What's in YOUR wallet?

Sophie Hermann

20. Jänner 2017

Wie kann man einen gegebenen Betrag $n > 0$ mit den einem zur Verfügung stehenden Münzen darstellen?

$\rightarrow$ Unterscheidung zwischen Kassierer und Kunde

Literatur

- ”What’s in YOUR Wallet?” von Lara Pudwell und Eric Rowland

  $\rightarrow$ Was ist die erwartete Anzahl an Münzen in meiner Geldbörse?

- ”Combinatorics of the change-making problem” von Anna Adamaszek und Michael Adamaszek

  $\rightarrow$ Wann ist es sinnvoll, den Greedy-Algorithmus anzuwenden?

- Da wir uns für die Verteilung der Münzen interessieren, gehen wir von Preisen Modulo 100 Cent aus, also zwischen 0 und 99.

  $\rightarrow$ Zusätzliche Annahme:
  Der Cent-Anteil der Preise ist im Intervall 0 bis 99 gleichverteilt

- Welche Währung ist in Verwendung?

  $\rightarrow$ US-Dollar: $\$ = (1, 5, 10, 25)$

Perspektive des Kassierers

Wie bestimmt der Kassierer nun die minimale Anzahl an Münzen, die er zum Wechseln benötigt?

Dazu verwendet er (wohl meist unbewusst) einen sogenannten

"Greedy"-Algorithmus

Beispiel 1.1

Der Kassierer schuldet dem Kunden $n = 43$ Cent.

Er nimmt also:

- $1\times$ eine 25-Cent Münze ("Quarter"). Nun schuldet er nur noch 18 Cent.
- $1\times$ eine 10-Cent Münze ("Dime"), fehlen noch 8 Cent.
- $1\times$ eine 5-Cent Münze ("Nickel") und schlussendlich
- $3\times$ eine 1-Cent Münze ("Penny")

Der Greedy-Algorithmus führt beim Dollar immer zu der minimalen Anzahl an Münzen, die für das Retourgeld benötigt werden, vorausgesetzt man hat genügend von jeder Münze zur Verfügung!

Hat man z.B. keine 5-Cent Münze mehr zur Verfügung und möchte den Betrag $n = 40$ mit so wenig Münzen wie möglich begleichen, so erhält man mit dem Greedy-Algorithmus

$$\{25, 10, 1, 1, 1, 1, 1\} \text{ anstelle von } \{10, 10, 10, 10\}.$$

Definition 1.1

Eine Währung, bei der der Greedy-Algorithmus für jeden Betrag $n > 0$ erfolgreich angewendet werden kann, nennt man "orderly".

Ist dies für mindestens einen Betrag n nicht erfüllt, so sagt man die Währung ist "disorderly". Solche n bezeichnet man als "Counterexample".

Wann kann man nun den Greedy-Algorithmus erfolgreich bei einer Währung anwenden?

Für eine Währung $A' = (1)$, die nur aus einer einzigen Münze besteht, ist der Greedy-Algorithmus offensichtlich immer erfolgreich anwendbar.

Also ist A' "orderly".

Definition 1.2

Sei $A = (1, a_1, ..., a_k)$ eine Währung. Wir definieren uns für einen Betrag n:

$opt_A(n)$ sei die minimale Anzahl an Münzen, die für n benötigt wird und

$grd_A(n)$ sei die Anzahl an Münzen, die der Greedy-Algorithmus benötigt.

Satz 1.1 One-Point-Theorem

Sei $A' = (1, a_1, ..., a_k)$ "orderly", $a_{k+1} > a_k$ und $m := \lceil \frac{a_{k+1}}{a_k} \rceil$.

Dann ist $A = (1, a_1, ..., a_k, a_{k+1})$ genau dann "orderly", wenn

$$opt_A(ma_k) = grd_A(ma_k)$$

Perspektive des Kunden

Was ist die erwartete Anzahl an Münzen in der Geldbörse des Kunden?

$\rightarrow$ hängt von dem Muster ab, nach welchem die Münzen beim bezahlen ausgewählt werden:

- Coin-Keeper
- Minimalist Spender
- Big Spender
- Pennies-First Big Spender

Der Coin-Keeper

Ein Coin-Keeper bezahlt ausschließlich mit Scheinen:
Er sammelt somit eine große Anzahl an Münzen.

$\rightarrow$ Was ist nun deren Verteilung?

