Albert-Ludwigs-Universität
Institut für Informatik
Prof. Dr. F. Kuhn
P. Bamberger
P. Schneider

Algorithmen und Datenstrukturen
Sommersemester 2020
Musterlösung Übungsblatt 1

Abgabe: Mittwoch, 20.05.2020, 16:00 Uhr.

Aufgabe 1: Anmeldung *(5 Punkte)*

Melden Sie sich beim Kurssystem Daphne an. Den Link dazu finden Sie auch auf der Kurs-Website. Achten Sie darauf, dass Ihre Daten korrekt sind, insbesondere dass Sie unter der angegebenen E-Mail Adresse auch erreichbar sind. Führen Sie ein `checkout` auf Ihr SVN-Repository durch.[1]

Aufgabe 2: Quicksort *(5 Punkte)*

Implementieren Sie den in der Vorlesung erklärten Algorithmus *QuickSort*. Verwenden Sie dazu die auf der Webseite verlinkte Design-Vorlage `QuickSort.py`. Schreiben Sie je einen Unit Test für die `quicksort_divide` und die `quicksort_recursive` Methode. Die Unit Tests sollten grundsätzlich mindestens ein nicht-triviales Beispiel überprüfen. Wenn es kritische Grenzfälle gibt, die sich leicht nachprüfen lassen (z.B. Verhalten einer Methode bei leerem Eingabefeld), sollen Sie dies tun.

Musterlösung

Siehe `Quicksort.py` im public Ordner oder auf der Website.

Aufgabe 3: Zeitmessungen *(5 Punkte)*

Messen Sie die Laufzeit der Sortieralgorithmen *SelectionSort* und *MergeSort*[2], sowie von Ihrer *QuickSort*-Implementierung für verschiedene Eingabegrössen n (Anzahl Sortierschlüssel). Testen Sie bei *QuickSort zwei Varianten* der Pivot-Wahl: "Element an erster Position", "Element an zufälliger Position". Wiederholen Sie das Experiment für zwei unterschiedlich generierte Eingabearten. Einmal mit einem Array das mit *zufälligen* Integers gefüllt ist und einmal mit einem Array in welchem die Elemente paarweise verschieden und in *umgekehrter* Reihenfolge sortiert sind.
Stellen Sie die Laufzeit der *vier* Algorithmen jeweils mit den *zwei* Eingabearten für die Eingabegrößen $n \in \{100, 200, \ldots, 5000\}$ graphisch dar.[3] Vergleichen Sie die Laufzeiten anhand der erhaltenen Schaubilder und diskutieren Sie Ihre Ergebnisse kurz in Ihren `erfahrungen.txt` (siehe Aufgabe 4).

[1] Ihr SVN-Repository wird bei der ersten Anmeldung bei Daphne automatisch angelegt. Die URL ist: `https://daphne.informatik.uni-freiburg.de/ss2020/AlgoDat/svn/ihr-rz-account-name`

[2] Diese liegen im public repository `https://daphne.informatik.uni-freiburg.de/ss2020/AlgoDat/svn/public`

[3] Die Unterschiede der Laufzeiten der Algorithmen werden am deutlichsten wenn diese gemeinsam in einem einzigen Schaubild aufgetragen werden mit n auf der x-Achse und der Laufzeit $T(n)$ auf einer *logarithmischen* y-Achse.

Musterlösung

Abbildungen 1 und 2 stellen die Laufzeiten jeweils mit unterschiedlicher y-Skala graphisch dar. Wir machen folgende Beobachtungen (Aufzählung sicherlich nicht erschöpfend):

