Wiederholung

Informatik 1 für 2-Fach und Nebenfach (Inf-I1-2FNF)
Informatik für Naturwissenschaften (Inf-InfNat)

Wintersemester 2019/2020
Prof. Dr. Andreas Mühling, CAU Kiel

Laufzeit von Algorithmen

Können Informatiker*innen nicht rechnen?

Oder sind sie zumindest sehr unpräzise?

Wieso sollte $n^2 + n$ genauso schnell wachsen wie n^2?

Laufzeit von Algorithmen

Einerseits (für $\mathrm{n} \in \mathbb{N}$):

$$n^2 + n > n^2$$

Also ist n^2 eine untere Schranke für das Wachstum von $n^2 + n$.

Laufzeit von Algorithmen

Andererseits (für $\mathrm{n} \in \mathbb{N}$):

$$n^2 + n \leq n^2 + n^2 \leq 2n^2 \text{ für } \mathrm{n} \in \mathbb{N}$$

Also ist $2n^2$ eine obere Schranke für das Wachstum von $n^2 + n$.

Nach den Regeln der Vorlesung wächst $n^2 + n$ damit genauso schnell wie n^2.

Laufzeit von Algorithmen

Andere Sichtweise:

Gegeben zwei Funktionen $f(x)$ und $\mathrm{g}(x)$, beide $\mathbb{N} \to \mathbb{N}$

Betrachte: $\lim_{\mathrm{x} \to \infty} \frac{f(x)}{g(x)}$

Laufzeit von Algorithmen

Fall 1: $\lim_{x \to \infty} \frac{f(x)}{g(x)} = \infty$

=> f wächst (asymptotisch) stärker als g

Fall 2: $\lim_{x \to \infty} \frac{f(x)}{g(x)} = 0$

=> f wächst (asymptotisch) langsamer als g

Fall 3: $\lim_{x \to \infty} \frac{f(x)}{g(x)} = c$

$\Rightarrow$f wächst (asymptotisch) genauso schnell wie g

(Fall 4: Grenzwert existiert nicht – für „unsere“ Funktionen irrelevant)

Laufzeit von Algorithmen

$$\lim_{x \to \infty} \frac{x^2 + x}{x^2} = ?$$

Regel von L'Hôpital

$$= \lim_{x \to \infty} \frac{2x \quad + 1}{2x}$$

Regel von L'Hôpital

$$= \lim_{x \to \infty} \frac{2}{2} = 1$$

=> $x^2 + x$ und x^2 wachsen gleich schnell!

Laufzeit von Algorithmen

Zu jeder Funktion gibt es also eine (unendliche) Menge an anderen Funktionen, die genauso schnell wachsen.

Die Idee der Laufzeitkomplexität in der Informatik ist es:

- Das Wachstum der Laufzeit in Abhängigkeit von der Eingabegröße als Funktion darzustellen aber
- anstelle dieser Funktion einen möglichst einfachen „Repräsentanten“ anzugeben, der (asymptotisch) gleich schnell wächst.
- Interessant: Wort-case Eingabe, „scharfe“ obere Schranke…

Laufzeit von Algorithmen

Das Erlaubt Aussagen der Form:

„(Die Laufzeit von) Selectionsort wächst quadratisch (mit der Eingabegröße).“

Und Vergleiche der Form:

„Mergesort ist quasi-linear und damit schneller als alle quadratischen Sortierverfahren“.

Reflexionsfragen

- Ist mir klar, dass es zu einem Problem viele Algorithmen gibt, die sich in ihrer Laufzeit unterscheiden?
- Ist mir klar, wieso die Laufzeit von Algorithmen nicht einfach gemessen und dann geeignet angegeben wird?
- Ist mir klar, wieso Selectionsort „quadratische Laufzeit“ hat?
- Ist mir klar, wieso die binäre Suche „logarithmische Laufzeit“ hat?

Reflexionsfragen

- Ist mir klar, wie die lineare Suche funktioniert?
- Ist mir klar, wie die binäre Suche funktioniert?
- Ist mir klar, wie Selectionsort funktioniert?
- Ist mir klar, wie Mergesort funktioniert?

Unterprogramme

Drei Blicke auf Unterprogramme:

1. Kurzschreibweise für einen Codeblock, der immer wieder auftaucht.
2. „Autarker“ Codeblock, der aus einer Eingabe (Parameter) einen Effekt (Prozedur) oder eine Ausgabe (Funktion) generiert.
3. Möglichkeit, rekursive Algorithmen umzusetzen.

Reflexionsfragen

- Ist mir klar, wie ein Unterprogramm aufgerufen wird?
- Ist mir klar, wie ein Unterprogramm definiert wird?
- Ist mir klar, was ein „Lebensbereich“ ist?
- Ist mir klar, wie eine Funktion einen Wert zurückgibt?

Aufgaben

- Implementieren Sie die Multiplikation zweier Zahlen durch wiederholte Addition:
  - Als Programm
  - Als Unterprogramm ohne Rekursion
  - Als rekursives Unterprogramm

- (*) Wie können lineare Suche und binäre Suche rekursiv implementiert werden?

Reflexionsfragen

- Ist mir klar, was eine Variable ist?
- Ist mir klar, was der Unterschied zwischen if und while ist?
- Ist mir klar, was der Unterschied zwischen while und for ist?
- Ist mir klar, wie ein Python-Programm abläuft?

Aufgaben

Implementieren Sie eine (annähernde) Berechnung der Wurzelfunktion durch Intervallschachtelung:

1. Bestimmen Sie ein Intervall, in dem die Wurzel sicher liegt
2. Teilen Sie dieses Intervall und entscheiden Sie, ob die Wurzel in der linken oder rechten Hälfte liegen muss.
3. Wiederholen Sie 2. solange, bis das Intervall eine bestimmte Größte unterschritten hat (z.B. 0.000001).
4. Geben Sie die Mitte des Intervalls als Ergebnis aus.

Reflexionsfragen

- Ist mir klar, was ein Primär- und ein Fremdschlüssel ist?
- Ist mir klar, wie das Kreuzprodukt zweier Relationen gebildet wird?
- Ist mir klar, wieso man üblicherweise ein Kreuzprodukt nicht ohne zusätzliche Selektions-Bedingungen verwendet?
- Ist mir klar, wie Gruppierung (group by) funktioniert?
- Ist mir klar, was der Unterschied zwischen „where“ und „having“ in einer SQL-Anfrage ist?

Reflexionsfragen

- Ist mir klar, was EBNF-Grammatiken und reguläre Ausdrücke beschreiben?
- Ist mir klar, wie eine Ableitung eines Wortes aus einer EBNF-Grammatik funktioniert?
- Ist mir klar, was ein Escape-Zeichen ist und warum man es benötigt?