Das Bahncard-Problem

von Andre Hebben

March 5, 2001

1 Vorstellung des Problems

1.1 Praxis:

Ein unregelmässiger Fahrgast der Bahn stellt sich vor jeder Fahrt die Frage, ob es sich für ihn lohnt, eine Bahncard zu kaufen oder nicht. Diese Frage soll ihm ein geeigneter Algorithmus möglichst gut (d.h. bei möglichst geringen Kosten) beantworten.

1.2 Abstraktion

Das Problem wird durch drei Konstanten charakterisiert:

$$BP(C, \beta, T).$$

Wobei:

$C > 0$:=Kosten einer Bahncard, (240DM).
$\beta[0, 1]$:=Faktor mit dem die Fahrpreise multipliziert werden dürfen, (0.5).
$T > 0$:=Gültigkeitsdauer einer Bahncard (1Jahr).

(In Klammern die jeweiligen Werte für die Bahncard der Deutschen Bahn)

1.2.1 Input:

Eingabefolge von Anfragen $\sigma = \sigma_1\sigma_2\sigma_3...$ mit $\sigma_i = (t_i, p_i)$.

Wobei t_i =Zeitpunkt der Anfrage und p_i =Regulärer Preis der Anfrage. Der Algorithmus kann sich nun bei jeder Anfrage, bei der er noch nicht in Besitz

einer gültigen Bahncard ist, dazu entscheiden eine Bahncard zu kaufen. Sie ist dann sofort gültig.

1.2.2 Output:

B-Schedule $\Gamma(\sigma)$ mit $\tau_1 < \tau_2 < \tau_3 ...$ wobei jedes τ_i einem Zeitpunkt t_i aus der Eingabefolge entspricht, an dem der Algorithmus eine Bahncard kaufen möchte.

1.2.3 Kosten:

Die zu minimierenen Kosten entsprechen den tatsächlich anfallenden Kosten für die Bahncards sowie für die zu bezahlenden Fahrtickets.

2 Ein optimaler Offline-Algorithmus

Sei $\sigma = \sigma_1 ... \sigma_n$ eine Eingabefolge mit n Anfragen. Der Algorithmus konstruiert daraus einen gewichteten azyklischen Graphen G_σ mit Knoten $s = \sigma_0, \sigma_1, ..., \sigma_n, \sigma_{n+1} = t$, wobei $s = (0,0)$ und $t = (t_n + T, 0)$ zwei "künstliche" Anfragen sind, die zu Beginn und am Ende hinzugefügt werden. Von jedem Knoten σ_i geht nun eine Kante zum Knoten σ_{i+1} mit dem Kantengewicht p_i. Diese Kante entspricht dem Fahren ohne Bahncard. Ausserdem wird noch eine Kante zum dem Knoten, der nach einer Gültigkeitsdauer T erreicht wird, hinzugefügt. Diese Kante bekommt als Gewicht die Summe aller reduziert bezahlten Anfragen im Intervall $[t_i, t_i + T)$ zuzüglich der Kosten für eine Bahncard zugeordnet und entspricht dem Fahren mit Bahncard innerhalb dieses Zeitraums. $G_\sigma$hat nun die Eigenschaft, dass jeder $(s \rightarrow^* t)$-Pfad in G_σ genau einem B-Schedule und das Gesamtgewicht dieses Pfades den anfallenden Kosten entspricht. Es wird nun also mittels Dijkstra-Algorithmus der kürzeste $(s \rightarrow^* t)$-Pfad gesucht.

Es zeigt sich, dass die Kosten die in den Intervallen in denen der optimale Algorithmus eine Bahncard besitzt immer über einem kritischen Wert liegt. Dieser Wert ist

$$c_{crit} = \frac{C}{1-\beta}$$

und entspricht den Kosten ab denen die Ersparnis durch reduzierte Fahrpreise die Mehrkosten durch Kauf der Bahncard aufwiegt. Phasen in denen die Anfragesumme über c_{crit} liegt heissen teuer, die anderen Zeiträume preiswert.

