

## **Lawinensortierung – von Dr. Dieter Porth**

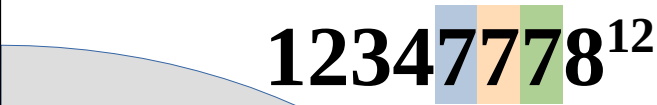

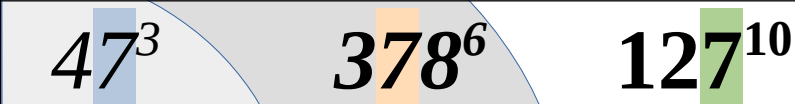

In einer Behörde hatte die Zentrale Klimaanlage den Corona-Virus weit verbreitet. Der Amtsarzt soll nun eine Untersuchung durchführen. Damit die älteren Hochrisiko-Beamten möglichst schnell nach Hause können, soll der Hausmeister alle Beamten dem Alter nach absteigend sortiert in einer Schlange aufstellen. Da auf Grund seiner Erfahrungen weist der Amtsarzt dem Hausmeister darauf hin, dass er mit dem Sortieren anfangen solle und dass er mit Nachzüglern rechnen müsse. Wegen dieser Nachzügler wählt der programierbegeisterte Hausmeister sein Lawinensortier-Verfahren. Das ist besser als das ähnliche Mergesort, weil es nicht nur stabil und gerecht sortiert, sondern auch Nachzügler-robust und Vorsortierungen belohnt.

Da Beamte das Schlange stehen schon in der Kantine gelernt haben, findet der Hausmeister an der Tür zur Amtsarztstube schon eine langsam länger werdende Schlange wartender vor. Beginnend von der Tür bestimmt er den ersten Lauf, also die erste Teilschlange von Beamten der richtigen Sortierung von alt nach jung. Dieser Run ist seine erste Teilschlange nullter Ordnung. Bei ersten Bruc – also wenn der Person in der Schlange älter als sein Vorgänger ist, beginnt er den zweiten Run (Teilschlange) nullter Ordnung zu bestimmen. Sobald der Hausmeister zwei Run gleicher Ordnung hat, fügt (=mergt) er diese beiden Schlangen zum Run (Teilschlange) erster Ordnung zusammen. Die Ordnung definiert also über die Formel  $2^{\text{Ordnung}}$  die maximale Zahl der Runs im gemergten Run. Da er jetzt einen Run erster Ordnung hat, beginnt er frisch bei der Restschlange und sucht eine dritte und vierte Teilschlange nullter Ordnung. Diese fügt er dann zum zweiten Run erster Ordnung zusammen. Jetzt kennt er zwei Schlangen erster Ordnung und fügt sie zum Run (Teilschlange) zweiter Ordnung zusammen. In der Zwischenzeit haben sich weitere fünfzig nachzügeln Beamte ans Ende der Schlange angestellt. Der Hausmeister beginnt sein zweite Teilschlange zweiter Ordnung zusammenzustellen, indem er nach dem gleichen Muster wie zuvor die Teilschlangen Nr. 5,6,7 und 8 zum Run zweiter Ordnung zusammenfügt. Seine beiden Schlangen zweiter Ordnung fügt (mergt) er dann zu einer Schlange dritter Ordnung zusammen. Wie einer Lawine, die klein startet und sich immer mehr Schnee einverleibt, so wächst bei der Lawinensortierung (avalanche-sort) des Hausmeisters die Ordnung der Teilschlange und die bis dahin einverleibten Runs. Als unser Hausmeister gerade zum Abschluss seiner Sortierung seine beiden Schlangen achter Ordnung zu einer Schlange neunter Ordnung zusammengefügt hat, kommt noch ein letzter Nachzügler, der sich in der Behörde verlaufen hatte. Die Mathe-Freak können ja mal ausrechnen, wie viele Menschen mindestens in der sortierten Schlange gestanden haben. [17,18,19, 128, 129, 130, 256, 257 oder 258?].