Da wir angenommen haben, dass die Preise zwischen 0 und 99 gleich wahrscheinlich sind, müssen wir nur noch die Münzen, die wir für jede der 100 Möglichkeiten brauchen, zusammenzählen:

Wir erhalten:

- 150 Quarters $\rightarrow$ 31.9%
- 80 Dimes $\rightarrow$ 17%
- 40 Nickels $\rightarrow$ 8.5%
- 200 Pennies $\rightarrow$ 42.6%

Markov-Kette

Der Zustand, in dem sich eine Geldbörse nach einer Transaktion befindet, hängt nur ab

- vom Zustand der Geldbörse vor der Transaktion,
- dem Preis und
- dem Muster, nachdem die Münzen ausgewählt werden.

$\rightarrow$ es liegt also eine Markov-Kette vor:

Unser System ist die Geldbörse und das zufällige Ereignis der Preis.

Markov-Kette

- Zustandsraum $S = \{s_1, s_2, ...\}$

  $\rightarrow$ z. B. $s_1 = \{1, 10, 10, 25\}$

- Anfangsverteilung $v = (v_1, ..., v_{|S|})$

- $|S| \times |S|$ Übergangsmatrix M

- vM^n Zustandswahrscheinlichkeit nach n Schritten

  $\rightarrow$ Damit ist das Lang-Zeit-Verhalten der Geldbörse gegeben durch vM^n für große n.

  $\rightarrow$ Existiert der Grenzwert $p = \lim_{n\rightarrow\infty} vM^n$?

Markov-Kette

- Existiert p, so muss gelten: $pM = p$, d.h. p ist ein linker Eigenvektor von M mit Eigenwert 1.

  Außerdem wissen wir: $p_1 + p_2 + ... + p_{|S|} = 1$

- Satz von Perron-Frobenius:
  Ist $M \geq 0$ irreduzibel und aperiodisch, so existiert ein positiver, einfacher EW von M mit einem positiven EV.

- Mit p können wir uns nun folgendes berechnen:

  $\rightarrow \sum_{i=1}^{|S|} p_i |s_i|$ ... erwartete Anzahl an Münzen in der Geldbörse
  $\rightarrow \sum_{i=1}^{|S|} p_i \sigma(s_i)$ ... erwartete Betrag in der Geldbörse

Minimalist-Spender

Geldbörse: n Cent Preis: c Cent

- minimale Partition von $n - c \bmod 100$ Cent nach der Transaktion
- Übergangswahrscheinlichkeiten immer $\frac{1}{100}$
- Erwartete Anzahl an Münzen: $\frac{1}{100}\sum_{i=1}^{100}|s_i| = 4.7$

Quarters:	Dimes:	Nickels:	Pennies:
1.5	0.8	0.4	2

- Erwarteter Betrag: $\frac{1}{100}\sum_{n=1}^{99} n = 49.5$

Big-Spender

- Mögliche Zustände sind jene mit $n \leq 99$ Cent

  $\rightarrow$ ergibt 6720 mögliche Zustände und $6720 \times 100 = 672000$ mögliche Transaktionen

  $\rightarrow$ Numerische Berechnung von p

- Erwartete Anzahl an Münzen: ~ 10.05

Quarters:	Dimes:	Nickels:	Pennies:
1.06	1.15	0.91	6.92

- Erwarteter Betrag: 49.5 Cent

Pennies-First Big Spender

- Berechnet den Preis Modulo 5
- Hat er vor der Transaktion ≤ 5 Pennies, hat er danach auch ≤ 5 Pennies

  $\rightarrow$ 1065 Zustände
- Erwartete Anzahl an Münzen: 5.74

Quarters:	Dimes:	Nickels:	Pennies:
1.12	1.27	1.35	2.00

Vergleich mit U.S.Mint 2014

	U.S. Mint	C-K	M-S	B-S	P-F-B-S
Quarters	11.9%	31.9%	31.9%	10.6%	19.51%
Dimes	17.4%	17%	17%	11.5%	22.13%
Nickels	9.1%	8.5%	8.5%	9.1%	23.52%
Pennies	61.6%	42.6%	42.6%	68.9%	34.84%

What's in YOUR wallet?

Vielen Dank für Ihre Aufmerksamkeit!

Satz 1.1 One-Point-Theorem

Sei $A' = (1, a_1, ..., a_k)$ "orderly", $a_{k+1} > a_k$ und $m := \lceil \frac{a_{k+1}}{a_k} \rceil$.

Dann ist $A = (1, a_1, ..., a_k, a_{k+1})$ genau dann "orderly", wenn

$$opt_A(ma_k) = grd_A(ma_k)$$