Datenstrukturen und Algorithmen, SS 2024
Prof. Dr. P. Rossmanith
D. Cloerkes, M. Gehnen, D. Mock, N. Speetzen

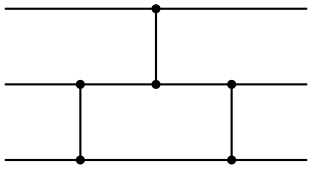
Theoretische Informatik | RWTH AACHEN UNIVERSITY

Abgabe: 31.05.2024

Übungsblatt 06

Aufgabe T19

Manche Sortieralgorithmen lassen sich mithilfe so genannter Sortiernetzwerke[1] implementieren. In einem solchen Netzwerk repräsentieren wir die Felder des zu sortierenden Arrays als Linien, die von links nach rechts gehen. Wir können jeweils zwei dieser Linien mit einer Operation verknüpfen, dem Vergleich-Swap-Modul. Das Vergleich-Swap-Modul vertauscht die Inhalte der beiden verbunden Linien genau dann, wenn ihre Reihenfolge falsch ist. Das einfachste Sortiernetzwerk für 3 Linien sähe dann wohl so aus:



(a) Finden Sie ein Sortiernetzwerk für 5 Linien. Lässt sich Ihr Sortiernetzwerk einfach für beliebige n konstruieren?

(b) Wir definieren die Tiefe eines Sortiernetzwerkes als die maximale Anzahl an Vergleich-Swap-Modulen auf irgendeinem Pfad des Sortiernetzwerkes. Geben Sie die Tiefes ihres Sortiernetzwerkes an.

(c) Welche Vorteile sehen Sie in Sortiernetzwerken?

Aufgabe T20

	Quicksort	Heapsort	Mergesort	Insertion-Sort	Straight-Radix	Radix-Exchange
In-place?						
Stabil?						
Laufzeit (worst-case)						
Laufzeit (Durchschnitt)						
Vergleichsbasiert?						

Beantworten Sie die Fragen für alle Sortierverfahren. Gehen Sie davon aus, dass ein Vergleich in konstanter Zeit durchgeführt wird und die Anzahl der zu sortierenden Elemente n beträgt. Für Laufzeiten tragen Sie eine Funktion $f(n)$ in die Tabelle ein, um eine Laufzeit von $O(f(n))$ auszudrücken.

[1] `https://www.youtube.com/watch?v=30WcPnvfiKE&t=44s`

Aufgabe T21

	Liste	Sortierte Liste	Array	Sortiertes Array	Binärer Suchbaum	AVL-Baum	Splay-Tree	Treap	Skip-List	Hashtabelle	Min-Heap
Randomisiert											
Einfügen											
Suchen											
Löschen											
Minimum											
Maximum											
Sort. Ausgeben											

Beantworten Sie die Fragen für alle Datenstrukturen bzw. geben Sie eine obere Schranke für den Aufwand an. Gehen Sie davon aus, dass die Anzahl der enthaltenden Elemente n beträgt. Für Laufzeiten tragen Sie eine Funktion $f(n)$ in die Tabelle ein, um eine Laufzeit von $O(f(n))$ auszudrücken. Bei randomisierten Datenstrukturen ist die erwartete Laufzeit gemeint. Für Splay-Bäume tragen Sie die amortisierten Laufzeiten ein.

Aufgabe T22

Entwickeln Sie eine Möglichkeit, die Adjazenzmatrix eines ungerichteten Graphens mit n Knoten in höchstens $n(n-1)/2$ Einträgen zu speichern, so dass es trotzdem in $O(1)$ Schritten möglich ist zu entscheiden, ob zwei Knoten i, j adjazent sind.

Aufgabe H19 (1 + 1 + 2 + 6 Punkte)

Gegeben seien zwei $n \times n$-Matrizen A und B, welche jeweils n^2 Buchstaben $A_{ij}, B_{ij} \in \{a, \ldots, z\}$ enthalten.

Ziel dieser Aufgabe ist ein schneller Algorithmus, der entscheiden kann, ob A und B genau die gleichen Zeilen (in möglicherweise anderer Reihenfolge) enthalten.

