Informatik II:
Algorithmen und Datenstrukturen
SS 2017

Vorlesung 11a, Dienstag, 11. Juli 2017
(Editierdistanz, Teil 1)

Prof. Dr. Hannah Bast
Lehrstuhl für Algorithmen und Datenstrukturen
Institut für Informatik
Universität Freiburg

Blick über die Vorlesung heute

- Organisatorisches
  - Erfahrungen ÜB10 — Routenplanung in BW / SL
- Inhalt
  - Editierdistanz — Motivation + Notation
  - Rekursive Berechnung — monotone Folgen
  - Rekursive Implementierung — Code + Laufzeit
  - ÜB11: diesmal wieder ein reines Theorieblatt

Erfahrungen ÜB10 1/2

- Zusammenfassung
  - Schöne Aufgabe mit Praxisbezug, hat vielen Spaß gemacht
  - Einige Kämpfe mit dem Design / Code / Debuggen

    Extra weniger Vorgaben diesmal, ist ja schon das ÜB10

    Es hat sich aber gezeigt, dass viele noch Probleme mit der Umsetzung eines abstrakten Algorithmus in Code haben
  - Einigen ist offenbar nicht klar, wie Sie richtig testen sollen

    Jede Funktion einzeln, siehe 10 Gebote auf dem ÜB10

    Grundsätzlich auf kleinen Beispielen, das geht **immer**
  - Einige haben etwas Zeit gebraucht, um zu verstehen, was der "längste kürzeste Weg" ist

Erfahrungen ÜB10 2/2

- Ergebnis
  - Erklärung längster kürzester Pfad ab einem Startknoten **s**

    Der Zielknoten v, für den dist(s, v) maximal ist

    Wobei dist(s, v) der kürzeste Weg von s nach v ist
  - Längster kürzester Weg ab der Freiburger TF in BaWü:

    http://share.mapbbcode.org/cusya "Auf zur Kunigundenkapelle"
  - Längster kürzester Weg ab der Informatik im Saarland:

    http://share.mapbbcode.org/psdbh eine Kreuzung, laangweilig

Editierdistanz 1/9

- Motivation
  - Es gibt viele Anwendungen, wo man ein Maß für die Ähnlichkeit zwischen zwei Zeichenketten braucht

    Drei Beispiele dazu auf der nächsten Folie
  - Die **Editierdistanz** ist ein solches Ähnlichkeitsmaß

    Das in der Praxis am häufigsten verwendete Maß

    Definition auf Folie 7

    Benannt nach dem **Edi-Tier**

Editierdistanz 2/9

- Anwendungsbeispiele
  - Beispiel 1: Dubletten in Adressdatenbanken

    Hein Blöd, 27568 Bremerhaven
    Hein Bloed, 27568 Bremerhafen
    Hein Doof, 27478 Cuxhaven

  - Beispiel 2: Produktsuche

    Memori Stik

  - Beispiel 3: Websuche

    eyjaföllajaküll
    semesta verien 2017

Editierdistanz 3/9

- Definition Editierdistanz … alternativ: Levenshtein-Distanz
  - Gegeben zwei Zeichenketten x und y
  - ED(x, y) = die minimale Anzahl der folgenden Operationen, die man braucht, um x in y zu transformieren:

    Einfügen **(insert)** eines Buchstabens

    Ersetzen **(replace)** eines Buchstabens durch einen anderen

    Löschen **(delete)** eines Buchstabens
  - Die Position einer Operation ist die Stelle im String, an der etwas geändert wird … siehe Beispiel nächste Folie

    Positionen fangen hier und in der Folge mit **1** an, nicht 0 (weil das intuitiver ist bei der mathematischen Analyse)

Editierdistanz 4/9

DOOF
SAUDOOF

- Beispiel
  - x = DOOF, y = BLOED … ED(x, y) = ?

