Das Heiratsproblem — Optimal Stopping

Das Modell

- Im Laufe unseres Lebens lernen wir mehrere potenzielle Partner kennen.
- Wir möchten den oder die „optimale(n)“ Partner(in) finden. Dazu und ordnen wir jedem Kandidaten einen Rang zu.
- Das Problem: Wir können nicht in die Zukunft sehen und wir können später auch nicht zu einem verlassenen Partner zurück.

— ♡ →

2	4	3	1	7	8	10	9	6	5

Eine präzise Formulierung der Problemstellung

Die folgenden Formulierungen sind willkürlich aus der Sicht einer Frau gewählt.

- Es gibt n Kandidaten und die Zahl n ist im voraus bekannt.
- Es wird genau ein Partner gewählt.
- Die Bewerber können, wenn sie alle zusammen gesehen werden, in aufsteigender Rangfolge eindeutig vom schlechtesten zum besten geordnet werden.
- Die Bewerber werden alle nacheinander in zufälliger Reihenfolge begutachtet, wobei jede Reihenfolge gleich wahrscheinlich ist.
- Nachdem man sich über einen potenzieller Partner ein Bild gemacht hat, trennt man sich oder man entscheidet sich für die Heirat und die Entscheidung ist unwiderruflich.
- Die Entscheidung, einen Bewerber anzunehmen oder abzulehnen, kann nur auf den relativen Rängen der bisher kennen gelernten Bewerbern beruhen.
- Das Ziel der allgemeinen Lösung ist es, mit der höchsten Wahrscheinlichkeit den besten Partner der gesamten Gruppe auszuwählen.

Die von uns untersuchte Strategie

- „Teste“ die ersten r potenziellen Partner, ordne ihnen einen eindeutigen Rang zu und verlasse sie danach.
- Wähle von den verbleibenden Bewerbern den ersten, der besser ist als jeder der ersten r. Wenn es keinen besseren gibt, wähle den letzten.

— ♡ →

2	4	3	1	7	8	10	9	6	5

$\underbrace{\qquad\qquad}_{r=3}$

Die zentrale Frage für den Rest dieses Dokuments

Wie gross muss r gewählt werden, damit die Chance auf den *besten* Partner (im obigen Beispiel ist es der mit dem Rang 10) maximal wird?

Aufgabe 1

Bestimme den Rang des Partners, den eine Partnersuchende nach der oben beschriebenen Strategie wählen muss.

(a)

4	1	2	3	6	9	8	5	10	7

$r = 3$

(b)

6	3	8	1	7	2	4	10	9	5

$r = 4$

(c)

3	1	2	4	5	7	8	6	9	10

$r = 2$

(d)

8	10	1	9	6	2	5	4	7	3

$r = 7$

(e)

4	2	3	1	7	8	6	9	10	5

$r = 1$

(f)

8	5	2	10	6	7	4	9	3	1

$r = 0$

Aufgabe 2

Beschreibe, warum die folgende Situation für die partnersuchende Person ungünstig ist.

2	1	3	4	5	7	10	9	8	6

$r = 3$

Aufgabe 3

Beschreibe, warum die folgende Situation für die partnersuchende Person ungünstig ist.

4	10	2	7	6	9	3	8	5	1

$r = 3$

Aufgabe 4

Die folgenden Tabelle enthält *alle* möglichen Reihenfolgen von 4 Kandidaten (Rängen).

(a) Warum gibt es 24 Reihenfolgen?

(b) Vervollständige die Tabelle, indem du jeweils den Rang des Kandidaten bestimmst, den wir mit der oben beschriebenen Stragie, nach $r = 0, 1, 2, 3$ begutachteten Kandidaten wählen müssen. Zwei Zeilen sind bereits ausgefüllt.

(c) Für welches r erhält man am häufigsten den Rang 4?

	bester nach $r =$					bester nach $r =$			
Reihenfolge	0	1	2	3	Reihenfolge	0	1	2	3
1, 2, 3, 4	1	2	3	4	3, 1, 2, 4	3	4	4	4
1, 2, 4, 3					3, 1, 4, 2				
1, 3, 2, 4					3, 2, 1, 4				
1, 3, 4, 2					3, 2, 4, 1				
1, 4, 2, 3					3, 4, 1, 2				
1, 4, 3, 2					3, 4, 2, 1				
2, 1, 3, 4					4, 1, 2, 3				
2, 1, 4, 3					4, 1, 3, 2				
2, 3, 1, 4					4, 2, 1, 3				
2, 3, 4, 1					4, 2, 3, 1				
2, 4, 1, 3					4, 3, 1, 2				
2, 4, 3, 1					4, 3, 2, 1				

	$r = 0$	$r = 1$	$r = 2$	$r = 3$
Anzahl Rang 4				

Aufgabe 5 (Python)

`L = [3, 1, 5, 7, 4, 8, 6, 10, 2, 9]`

`max(L[:3])`?

Das Simulationsprogramm

```python
from random import sample

def erzeuge_kandidaten(n):
    '''Gibt [1, 2, 3, ..., n] zufällig gemischt zurück.'''
    return sample(range(1, n+1), n)

def bester_nach_test(liste, r):
    '''Gibt besten Kandiaten bei r Testkandidaten zurück.'''
    test_max = max(liste[:r])
    for wert in liste[r:]:
        if wert > test_max: # ist jemand besser?
            return wert     # wenn ja, gibt seinen Wert zurück
    return liste[-1]        # ... sonst den Wert des letzten

def simulation(n, r, w):
    '''Gibt Anteil der optimalen Wahl bei w Wiederholungen zurück.'''
    erfolge = 0
    best = n
    for i in range(0, w):
        liste = erzeuge_kandidaten(n)
        wahl = bester_nach_test(liste, r)
        if wahl == best:
            erfolge += 1
    return erfolge/w # relative Häufigkeit

# Wiederhole die Simulation w-Mal für jedes r und schreibe
# r und die zugehörige Erfolgsquote in eine CSV-Datei.
dd = open('partnerwahl.csv', mode='w')
dd.write('r;q\n')
n, w = 100, 5000  # Anzahl Kandidaten & Wiederholungen
for r in range(1, n):
    erfolgsquote = simulation(n, r, w)
    print(r, erfolgsquote)
    dd.write('{0};{1}\n'.format(r, erfolgsquote))
dd.close()
```

Aufgabe 6

Führe das obige Simulationsprogramm mit $n = 100$ und $w = 5000$ durch. Öffne die so erzeugte Datei `heiratsproblem.csv` in einem Tabellenkalulationsprogramm und stelle die Daten als Liniendiagramm dar (Werte der erste Kolonne → Achsenbeschriftung).

Für welche Anteil Kandidaten (r/n) sind die Chancen, den besten Partner zu finden, maximal? Vergleiche mit: `https://de.wikipedia.org/wiki/Sekretärinnenproblem`