3 Untere Schranke für deterministische Online-Algorithmen

3.1 Competitivität

Ein Online-Algorithmus A ist *d-competitiv*, wenn für alle Eingabefolgen σ gilt:

$$c_A(\sigma) \leq d \cdot c_{opt}(\sigma).$$

3.2 Untere Schranke

Wir konstruieren uns zunächst eine möglichst schlechte Eingabefolge für die Online-Algorithmen: Sei $\epsilon > 0$ eine vernachlässigbar kleine Konstante. Wie konfrontieren den Online-Algorithmus nun mit sehr dicht gedrängten Anfragen, die alle die Kosten ϵ produzieren. Alle Anfragen liegen innerhalb einer Gültigkeitslänge $[0, T)$. Entschliesst sich der Algorithmus schliesslich eine Bahncard zu kaufen, endet die Eingabefolge sofort. Sei s die angefallenen Kosten ohne die Kosten für die letzte Anfrage. Wir bilden nun $\frac{c_{onl}}{c_{opt}}$.

Es ist:

$c_{onl} = C + s + \beta\epsilon$

und:

$$c_{opt} = \begin{cases} s + \epsilon & s + \epsilon \leq c_{crit} \\ & \text{für} \\ C + \beta(s + \epsilon) & s + \epsilon \geq c_{crit} \end{cases}$$

Also ist

$$\frac{c_{onl}}{c_{opt}} = \begin{cases} \frac{C+s+\beta\epsilon}{s+\epsilon} & s + \epsilon \leq c_{crit} \\ & \text{für} \\ \frac{C+s+\beta\epsilon}{C+\beta(s+\epsilon)} & s + \epsilon \geq c_{crit} \end{cases}$$

Wir setzen nun s so, dass der Quotient möglichst klein wird, was für beide Fälle bei $s = c_{crit} - \epsilon$ gilt. Eingesetzt ergibt sich:

$$\frac{c_{onl}}{c_{opt}} \geq \frac{C+c_{crit}-\epsilon+\beta\epsilon}{c_{crit}} = \frac{C+\frac{C}{(1-\beta)}-(1-\beta)\epsilon}{\frac{C}{(1-\beta)}} = \frac{(1-\beta)C+C-\epsilon(1-\beta)^2}{C} =$$

$$2 - \beta - \frac{\epsilon(1-\beta)^2}{C} \quad \text{für } s + \epsilon \leq c_{crit}$$

und

$$\frac{c_{onl}}{c_{opt}} \geq \frac{C + c_{crit} - \epsilon + \beta\epsilon}{C + \beta(c_{crit})} = \frac{C + \frac{C}{(1-\beta)} - (1-\beta)\epsilon}{\frac{C}{(1-\beta)}} =$$

$$2 - \beta - \frac{\epsilon(1-\beta)^2}{C} \quad \text{für } s + \epsilon \geq c_{crit}$$

Ein optimaler Online-Algorithmus muss also $(2 - \beta)$-competetiv sein.

Zwei einfache Online-Algorithmen erreichen diese Schranke nicht. Zum einen NEVER, der niemals eine Bahncard kauft. Dieser ist $\frac{1}{\beta}$-competetiv. Der TICKET-OFFICE-Algorithmus ist dem Verhalten eines Bahnangestellten nachempfunden, der einem erst dann zu einer Bahncard rät, wenn die Kosten einer einzelnen Fahrt c_{crit} übersteigen. Der TOA ist im Durchschnitt zwar besser als NEVER, da er teuere Fahrten besser handelt. Im Worst-Case ist er allerdings nicht besser, da auch er bei vielen Anfragen mit $c = c_{crit} - \epsilon$ keine Bahncard kaufen wird. Besser schlägt sich der im Folgenden vorgestellte Online-Algorithmus SUM:

4 Optimaler Online-Algorithmus SUM

4.1 Wie arbeitet SUM?

SUM merkt sich die Anfragen für die Dauer der Gültigkeit einer Bahncard. Erreicht die Summe der nicht ermässigten Fahrten in diesem Zeitraum einschliesslich der aktuellen Anfrage mindestens c_{crit}, wird eine Bahncard gekauft, ansonsten nicht. Also

$$rc(t) \geq c_{crit}$$

4.2 Competitivität

Behauptung: SUM ist $(2-\beta)$-competitiv und somit optimal.