- *SelectionSort* hat ganz klar einen superlinearen Trend (genauer: einen quadratischen).
- *SelectionSort* ist für randomisierte Eingabe etwas schneller als für die umgekehrt sortierte Eingabe (das liegt unserer Einschätzung nach daran, dass bei der umgekehrt sortierten Liste die Zeile in der If-Abfrage *immer* ausgeführt wird, was bei der zufälligen Eingabe nicht immer der Fall ist).
- Einen superlinearen (quadratischen) Trend hat auch *QuickSort* für deterministische Pivotwahl (erstes Element) und umgekehrt sortierter Liste, welches sogar die langsamste Laufzeit aller getesteten Sortiervarianten hat (der Fall ist ein worst case für *QuickSort*).
- Dahingegen ist *Quicksort* für alle anderen Varianten in denen eine Randomisierung des Pivots oder der Eingabe stattfindet, für große Eingaben wesentlich schneller als die oberen Varianten (genauer: $\Theta(n \log n)$ *"mit hoher Wahrscheinlichkeit"*, siehe Vorlesung Woche 2). Gut zu erkennen an Abbildung 2.
- *MergeSort* hat auch eine wesentlich schnellere Laufzeit (als die Algorithmen mit quadratischem Trend) für *alle* getesteten Eingaben (genauer: $\Theta(n \log n)$ *garantiert*, siehe Vorlesung Woche 2).

Aufgabe 4: Abgabe *(5 Punkte)*

Committen Sie Ihren Code (inkl. Tests) und die acht Schaubilder in das SVN, in einen eigenen Unterordner `uebungsblatt-01`. Gehen Sie dabei so vor, wie in der Vorlesung vorgeführt. Stellen Sie sicher, dass auf Jenkins alles (inkl. Style Check und Unit Tests) fehlerfrei durchläuft.
Committen Sie in diesem Unterordner ausserdem eine Textdatei `erfahrungen.txt`. Beschreiben Sie dort in ein paar Sätzen Ihre Erfahrungen mit diesem Übungsblatt und den Vorlesungen dazu. Insbesondere: Wie lange haben Sie ungefähr gebraucht? An welchen Stellen gab es Probleme und wieviel Zeit hat Sie das gekostet?

