Optimalität im Sekretärproblem

Effiziente Algorithmen – Sommer 2019

Martin Hoefer*

Szenario:
- n Bewerber. Jeder Bewerber $i = 1, \ldots, n$ hat **eindeutigen Wert** $v_i \geq 0$, mit $v_i \neq v_j$ für $i \neq j$
- Bewerber erscheinen in uniform zufälliger Reihenfolge, einer nach dem anderen
- Jede Runde $t = 1, \ldots, n$ läuft wie folgt:
- Bewerber I_t kommt an und legt v_{I_t} bei Ankunft offen
- **Entscheidung:** Stelle Bewerber I_t ein oder lehne ab.
- $\rightarrow$ Einstellung: Prozess vorbei, Runden $t + 1, \ldots, n$ finden nicht mehr statt
- $\rightarrow$ Ablehnung: Abgelehnter Bewerber kann nie mehr eingestellt werden.

Ziel: Maximiere die Wahrscheinlichkeit, den besten Bewerber einzustellen.

Wir nennen den besten Bewerber b. Dieser ist durch die Werte von Anfang an festgelegt.
Der Bewerber I_t in Runde t wird gemäß der uniform zufälligen Ankunftsreihenfolge festgelegt.

Beobachtungen:
- Wir definieren in Runde t zwei Ereignisse:
  B_t: I_t ist der beste bisher gesehene Bewerber.
  N_t: I_t ist nicht der beste bisher gesehene Bewerber.
- Beachte:
$$\Pr[I_t = b \mid N_t] = 0. \qquad \Pr[I_t = b \mid B_t] = \frac{t}{n}.$$
- Bei N_t ist I_t nicht der beste bisherige Bewerber. Dann kann I_t nicht b sein.
- Bei B_t ist I_t der beste bisherige Bewerber. Wann ist der beste aus den ersten t Runden auch der global beste Bewerber? Genau dann, wenn b in den ersten t Runden kommt. Die Wahrscheinlichkeit dafür ist t/n.

Wann sollten wir nun einstellen? Bei N_t stellen wir nie ein – das kann ja niemals b sein. Bei B_t stellen wir nur ein, wenn die Chancen niedrig sind, dass wir später noch einen besseren Bewerber finden. Das ist am Anfang für kleine Rundenzahlen t so, am Ende aber eher nicht.

Ein optimaler Entscheidungsalgorithmus **nutzt nur Informationen über n, t, B_t und N_t für die Entscheidung** in Runde t. Andere Informationen (z.B. über die bisher gesehenen Werte) sind egal (warum?).

*Die Analyse ist von Beckmann [1].

Wir betrachten folgende Wahrscheinlichkeiten:

- v_t: Wahrscheinlichkeit, dass wir b in Runde $t, t+1, \ldots, n$ einstellen wenn wir N_t erleben.
- u_t: Wahrscheinlichkeit, dass wir b in Runde $t, t+1, \ldots, n$ einstellen wenn wir B_t erleben.

Wir optimieren nun unseren Algorithmus und versuchen, alle u_t und v_t so gross wie möglich zu machen. Wir gehen dabei **von hinten nach vorne** vor.

Rückwärtsiteration:

- Betrachte Runde $t = n$.
  Wenn wir B_n erleben, dann ist $I_n = b$ und wir stellen ihn ein. Also ist $u_n = 1$. Wenn wir N_n erleben, kriegen wir b niemals. Also ist $v_n = 0$.

- Betrachte Runde $t = n-1$.
  Wenn wir B_{n-1} erleben, dann ist $\Pr[I_{n-1} = b \mid B_{n-1}] = \frac{n-1}{n}$.
  $\to$ Wenn wir einstellen, dann kriegen wir b mit Wkeit $(n-1)/n$.
  $\to$ Wenn wir dagegen warten, dann gilt $\Pr[B_n \mid B_{n-1}] = \frac{1}{n}$ und $\Pr[N_n \mid B_{n-1}] = \frac{n-1}{n}$. Die Wkeit b noch in Runde n einzustellen ergibt sich als

  $$v_n \cdot \frac{n-1}{n} + u_n \cdot \frac{1}{n} \ .$$

  Also ist

  $$\begin{aligned} u_{n-1} &= \max\left(\frac{n-1}{n},\ v_n \cdot \frac{n-1}{n} + u_n \cdot \frac{1}{n}\right) \\ &= \frac{n-1}{n} \cdot \max\left(1,\ v_n + u_n \cdot \frac{1}{n-1}\right) \\ &= \frac{n-1}{n} \cdot \max\left(1,\ \frac{1}{n-1}\right) = \frac{n-1}{n} \ . \end{aligned}$$

  Wir sollten also in diesem Fall unbedingt in Runde $n-1$ einstellen.