Beweis: Sei $\sigma = \sigma_1\sigma_2...$ eine Eingabefoge und $OPT(\sigma) = \tau_1\tau_2...\tau_k$ der dazugehörige optimale B-Schedule. Mittels dieses B-Schedules lässt sich die Zeit in Epochen einteilen $[\tau_j, \tau_{j+1})$ mit $0 \leq j \leq k$. Jede Epoche beginnt mit einer teuren Phase (die Zeit in der OPT eine Bahncard besitzt) und endet mit einer preiswerten Phase. SUM wird in jeder Epoche höchstens eine Bahncard kaufen. Diese lässt sich der teuren Phase dieser Epoche zurechnen. Die preiswerten Phasen sind dann unkritsch, da in dieser Zeit SUM nicht mehr Kosten verursachen kann als OPT. Die teuren Phasen werden nun genauer betrachtet und in drei Unterphasen I_1, I_2, I_3 zerlegt. Einzelne Unterphasen können auch leer sein. In I_1 und I_3 besitzt SUM eine Bahncard, während er in Phase I_2 keine Bahncard besitzt und somit die vollen Fahrpreise zahlen muss. Sei nun s_i die Anfragesumme, die in Phase I_i entsteht. Dann ergibt sich für die Kosten von SUM bzw. OPT:

$$c_{sum} = C + s_2 + \beta \cdot (s_1 + s_3)$$
$$c_{opt} = C + \beta(s_1 + s_2 + s_3)$$

$$\frac{c_{sum}}{c_{opt}} = \frac{C+s_2+\beta\cdot(s_1+s_3)}{C+\beta(s_1+s_2+s_3)} \leq \frac{C+c_{crit}}{C+\beta(c_{crit})} = \frac{C+\frac{C}{(1-\beta)}}{C+\frac{\beta C}{(1-\beta)}} = 2-\beta$$

SUM ist also optimal. Er verhält sich aber recht pessimistisch, da er erst dann eine Bahncard kauft, wenn OPT mit Sicherheit schon eine Bahncard besitzt. Im Folgenden werde ich einen weiteren optimalen Online-Algorithmus vorstellen, der früher als SUM seine Bahncards kauft.

5 Optimaler Online-Algorithmus OSUM

5.1 Wie arbeitet OSUM?

OSUM merkt sich ebenfalls die innerhalb des letzten Gültigkeitsintervalls durch reguläre Anfragen angefallenen Kosten und setzt diese mit der zu entscheidenden

Anfrage wie folgt in Beziehung:

$$p \geq \frac{C - s(1-\beta)}{2(1-\beta)} =: a$$

5.2 Competitivität

Behauptung: OSUM ist $(2-\beta)$-competitiv und somit optimal.

Beweis: Der Beweis verläuft ähnlich wie bei SUM. Da OSUM aber innerhalb einer Epoche eventuell mehr als eine Bahncard kauft, können wie nicht mehr alle Bahncards den teuren Phasen zurechnen. Kauft sich OSUM innerhalb einer Epoche eine zusätzliche Bahncard, bezeichenen wir den Zeitabschnitt ab Ablauf der letzten Bahncard bis zu dem Zeitpunkt an dem die Entscheidung fällt als kritische Phase. Eine Epoche besteht dann aus einer teuren Phase und sich abwechselnden preiswerten und kritischen Phasen. Die teuren und preiswerten Phasen lassen sich wie bei SUM gezeigt abhandeln. Für kritische Phasen ergeben sich folgende Kosten:

$$c_{osum} = C + s + \beta p$$
$$c_{opt} = s + p$$

$$\frac{c_{osum}}{c_{opt}} = \frac{C+s+\beta p}{s+p} \leq \frac{C+s+\beta a}{s+a} = \frac{C+s+\beta(\frac{C-s(1-\beta)}{2(1-\beta)})}{s+\frac{C-s(1-\beta)}{2(1-\beta)}} =$$

$$\frac{2[C+s(1-\beta)]-\beta[C+s(1-\beta)]}{C+s(1-\beta)} = 2-\beta$$