Die Zusammenfassung. Wie der Mergesort sortiert die Lawinensortierung stabil. Im Worstcase sind die Sortierkennzahlen den Lawinensorts mit denen des Mergesorts vergleichbar. Im Gegensatz zum Quicksort oder Mergesort wird hier die Vorsortierung ausgenutzt. In Abgrenzung zu den beiden klassischen Sortierverfahren ist die Lawinensortierung robust gegen Nachzügler. Der Lawinensortierung ist Nachzügler-robust, weil für das Sortieren ein Datensatz seinen Nachfolger kennen muss. In verketteten Listen muss man also nur die Verlinkung mit den Nachfolgern vertauschen, um eine Sortierung herzustellen. Bei klassischen Quicksort muss dagegen jeder Datensatz – zum Beispiel über den Array-Index - direkt angesprochen werden können, um die Elemente wirklich per ‚swap‘ vertauschen zu können. Quicksort ist wegen der direkten Datenansprache eher für Arrays geeignet, während die Lawinensortierung eher für Listen ideal ist.

Abbildung:

8 Einträge in 6 Runs (=sortierte Teillisten) mit Indexnummer bei Sortierzwischenständen und mit Kennzeichnung, wie der Sortiererfolg mit Zunahme der Ordnung der Runs von links nach rechts lawinenartig anwächst.

Runordnung - Anzahl Runs	Anzahl Runs 0. Ordnung (max)	Snapshot der Teillisten in der Datenliste
3 – 1	$2^3=8$	 12347778 <sup>12</sup>
2 – 2	$2^2=4$	 34778 <sup>7</sup> 127 <sup>11</sup>
1 – 3	$2^1=2$	 47 <sup>3</sup> 378 <sup>6</sup> 127 <sup>10</sup>
0 – 6	$2^0=1$	 7 <sup>1</sup> 4 <sup>2</sup> 38 <sup>5</sup> 7 <sup>4</sup> 17 <sup>8</sup> 2 <sup>9</sup>

## Kurzfassung zur Lawinensortierung (abstract) "Optimale Sortieren mit avalanchesort"

Quicksort und Mergesort sind die klassischen Sortierverfahren mit guten Sortierverhalten. Sie sind suboptimal, weil sie Vorsortierungen nicht ausnutzen. Wenn Daten als einfach verkettete Liste vorliegen, dann kann man mit der Lawinensortierung (avalancheSort) Vorsortierungen ausnutzen. Die Lawinensortierung (avalancheSort) verfolgt die Strategie, rekursive aufsteigend sortierte Teilfolgen (=Runs) zu bestimmen und Runs gleicher Ordnung zu verschmelzen (mergen). Ein Run nullter Ordnung meint dabei eine sortierte Teilfolge in den originalen unsortierten Daten. Ein Run erster Ordnung meint eine sortierte Teilfolge, die aus zwei Runs nullter Ordnung gebildet wurde. Für einen Run zweiter Ordnung wurden aus zwei Runs erster Ordnung gemergt, die ihrerseits jeweils aus maximal zwei Runs nullter Ordnung gemergt wurden. Da die Ordnung je nach Sortiervolumen anwachsen kann, liegt hier eine aufsteigende bzw. eine angepasst aufsteigende Rekursion vor. Im Worst-Case wird der Lawinensort vergleichbar mit einem Mergesort vergleichbar. Im Worstcase, also bei umgekehrter Sortierung, besteht jeder Run nullter Ordnung nur aus einem Element. Im Bestcase, also wenn die zu sortierenden Daten schon sortiert sind, werden nur  $n-1$  Vergleiche benötigt.

Anmerkung: Wenn die Sortiermenge statisch ist, und wenn jedes Element sowohl seine Vorgänger wie Nachfolger kennt, könnte man den avalancheSort optimieren, indem man bei einelementigen Runs einen run umgekehrter Reihung nutzt. Bei dieser Optimierung müsste eine Merge-Methode entwickeln, die die vier Fälle von Merge-Variationen händeln kann. In diesem Fall wäre der Avalanche-Sort vermutlich immer besser als ein Mergesort, weil jeder Run immer aus mindestens zwei Elementen besteht und weil Vorsortierung und Anti-Vorsortierungen genutzt werden können.