1 2 3 4
D O O F
B O O F — REPLACE(1, B)
B L O F — REPLACE(2, L)
B L O E F — INSERT(4, E)
B L O E D — REPLACE(5, D)

Das ist eine monotone Folge (s. Folie 14)
$1 < 2 < 4 < 5$

Damit haben wir erstmal nur gezeigt: $ED(x, y) \leq 4$

Editierdistanz 5/9

- Notation

  - Mit ε bezeichnen wir das leere Wort
  - Mit $|x|$ bezeichnen wir die Länge von x (= Anzahl Zeichen)
  - Mit $x[i..j]$ bezeichnen wir die Teilfolge der Zeichen i bis j der Zeichenkette x, wobei $1 \leq i \leq j \leq |x|$

  Wie gesagt: Positionen / Indizes fangen mit **1** an, nicht 0

  $x = \overset{1}{B}\overset{2}{L}\overset{3}{O}\overset{4}{E}\overset{5}{D}$, $|x| = 5$, $x[2..4] = LOE$

Editierdistanz 6/9

- Ein paar einfache Eigenschaften
  - $ED(x, y) = ED(y, x)$
  - $ED(x, \varepsilon) = |x|$
  - $ED(x, y) \geq abs(|x| - |y|)$ $\quad$ $abs(x) = x > 0 ~?~ x : -x$
  - $ED(x, y) \leq max(|x|, |y|)$
  - $ED(x, y) \leq ED(x[1..n-1], y[1..m-1]) + 1$ $\quad$ $n = |x|, m = |y|$
  - Die Beweise sind alle einfache Zwei- oder Dreizeiler

    Sehr gute Übung zur Klausurvorbereitung und zur Überwindung der Mathe-Phobie

Editierdistanz 7/9

- Lösungsidee 1: möglichst viel "erhalten"
  - ED("VERIEN", "FERIEN") = 1

    Einfach, weil die Zeichenketten größtenteils gleich
  - ED("SEMESTERFERIEN", "SEMESTERVERIEN") = 1

    Dito … dann auch für längere Zeichenketten noch einfach
  - ED("MEXIKO", "AMERIKA") = 3

    Auch hier gibt es noch relativ große Übereinstimmung
  - ED("AAEBEAABEAREEAE", "RBEAAEEBAAAEBBAE") = ?

    Spätestens hier wird es sehr schwierig mit dieser Idee

Editierdistanz 8/9

- Lösungsidee 2: in zwei Hälften teilen
  - In zwei gleich große Hälften teilen und die dann jeweils rekursiv lösen

    ED(GRAU, RAUM) = 2

    ED(GR, RA) = 2

    ED(AU, UM) = 2

  - **Keine gute Idee:** die ED zwischen den Hälften hat nicht viel zu tun mit der ED zwischen den ursprünglichen Zeichenketten

    Formal: wenn $x = x_1 x_2$ und $y = y_1 y_2$, dann ist im Allgemeinen **nicht** $ED(x, y) = ED(x_1, y_1) + ED(x_2, y_2)$

Editierdistanz 9/9

- Lösungsidee 3: alternative rekursive Formel
  - Das Problem auf ein Problem "eins kleiner" zurückführen
  - ED("FERIEN", "VERIEN") = ED("FERIE", "VERIE") = 1

    Weil die Worte rechts mit dem selben Buchstaben aufhören
  - ED("SAUM", "RAUS") = ED("SAU", "RAU") + 1 = 2

    Weil eine optimale Folge ein replace am Ende macht

    Das gilt aber nicht immer, wenn sich die letzten beiden Buchstaben unterscheiden, zum Beispiel:

    2 = ED("RAUM", "GRAU") ≠ ED("RAU", "GRA") + 1 = 3
  - Mit einer etwas komplizierteren Formel kriegt man es aber hin, das sehen wir jetzt auf den nächsten Folien

Rekursive Formel 1/7

INS(1,U) INS(1,A) INS(1,S)
DOOF → UDOOF → AUDOOF → SAUDOOF
nicht monoton

INS(1,S) INS(2,A) INS(3,U)
DOOF → SDOOF → SADOOF → SAUDOOF
das ist monoton

UNI FREIBURG

- Monotonie
  - Seien **x** und **y** unsere beiden Zeichenketten
  - Seien $\boldsymbol{\sigma_1, \ldots, \sigma_k}$ eine Folge von k = ED(x, y) Operationen, für x → y, das heißt, um x in y zu überführen

    Wir nehmen im Folgenden nicht an, dass wir die Folge schon kennen, sondern nur, dass es so eine gibt
  - Eine Folge von Operationen heißt **monoton**, wenn die Position von σ_{i+1} ist ≥ die Position von σ_i, wobei = nur dann erlaubt ist, wenn beides "delete" Operationen sind

