDURE tree(n)
if $n.left \neq NIL$
    tree(n.left)
if $n.right \neq NIL$
    tree(n.right)

$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + 1$
$\Longrightarrow \theta(n)$

Mastertheorem – Weitere Übungen

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + n$$

Mastertheorem – Weitere Übungen

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + n$$
$$\Longrightarrow \theta(n \cdot log(n))$$

Mastertheorem – Weitere Übungen

$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + n$
$\Longrightarrow \theta(n \cdot log(n))$

$T(n) = 4 \cdot T\left(\frac{n}{2}\right) + n^3$

Mastertheorem – Weitere Übungen

$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + n$
$\Longrightarrow \theta(n \cdot log(n))$

$T(n) = 4 \cdot T\left(\frac{n}{2}\right) + n^3$
$\Longrightarrow \theta(n^3)$

Amortisierte Analyse

Amortisierte Analyse

Übung

$$T(\phi(n)) = 1, \nexists i \in \mathbb{N} : i^2 = n$$

$$T(\phi(n)) = n, \exists i \in \mathbb{N} : i^2 = n$$

Hashing – Grundlagen

- U Universalmenge
- K Schlüsselmenge
- h Hashfunktion
  $h : U \rightarrow K$

Hashing – Grundlagen

- U Universalmenge
- K Schlüsselmenge
- h Hashfunktion
  $h : U \to K$

Hinweis: Bei $|K| < |U|$ immer Kollisionen möglich

Hashing – Kollisionsauflösung

- Verkettung
- Offene Adressierung
  - lineares Sondieren
  - quadratisches Sondieren
  - doppeltes Hashing

Hashing – Übung

$U = \{i \in \mathbb{N}_0 | i \leq 20\}, K = \{i \in \mathbb{N}_0 | i \leq 6\}, h(k) = k \mod 7$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)

Hashing – Übung

$U = \{i \in \mathbb{N}_0 | i \leq 20\}, K = \{i \in \mathbb{N}_0 | i \leq 6\}, h(k) = k \mod 7$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)

Hashing – Übung

$U = \{i \in \mathbb{N}_0 | i \leq 20\}, K = \{i \in \mathbb{N}_0 | i \leq 6\}, h(k) = k \mod 7$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)

Hashing – Übung

$U = \{i \in \mathbb{N}_0 | i \leq 20\}, K = \{i \in \mathbb{N}_0 | i \leq 6\}, h(k) = k \mod 7$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)
- DELETE(4)

Hashing – Übung

$U = \{i \in \mathbb{N}_0 | i \leq 20\}, K = \{i \in \mathbb{N}_0 | i \leq 6\}, h(k) = k \mod 7$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)
- DELETE(4)
- SEARCH(2)

Hashing – Übung

$U = \{i \in \mathbb{N}_0 | i \leq 20\}, K = \{i \in \mathbb{N}_0 | i \leq 6\}, h(k) = k \mod 7$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)
- DELETE(4)
- SEARCH(2)
- SEARCH(5)

Hashing – Übung

$U = \{i \in \mathbb{N}_0 | i \leq 20\}, K = \{i \in \mathbb{N}_0 | i \leq 6\}, h(k) = k \mod 7$

Skizziere die folgenden Operationen mit Verkettung; offener Adressierung mit linearer Konfliktauflösung

- INSERT(2)
- INSERT(3)
- INSERT(4)
- INSERT(10)
- INSERT(11)
- DELETE(4)
- SEARCH(2)
- SEARCH(5)
- SEARCH(11)

Vielen Dank für die Aufmerksamkeit!