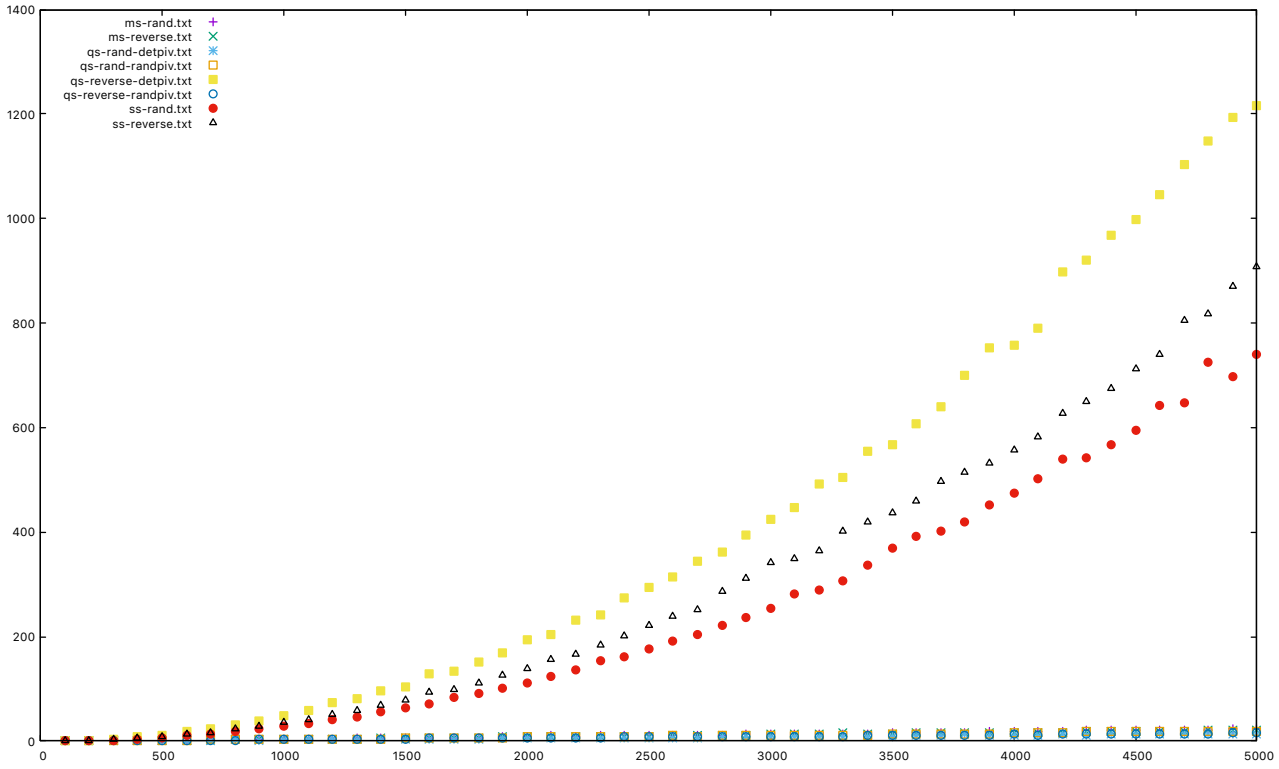

Figure 1: Der erste Plot gibt die Laufzeiten von allen geforderten Varianten von Sortieralgorithmen für die geforderten Eingaben über der Eingabegröße n an.

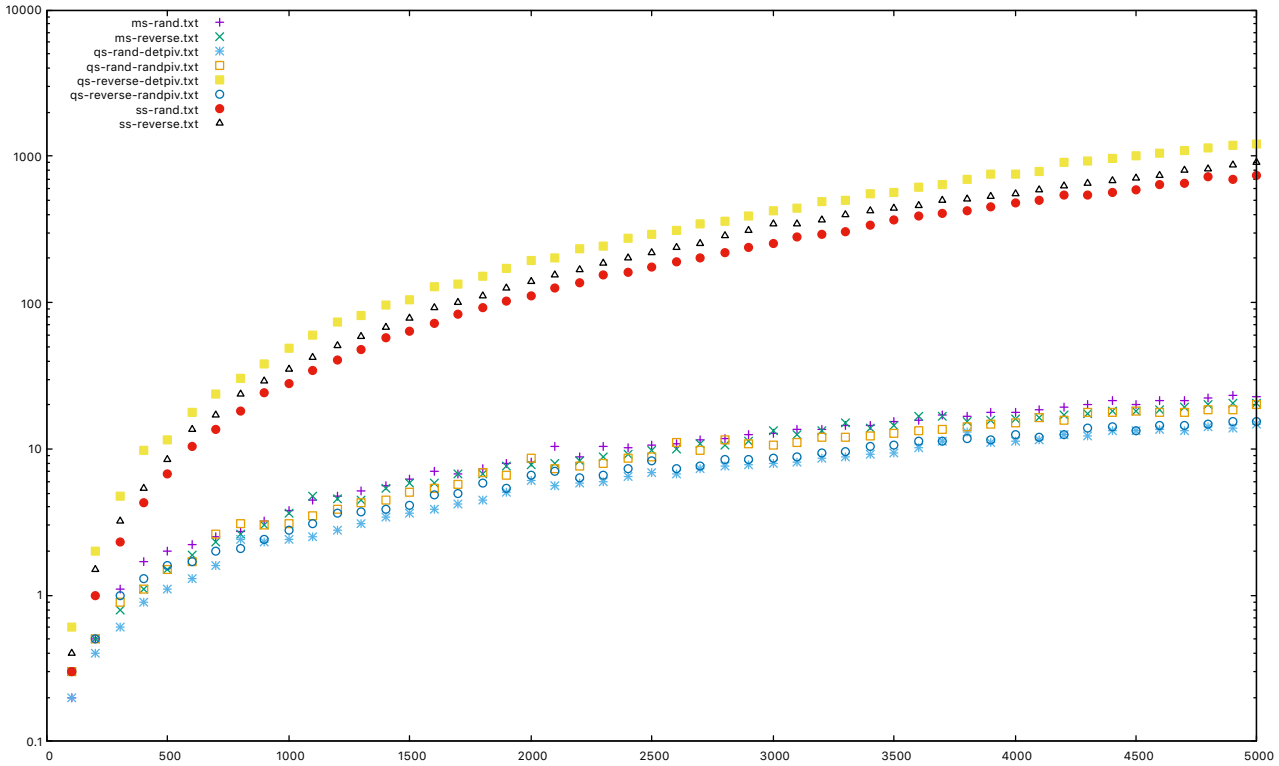

Figure 2: Der zweite Plot gibt die Laufzeiten von allen geforderten Varianten von Sortieralgorithmen für die geforderten Eingaben über der Eingabegröße n an. Die y-Skala ist logarithmisch.