    Eine Folge von delete Operationen mit **gleichen** Positionen braucht man zum Beispiel bei ED("saudoof", "doof")

    SAUDOOF → AUDOOF → UDOOF → DOOF
    DELETE(1) DELETE(1) DELETE(1)

Rekursive Formel 2/7

- Hilfssatz zur Monotonie
  - **Lemma:** Für beliebige x und y mit $k = ED(x, y)$ gibt es eine monotone Folge von k Operationen für $x \rightarrow y$
  - Der Beweis ist Aufgabe 1 vom ÜB11
  - Beweisidee 1: für den Fall $k = 2$ (zwei Operationen), muss man sich im Prinzip nur alle neun Kombinationen der drei Arten von Operationen anschauen
  - Beweisidee 2: eine allgemeine nicht-monotone Folge lässt sich durch "Nachbartranspositionen" immer in eine monotone Folge derselben Länge überführen

    Nachbartransposition = zwei Operationen hintereinander in nicht-monotoner Reihenfolge werden "umgedreht"

Rekursive Formel 3/7

BEISPIELE FÜR DIE FÄLLE

FALL 1A: $\overset{x}{\text{GRAU}} \xrightarrow[\sigma_1]{} \overset{z}{\text{RAU}} \xrightarrow[\sigma_2]{} \overset{y}{\text{RAUM}}$

FALL 1B: $\overset{x}{\text{BÄUME}} \xrightarrow[\sigma_1]{} \overset{z}{\text{BAUME}} \xrightarrow[\sigma_2]{} \overset{y}{\text{BAUM}}$

FALL 1C: $\overset{x}{\text{DOOF}} \xrightarrow[\sigma_1,\sigma_2,\sigma_3]{} \overset{z}{\text{BLOEF}} \xrightarrow[\sigma_4]{} \overset{y}{\text{BLOED}}$

das ist also die, die am „weitesten rechts" was macht

- Fallunterscheidung
  - Wir betrachten die letzte Operation σ_k einer monotonen Folge

    $\sigma_1, \sigma_2, \sigma_3$ DOOF BLOEF σ_4: BLOEF → BLOED

    $\sigma_1, \ldots, \sigma_{k-1} : x \rightarrow z$ und $\sigma_k : z \rightarrow y$

    Seien $n = \overset{4}{|x|}$ und $m = \overset{5}{|y|}$ und $m' = \overset{5}{|z|}$

    σ_k: INSERT REPLACE DELETE

    Man beachte, dass $m' \in \{m - 1, m, m + 1\}$
  - **Fall 1:** σ_k macht etwas "ganz am Ende" von z :

    Fall 1a: σ_k = insert($m' + 1$, $y[m]$) [dann ist $m' = m - 1$]

    Fall 1b: σ_k = delete(m') [dann ist $m' = m + 1$]

    Fall 1c: σ_k = replace(m', $y[m]$) [dann ist $\underset{|z|}{m'} = \underset{|y|}{m}$]

    Wenn keines von den dreien der Fall ist, dann sind die letzten Zeichen von x und y (und z) gleich … siehe nächste Folie

Rekursive Formel 4/7

- Fallunterscheidung
  - Wir betrachten die letzte Operation σ_k

    $\sigma_1, \ldots, \sigma_{k-1} : x \rightarrow z$ und $\sigma_k : z \rightarrow y$

    Seien $n = |x|$ und $m = |y|$ und $m' = |z|$

    Man beachte, dass $m' \in \{m - 1, m, m + 1\}$
  - **Fall 2:** σ_k macht nichts "ganz am Ende" von z

    Dann $z[m'] = y[m]$ und $x[n] = z[m']$

    Damit $\sigma_1, \ldots, \sigma_k : x[1..n-1] \rightarrow y[1..m-1]$ und $x[n] = y[m]$

DOOFI $\longrightarrow$ BLOEDI

$\sigma_1, \sigma_2, \sigma_3, \sigma_4$

Rekursive Formel 5/7

y ist das, was man am Ende haben will

- Wir haben also einen dieser vier Fälle
  - Fall 1a (insert am Ende): $\sigma_1, \ldots, \sigma_{k-1} : x[1..n] \rightarrow y[1..m-1]$
  - Fall 1b (delete am Ende): $\sigma_1, \ldots, \sigma_{k-1} : x[1..n-1] \rightarrow y[1..m]$
  - Fall 1c (replace am Ende): $\sigma_1, \ldots, \sigma_{k-1} : x[1..n-1] \rightarrow y[1..m-1]$
  - Fall 2 (nichts am Ende): $\sigma_1, \ldots, \sigma_k : x[1..n-1] \rightarrow y[1..m-1]$

