Bucketsort/Radixsort

In diesem Exkurs wollen wir Sortierverfahren betrachten, die es in vielen Fällen erlauben, n Werte in Linearzeit zu sortieren. Dies steht zunächst scheinbar im Widerspruch zur informationstheoretischen unteren Schranke von $\Omega(n \log n)$ für Sortierprobleme, liegt aber an einer kleinen (in der Praxis meist irrelevanten) Einschränkung auf bestimmte Werte.
Stellen Sie sich zunächst vor, Sie wüssten genau, dass als Schlüssel in Ihrem Sortierproblem nur die Zahlen $1 \cdots 10$ auftreten und egal ist, in welcher Reihenfolge Werte mit gleichem Schlüssel angeordnet werden.
In diesem Fall kann das Sortierproblem gelöst werden, indem man 10 Eimer "Buckets" (z.B. realisiert durch verkettete Listen) aufstellt und jedes Objekt in den passenden Eimer "wirft". Danach holt man erst alle Elemente aus dem ersten Eimer, dann alle Elemente aus dem zweiten usw. Dies kann offensichtlich in linearer Zeit in der Zahl der zu sortieren Elemente geschehen.
Aufgabe 1: Nehmen Sie an, die Eimer sind als Schlangen realisiert. Zeigen Sie, dass Bucket-Sort stabil ist, d.h. dass Elemente mit gleichem Schlüssel in der Reihenfolge in der sie eingelesen wurden in der Ausgabe auftauchen.

Dieses Verfahren wird gleich mit einem allgemeineren Schema gekoppelt werden, das wir zunächst unabhängig davon vorstellen.
Viele von Ihnen werden z.B. in einer Datenbank oder in einem Tabellenkalkulationsprogramm schon einmal Namen, bei denen Vorname und Name getrennt gegeben waren oder Daten, bei denen Monat und Tag in verschiedenen Feldern gestanden haben, sortiert haben. Das Vorgehen ist normalerweise das, dass man zunächst nach dem "unwichtigeren" (Tag,Vorname) und dann nach dem "wichtigeren" (Monat,Name) sortiert.
Aufgabe 2: Überlegen Sie sich, dass das nur bei stabilen Sortierverfahren funktioniert.

Nehnmen wir nun an, wir hätten ganze Zahlen mit k Stellen gegeben. Auf diese können wir nun dasselbe Schema mehrfach anwenden. Wir sortieren sie mit einem stabilen Verfahren zunächst nach der letzten Stelle, dann nach der vorletzten usw. Man kann sich mit Induktion leicht überlegen, dass die 1te bis $i-1$te Operation eine Sortierung nach den hinteren $i-1$ Stellen bildet, so dass, wie gerade überlegt, eine stabile Sortierung dieser vorsortierten Daten nach der it-hintersten Stelle eine Sortierung nach den letzten i Stellen erzeugt.
Da die $i$te Stelle aber (bei Dezimalschreibweise) nur die Werte $0 \cdots 9$ annehmen kann, können wir diese Sortierung in Linearzeit mit Bucketsort vornehmen.
k-stellige Zahlen (also nicht beliebige Zahlen !) können wir also in k Runden, d.h. in Zeit $\mathcal{O}(km)$ sortieren. Analog kann man Sortierverfahren für andere "Stellenwertsysteme" und andere stabile Sortieralgorithmen konstruieren.
Aufgabe 3 Überlegen Sie, ob Sie IEEE-Floating-Point-Zahlen in lineaer Zeit sortieren können.