Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

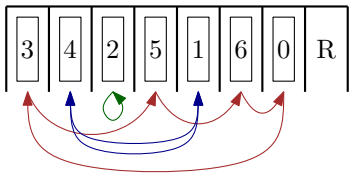
Departement Informatik
Markus Püschel
Peter Widmayer
Thomas Tschager
Tobias Pröger
Tomáš Gavenčiak

3. November 2016

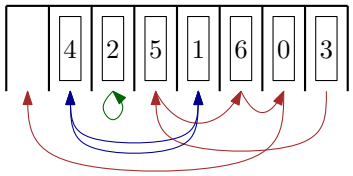
Datenstrukturen & Algorithmen Lösungen zu Blatt P6 HS 16

Lösung P6.1 *Chinese wall parking.*

Für jedes Auto zeichnen wir einen Pfeil zu der Position, an der es abgestellt werden muss. Für das Beispiel auf dem Aufgabenblatt erhalten wir die folgende Abbildung:



Betrachten wir einen einzelnen Zyklus der Länge $l \geq 2$ (zum Beispiel den Zyklus $3 \to 5 \to 6 \to 0 \to 3$ der Länge 4). Wir können ein einziges, beliebiges Auto auf den Reserveplatz (R) abstellen:



Nun haben wir den Zyklus aufgebrochen und können alle Autos auf dem Zyklus direkt an die richtige Stelle bewegen (für das obige Beispiel in der folgenden Reihenfolge: $0, 6, 5, 3$). Dazu benötigen wir $l + 1$ Bewegungen für diesen Zyklus.

Wir machen das für jeden Zyklus der Permutation. Natürlich benötigen wir keine Bewegungen für Zyklen der Länge 1 (da das Auto bei einem solchen Zyklus bereits auf der richtigen Position ist; z.B. Auto 2 im obigen Beispiel).

Unser Algorithmus muss also alle Zyklen der Permutation und deren Längen finden. Die Zyklen der Länge 1 werden ignoriert. Die Summe der Längen der übrigen Zyklen muss jeweils um 1 erhöht werden.

Auf der Webseite zur Vorlesung finden Sie eine Lösung mit Laufzeit in $\mathcal{O}(n)$, welche die Länge der Zyklen berechnet. Die Lösung enthält weitere Kommentare zur Implementierung.

Daten

`judge1` $n = 10\,000$, eine sortierte Sequenz.

`judge2` $n = 10\,000$, Permutation mit einem einzigen Zyklus.

`judge3` $n = 10\,000\,000$, zufällige Permutation.

Hinweise zu den Abgaben. Einige abgegebene Lösungen verwenden Heuristiken um die Anzahl der benötigten Bewegungen zu erraten. Zum Beispiel wurde einfach $n - k + 1$ ausgegeben, wobei k die Anzahl der Autos ist, die bereits in der richtigen Position sind. Das funktioniert nur, wenn alle anderen Autos zu einem einzigen Zyklus gehören. Andere Lösungen wollten diesen Ansatz verbessern, indem zusätzlich die Paare von Autos gezählt wurden, die nur vertauscht werden müssen (wie beispielsweise Auto 4 und Auto 1 im Beispiel). Auch das reicht noch nicht, um die Anzahl korrekt zu berechnen. Andere Lösungen versuchten Teilbarkeit oder andere Eigenschaften zu verwenden, die aber im Allgemeinen auch nicht funktionieren.

Es gibt jedoch einen ähnlichen Ansatz, der funktioniert: Beim Aufsummieren der Längen der Zyklen $l + 1$ in der vorgeschlagenen Lösung zählen wir eigentlich die Anzahl der Autos, die bewegt werden müssen plus die Anzahl der Zyklen, oder $n - k + c$, wobei k die Anzahl der Autos ist, die bereits an der richtigen Position sind, und c die Anzahl der Zyklen ist. Zyklen zählen ist jedoch im Wesentlichen dieselbe Aufgabe wie deren Länge zu zählen, d.h. die Lösung wäre dann in etwa dieselbe wie die vorgeschlagene Lösung.

Programme, welche die Zyklen rekursiv durchlaufen, stürzen mit einem `RUNTIME ERROR` ab, falls die Länge des Zyklus mehr als etwa 1000 beträgt. Eine nicht-rekursive Lösung ist also von Vorteil.

Mathematische Anmerkung. In einer zufälligen Permutation von n Autos ist die *erwartete* (mittlere) Anzahl an Autos, die an der richtigen Position stehen (überraschenderweise) $k = 1$ und die erwartete Anzahl an Zyklen der Länge 2 ist $c = \log n - 1$. Die erwartete Anzahl an Bewegungen ist $n - 2 + \log n$. Besuchen Sie die Wikipedia-Webseite zu zufälligen Permutationen[1] für weitere Details.

[1] `https://en.wikipedia.org/wiki/Random_permutation`