Beispiel: Hier lässt sich A durch Zeilenvertauschungen in B verwandeln

$$A = \begin{pmatrix} e & t & y & o & u & d \\ g & o & n & n & a & l \\ p & n & e & v & e & r \\ v & e & y & o & u & u \\ o & n & n & a & g & i \\ n & e & v & e & r & g \end{pmatrix} \qquad B = \begin{pmatrix} p & n & e & v & e & r \\ e & t & y & o & u & d \\ g & o & n & n & a & l \\ o & n & n & a & g & i \\ n & e & v & e & r & g \\ v & e & y & o & u & u \end{pmatrix}$$

(a) Ein Lösungsansatz ist es, die Matrizen zu sortieren und anschließend zu vergleichen. Wie lange dauert es, zwei Zeilen nach lexikographischer Ordnung zu vergleichen?

(b) Wie lange würde Heapsort benötigen, die Zeilen einer $n \times n$-Matrix lexikographisch zu sortieren?

(c) Jemand schlägt vor, zunächst jede Zeile in eine (sehr große) Zahl zu verwandeln (indem die Binärkodierungen der Buchstaben konkateniert werden) und diese n Zahlen dann in $O(n \log n)$ Schritten zu sortieren. Wo steckt der Fehler in dieser Idee?

(d) Wie kann man die Aufgabe in $O(n^2)$ Schritten lösen? Beschreiben Sie Ihren Algorithmus verständlich und begründen Sie seine Korrektheit und die Laufzeitschranke.

Hinweis: Wenn Sie Teil (d) ohne Sortieren lösen, dann müssen Sie (a)–(c) nicht für die volle Punktzahl abgeben.

Aufgabe H20 (3 + 4 + 3 Punkte)

Gegeben sei ein Integer-Array der Größe m, welches jede Zahl von 1 bis n *genau einmal* enthält, wobei $n \leq m$. Die restlichen $m - n$ Positionen sind mit 0 gefüllt.

(a) Implementieren Sie einen Algorithmus, der ein gegebenes solches Array in $O(m)$ Schritten sortiert. Achten Sie insbesondere darauf, dass ein Fehler gemeldet wird, wenn das Array nicht dem gewünschten Format entspricht. Geben Sie eine kurze Beschreibung an. Implementieren Sie Ihren Algorithmus in Python/Java/C++/Rust.

(b) Wie kann man die obige Aufgabe *in place* lösen, also nur mit konstant viel Speicher zusätzlich zur Eingabe? Beschreiben Sie wieder Ihren Algorithmus und implementieren Sie diesen in Python/Java/C++/Rust.

(c) Geben Sie die gemessene Laufzeit Ihrer Programme (ohne die Zeit um die Datei einzulesen) in Sekunden an für die Listen, die Sie als Archiv `arrays.zip` auf unserer Website `https://tcs.rwth-aachen.de/lehre/DA/SS2024/arrays.zip` finden können.[2] Die Listen sind einfache ASCII Dateien `arrayX.txt`, in der jede Zeile eine Zahl enthält.

Aufgabe H21 (10 Punkte)

Implementieren Sie ein Programm, welches folgende Funktion berechnen kann. Dabei sollen Sie aber sehr sparsam vorgehen: Sie dürfen ein vorinitialisiertes Array mit 17 Zahlen verwenden, eine XOR-Operation, eine MOD-Operation und einen lesenden Zugriff auf Ihr Array.

n	9689	9941	11213	19937	21701	23209	44497	86243	110503	132049
$f(n)$	3	5	7	13	17	19	23	37	47	59

n	216091	756839	859433	1257787	1398269	2976221	3021377
$f(n)$	61	67	71	79	89	101	103

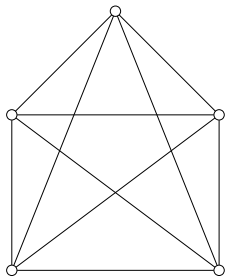
Wenn Ihr Programm eine hier nicht aufgeführte Zahl als Eingabe bekommt, darf die Ausgabe beliebig sein. Das Programm darf aber auch dann nicht „abstürzen“.

Aufgabe H22 (10 Punkte)

Ein *Dreieck*[3] in einem ungerichteten Graphen ist ein Untergraph, der aus drei Knoten besteht, welche paarweise miteinander verbunden sind.

Entwerfen Sie einen Algorithmus, der für einen Graphen mit n Knoten in $O(n^3)$ Schritten herausfinden kann, ob er ein Dreieck enthält.

Wie viele Dreiecke gibt es im rechten Graphen?



[2] In Moodle kann man nur Dateien hochladen, die nur höchstens 250 MB groß sind.

[3] `https://www.youtube.com/watch?v=dQw4w9WgXcQ`