2. Dezember 2013

HAW Hamburg

Dokumentation

Zu Aufgabenblatt 09 aus der Vorlesungsreihe „Algorithmen und Datenstrukturen“

Dokumentation

Zu Aufgabenblatt 09 aus der Vorlesungsreihe „Algorithmen und Datenstrukturen“

# ÜbungsAufgabe 9.1

*Dumm gelaufen: Sie arbeiten immer noch als Programmierer bei GG Investment. Da die Zeitreihen länger werden, muss der Algorithmus weiter verbessert werden.  
Ihre Aufgabe ist es, einen verbesserten Algorithmus zu entwickeln, der die Zeitpunkte i und j für ein optimales Investment findet bzw. meldet, wenn kein Investment sinnvoll wäre. Der Zeitbedarf Ihres Algorithmus soll linear sein, d.h. in O(n) liegen.*

## Teilaufgabe 2

*Angenommen Sie können fp(k1,...,kn) berechnen. Wie berechnet man damit das optimale Investment f(k1,...,kn), (d.h. ohne Einschränkungen an p)?*

Wir betrachten als erstes das optimale Investment mit der Einschränkung durch p.  
Dann betrachten wir das optimale Investment mit der umgedrehten Liste (also f(kn,...,k1)).  
Danach werden die beiden Ergebnisse miteinander verglichen und das Maximum von beidem ist das Ergebnis.

## Teilaufgabe 3

*Arbeiten Sie die Details dieses Ansatzes aus! Geben Sie insbesondere an, welche Hilfsinformationen Sie erzeugen und wie Sie diese einsetzen.*

Es wird über die Liste der Zeitpunkte iteriert, dabei werden 5 Hilfsinformationen gespeichert. Bei diesen handelt es sich um:  
bestMinId – Index des Einkaufswertes des bisherigen optimalen Investments  
bestMaxId – Index des Verkaufswertes des bisherigen optimalen Investments  
bestRange – Gewinnspanne des bisherigen des bisherigen optimalen Investments. Dient nur des einfacheren Vergleichs.  
smallest – bisher kleinster gefundener Wert  
smallestId – Index des bisher kleinsten gefundenen Wertes

Beim iterieren über die Liste vergleichen wir immer unser aktuelles Element mit dem bisher kleinsten gefunden und berechnen die Gewinnspanne des Investment. Ist sie besser als unser bisheriges Optimum wird sie gespeichert. Ist sie dies nicht iterieren wir weiter nachdem wir überprüft haben ob das Element kleiner ist als unser bisher kleinstes. Ist dem so, so wird es weg gespeichert. So steht am Ende der Iteration die größte Gewinnspanne sowie die Werte die sie aufspannen in den Hilfsvariablen.

## Teilaufgabe 4

*Erläutern Sie, warum die funktionsweise korrekt ist:  
a) Wenn es ein Optimum gibt, dann wird auch eines gefunden.  
b) Wenn es kein Optimum gibt, dann wird ein Fehler gemeldet.  
c) Und das Paar (i,j), das gefunden wird, ist auch ein Optimum.  
d) Wenn ein Fehler gemeldet wird, dann gibt es kein Optimum.*

a) Falls es eins gibt, dann wird durch das iterieren irgendwann der optimale Verkaufswert Gegenstand unserer Betrachtung sein. Ist dem so, dann steht im smallest-Hilfswert der bisher niedrigste betrachtete Wert der Liste. Es kann keine größere Gewinnspanne gegeben haben und so wird die alte überspeichert. In der fortlaufenden Betrachtung wird keine Spanne mehr besser sein als unser Optimum und so bleibt es in den Variablen stehen und wird am Ende ausgegeben

b) Die Hilfsvariablen werden mit Defaultwerten initialisiert. Für die Indizes -1 für die Werte plus und minus unendlich. Gibt es kein Optimum werden diese Defaultwerte niemals überschrieben bzw. stehen immer noch negative Werte in der Gewinnspanne. Diese können dann als Fehler interpretiert werden.

c) Wenn die betrachtete Spanne ein Optimum, in unserem Fall ein in der Iteration bisheriges, ist so speichern wir die Werte und außerdem die Indizes der Werte.

d) Falls ein Fehler gemeldet wird, dann gibt es zu keinem Moment ein Investment, dass eine positive Gewinnspanne hat.

## Teilaufgabe 5

*Weisen Sie nach, dass die Laufzeit T(n) tatsächlich in O(n) liegt.*

In der Implementation gibt es keine Rekursion, daher müssen diese hier nicht berücksichtigt werden.  
Es gibt allerdings eine Schleife, die einmal über die Investmentliste iteriert. Ansonsten werden nur konstant viele Schritte durchgeführt.  
Aufgrund der einen Schleife liegt die Laufzeit in O(n).

## Teilaufgabe 6

*Implementieren Sie Ihren Algorithmus. Testen Sie ihn gegen Ihre alte Implementation. Werden die gleichen Resultate gefunden?*

Ja, werden es!

**public** **static** ArrayList<Integer> ultimateInvest(List<Integer> investments) {

**int** bestMinId = -1;

**int** bestMaxId = -1;

**int** bestRange = Integer.*MIN\_VALUE*;

**int** smallest = Integer.*MAX\_VALUE*;

**int** smallestId = -1;

**for**(**int** i = 0; i < investments.size(); i++) {

**if**(investments.get(i) - smallest > bestRange) {

bestMinId = smallestId;

bestMaxId = i;

bestRange = investments.get(i) - smallest;

}

**if**(smallest >= investments.get(i)) {

smallest = investments.get(i);

smallestId = i;

}

}

**return** **new** ArrayList<Integer>(Arrays.*asList*(bestMinId, bestMaxId, bestRange));

}