    Wichtig für die Formel auf der nächsten Seite: diese vier Fälle decken **alle** Möglichkeiten ab, andere gibt es nicht

Rekursive Formel 6/7

- Daraus folgt die folgende rekursive Formel
  - Für $|x| > 0$ und $|y| > 0$ ist $ED(x, y)$ das **Minimum** von
    - $ED(x[1..n], y[1..m-1]) + \overset{\text{INS}}{1}$
    - $ED(x[1..n-1], y[1..m]) + \overset{\text{DEL}}{1}$
    - $ED(x[1..n-1], y[1..m-1] + \overset{\text{REP}}{1}$ … falls $x[n] \neq y[m]$
    - $ED(x[1..n-1], y[1..m-1]$ … falls $x[n] = y[m]$

    das +1 steht für die jeweils letzte Operation

    Jeder der vier führt zu einer möglichen Folge, und wir wissen, dass die minimale Folge dabei ist, deswegen das **Minimum**
  - Für $|x| = 0$ ist $ED(x, y) = |y|$
  - Für $|y| = 0$ ist $ED(x, y) = |x|$

Rekursive Formel 7/7

ALIGNMENT

- Alternative Sichtweise
  - Visualisieren einer Folge von Operationen, indem man **x** und **y** mit geeigneten "Lücken" untereinander schreibt

	REP	REP		INS	REP	REP	INS		DEL	REP
x	**D**	**O**	**O**		**V**	**M**		**A**	**N**	**N**
y	**B**	**L**	**O**	**E**	**D**	**F**	**R**	**A**		**U**

8 OPERATIONEN

insert = oben leer, darunter ein Buchstabe

delete = unten leer, darüber ein Buchstabe

replace = zwei ungleiche Buchstaben übereinander

  - Wenn man da jetzt von links nach rechts durchgeht, hat man wieder genau eine monotone Folge
  - Die rekursive Formel unterscheidet die vier Möglichkeiten in der letzten Spalte (insert, delete, replace, nix)

Rekursive Implementierung 1/4

- Code
  - Mit der Formel von der Folie vorher können wir jetzt sehr leicht ein rekursives Programm schreiben
  - Es funktioniert auch! (immerhin)

    Aber es dauert unverhältnismäßig lange, selbst schon für relative kurze Zeichenketten

Rekursive Implementierung 3/4

- Laufzeit
  - Für die Laufzeit gilt folgende rekursive Formel

    $$T(n, m) = \underbrace{T(n-1, m)}_{\geq T(n-1, m-1)} + \underbrace{T(n, m-1)}_{\geq T(n-1, m-1)} + T(n-1, m-1) + \Theta(1)$$

  - Insbesondere:

    $$T(n, n) \geq 3 \cdot T(n-1, n-1) \geq 3 \cdot 3 \cdot T(n-2, n-2) \geq \ldots$$

  - Also:

    $$T(n, n) \geq 3^{n-1} \cdot T(1, 1) \geq 3^{n-1}$$

    Also **exponentielle** Laufzeit, wie auch schon bei der rekursiven Fibonacci-Berechnung

    Insbesondere: Zeichenketten um 1 länger

    → drei mal so lange

Rekursive Implementierung 4/4

- Problem
  - Das rekursive Programm berechnet die gleichen ED Werte **immer und immer wieder**

    Das hatten wir auch schon bei der Berechnung der Fibonacci Zahlen gesehen (in Vorlesung 1b)

z.B. ED(ab, cd)

| = ein rekursiver Aufruf

19 AUFRUFE

	ε	c	d
ε	III	IIII	I
a	IIII	III	I
b	I	I	I

diese Zelle entspricht ED(ε, cd)

← Basisfall

Basisfall

die drei ED(ab, c)

Literatur / Links

- Editierdistanz
  - In Wikipedia (Definition + Algorithmen)

    http://en.wikipedia.org/wiki/Levenshtein_distance

    http://de.wikipedia.org/wiki/Levenshtein-Distanz