  Wenn wir N_{n-1} erleben, kriegen wir b nicht in Runde $n-1$. Daher sollten wir hier nicht einstellen, denn eventuell kriegen wir b in Runde n. Es gilt: $\Pr[B_n \mid N_{n-1}] = \frac{1}{n}$ und $\Pr[N_n \mid N_{n-1}] = \frac{n-1}{n}$. Also ist

  $$\begin{aligned} v_{n-1} &= v_n \cdot \frac{n-1}{n} + u_n \cdot \frac{1}{n} \\ &= \frac{1}{n} \\ &= \frac{n-1}{n} \cdot \left(\frac{1}{n-1}\right) \ . \end{aligned}$$

- Betrachte Runde $t = n-2$.
  Wenn wir B_{n-2} erleben, dann ist $\Pr[I_{n-2} = b \mid B_{n-2}] = \frac{n-2}{n}$.
  $\to$ Wenn wir einstellen, dann kriegen wir b mit Wkeit $(n-2)/n$.
  $\to$ Wenn wir dagegen ablehnen und warten, dann gilt $\Pr[B_{n-1} \mid B_{n-2}] = \frac{1}{n-1}$ und $\Pr[N_{n-1} \mid B_{n-2}] = \frac{n-2}{n-1}$. Die Wkeit b noch in Runde $n-1$ oder n einzustellen ergibt sich als

  $$v_{n-1} \cdot \frac{n-2}{n-1} + u_{n-1} \cdot \frac{1}{n-1} \ .$$

Also ist

$$
\begin{aligned}
u_{n-2} &= \max\left(\frac{n-2}{n},\ v_{n-1}\cdot\frac{n-2}{n-1} + u_{n-1}\cdot\frac{1}{n-1}\right) \\
&= \frac{n-2}{n}\cdot\max\left(1,\ v_{n-1}\cdot\frac{n}{n-1} + u_{n-1}\cdot\frac{n}{(n-1)(n-2)}\right) \\
&= \frac{n-2}{n}\cdot\max\left(1,\ \frac{1}{n-1}+\frac{1}{n-2}\right) = \frac{n-2}{n}
\end{aligned}
$$

Wir sollten also in diesem Fall unbedingt in Runde $n-2$ einstellen.

Wenn wir N_{n-2} erleben, kriegen wir b nicht in Runde $n-2$. Daher sollten wir hier nicht einstellen, denn eventuell kriegen wir b in Runde $n-1$ oder n. Es gilt $\Pr[B_{n-1} \mid N_{n-2}] = \frac{1}{n-1}$ und $\Pr[N_{n-1} \mid N_{n-2}] = \frac{n-2}{n-1}$. Also ist

$$
\begin{aligned}
v_{n-2} &= v_{n-1}\cdot\frac{n-2}{n-1} + u_{n-1}\cdot\frac{1}{n-1} \\
&= \frac{n-2}{n(n-1)} + \frac{1}{n} \\
&= \frac{n-2}{n}\cdot\left(\frac{1}{n-1}+\frac{1}{n-2}\right)
\end{aligned}
$$

- Für Runde t kann man per Induktion zeigen:

$$
\begin{aligned}
u_t &= \frac{t}{n}\cdot\max\left(1, \frac{1}{n-1}+\frac{1}{n-2}+\ldots+\frac{1}{t}\right) \\
v_t &= \frac{t}{n}\cdot\left(\frac{1}{n-1}+\frac{1}{n-2}+\ldots+\frac{1}{t}\right)
\end{aligned}
$$

Damit wird der optimale Algorithmus offensichtlich:

- Bei N_t ist es niemals sinnvoll, den momentanen Kandidaten einzustellen.
- Im Fall von B_t und für große t (später Zeitpunkt) so dass

$$\frac{1}{n-1}+\frac{1}{n-2}+\ldots+\frac{1}{t} = \sum_{i=t}^{n-1}\frac{1}{i} < 1\ .$$

  ist es optimal, den momentanen (= bisher besten) Kandidaten einzustellen.
- Wenn B_t eintritt und t zu klein (zu früher Zeitpunkt) ist, dann gilt

$$\frac{1}{n-1}+\frac{1}{n-2}+\ldots+\frac{1}{t} = \sum_{i=t}^{n-1}\frac{1}{i} > 1\ .$$

  Hier ist es optimal, niemals einzustellen und dafür auf spätere Runden und bessere Kandidaten zu warten.
- Der optimale Schwellwert r ist gegeben durch

$$\sum_{i=r}^{n}\frac{1}{i} \geq 1 \qquad \text{und} \qquad \sum_{i=r+1}^{n}\frac{1}{i} \leq 1\ .$$

  Durch Näherung der Summe mit Integral kann man zeigen, dass $r = \lfloor n/e \rfloor$. Der optimale Algorithmus wartet also bis Runde r. Ab Runde $r+1$ stellt er immer genau dann ein, wenn I_t der beste bisherige Kandidat ist.

Literatur

[1] M. Beckmann. Dynamic Programming and the Secretary Problem. *Computers Math. Applic.*, 19(11):25–28, 1990.