Fahrplanoptimierung im ÖPNV

Zülfükar Genç, genc@informatik.uni-koeln.de
Ewald Speckenmeyer esp@informatik.uni-koeln.de

Institut für Informatik, Universität zu Köln, D-50969, Germany

Zusammenfassung

Dieser Artikel behandelt einen Ansatz zur Optimierung von Fahrplänen im ÖPNV. Bei der Optimierung hat man einen festen Linienplan gegeben. Man möchte die Abfahrtszeiten der Linien so setzen, dass ein möglichst großer Sicherheitsabstand entsteht. Wir werden das Modell und das Fahrplan-Problem beschreiben und einen Branch-and-Bound Lösungsansatz mit einigen anderen Fragestellungen darstellen. Die Optimierung ist ein Teil eines Projekts "CATS" (Computer Aided Tram Scheduling), das die computerunterstützte Erstellung eines Fahrplanes für Straßenbahnnetze ermöglichen soll.

1 Einführung

"Der Betrieb eines öffentlichen Nahverkehrssystems kann - aus der abstrakten Sicht eines Mathematikers - als ein gigantisches Optimierungsproblem mit komplexen Nebenbedingungen aufgefaßt werden."[GLV96] Grob kann man die dabei auftretenden Optimierungsprobleme in fünf Bereiche gliedern. Dies sind die Optimierung des Linienplans, des Fahrplans, des Fahrzeugumlaufs, des Dienstplans und Dienstreihenfolgeplans.

Durch enorme Fortschritte in der kombinatorischen Optimierung und der Computertechnik ist es heute möglich, viele $\mathcal{NP}$-harte Probleme, die in diesen Bereichen entstehen, für Instanzgrößen exakt zu lösen, die in der Vergangenheit praktisch nicht gelöst werden konnten.

Das Ziel der Fahrplanoptimierung ist es, die Startzeiten der einzelnen Linien so zu setzen, dass ein guter Fahrplan entsteht. Bei vielen Arbeiten ist das Ziel der Fahrplanoptimierung eine gute Abstimmung der Umsteigebeziehungen. Damit versucht man die Umsteigezeiten und somit auch die Reisezeit der Fahrgäste zu verringern. ([DFV92],[Voß92],[Dom89],[DV95],[NV95],[KM00])

In unserem Projekt "CATS" (Computer Aided Tram Scheduling) wird ein Tool entwickelt, das die computerunterstützte Erstellung eines Fahrplanes für Straßenbahnnetze ermöglichen soll. Das Projekt setzt sich aus Optimierung und Simulation zusammen. Das Simulationstool soll es den Verkehrsunternehmen ermöglichen, interaktiv Störungsanfälligkeiten von möglichen wie aktuellen Fahrplänen zu testen und damit bessere Fahrpläne zu erstellen.

Bei der Fahrplanoptimierung wollen wir nicht die Umsteigezeiten minimieren, sondern wir konzentrieren uns auf den Sicherheitsabstand der Linien. Unter dem Sicherheitsabstand an einer Station für einen Fahrplan verstehen wir den minimalen zeitlichen Abstand der Ankunftszeiten zweier aufeinanderfolgender

Linien an dieser Station. Wir sind der Meinung, dass ein guter Fahrplan die Ankunftszeiten der Linien an einer beliebigen Station gut verteilen sollte.

Zur Lösung dieses Problems, das wir als Fahrplan-Problem bezeichen, haben wir den Branch-and-Bound Ansatz gewählt und Tests mit realen Daten der Kölner Verkehrs-Betriebe AG (KVB) durchgeführt. Das Streckennetzwerk der KVB besteht je nach Uhrzeit zwischen 18 und 28 Linien mit mehr als 500 Stationen, wobei die Linien und Stationen in Hin- und Rückrichtung unterschieden werden. Nach ein paar Minuten liefert der Branch-and-Bound Algorithmus für das Streckennetzwerk der KVB ganz gute Ergebnisse.

Da das Streckennetzwerk sehr gross ist, haben wir uns die Frage gestellt, ob auch wirklich alle Stationen für das Fahrplan-Problem eine Rolle spielen. Darum haben wir nach einer Reduktion des Streckennetzwerkes gesucht mit dem Ziel keine Lösungen für das Fahrplan-Problem zu verlieren. Mit Hilfe geeigneter Relationen auf der Menge der Stationen konnten wir die Anzahl der Stationen, die für das Fahrplan-Problem relevant sind, um 90% reduzieren.

Wegen der Grösse des Streckennetzwerks haben wir auch die Frage untersucht, ob man das Streckennetz geschickt aufteilen kann, so dass aus den optimalen Fahrplänen (bzgl. des Fahrplan-Problems) der Teilnetze eine optimaler Fahrplan für das gesamte Netz konstruiert werden kann. Dafür haben wir einen speziellen Graphen (Linienkonflikt-Graph) betrachtet. In diesem Graphen sind die Knoten die Linien und es existiert eine Kante zwischen zwei Knoten (=Linien), wenn beide Linien eine gemeinsame Station haben. Wenn man Teilnetze aus den zweifachen Zusammenhangskomponenten konstruiert, und auf diesen Fahrpläne erzeugt, kann man aus diesen Fahrplänen einen Fahrplan für das gesamte Streckennetzwerk konstruieren ohne dabei die Teilfahrplänen zu verschlechtern. Leider brachte dieser Ansatz für das Streckennetzwerk der KVB nicht viel, da der Linienkonflikt-Graph eine sehr grosse zweifache Zusammenhangskomponente besitzt.

2 Fahrplan-Problem

Gegeben ist ein Streckennetzwerk $N(S, C, t, L, T)$ mit Stationen $S = \{s_1, \ldots, s_n\}$, Linien $L = \{l_1, \ldots, l_m\}$ und Verbindungen (Connections) $C = \{c_1, \ldots, c_v\}$ zwischen diesen Stationen. Jeder Verbindung wird durch $t : C \to \mathbb{N}_0$ eine Fahrzeit zugeordnet. Eine Linie $l_j = (s_{i_1} s_{i_2} \ldots s_{i_{k(j)}})$ ist ein einfacher gerichteter Weg im Streckennetzwerk N. Jede Linie l_j besitzt einen bestimmten Takt T_j. Die Menge der Takte bezeichnen wir mit $T = \{T_1, T_2, \ldots, T_m\}$. Die Abfahrtszeit der Linie l_j an der Anfangsstation s_{i_1} bezeichnen wir mit λ_j, den Vektor der m Abfahrtszeiten (Fahrplan) aller Linien mit λ und die Menge aller Abfahrtszeiten mit $\Lambda = \mathbb{N}_{T_1} \times \mathbb{N}_{T_2} \times \cdots \times \mathbb{N}_{T_m}$, wobei $\mathbb{N}_k = \{0, \ldots, k-1\}$ sei. Man kann sich bei den Abfahrtzeiten einer Linie l auf die Zeiten $\mathbb{N}_{T_l}$ beschränken, da die Linie den Takt T_l besitzt.

Definition 2.1 *Sei $s \in S$ und $l \in L$.*
Die Menge aller Linien, die durch die Station s fahren, bezeichnen wir mit $L(s)$.
Die Menge aller Stationen, die von der Linie l befahren werden, bezeichnen wir mit $S(l)$.

Definition 2.2 *(Sicherheitsabstand)*
$\delta(s, \lambda)$ *ist der minimale* Sicherheitsabstand *zweier aufeinander folgender Linien an der Station* s *für den Abfahrtszeitenvektor* λ*.*

$$\delta(\lambda) := \min_{s \in S} \delta(s, \lambda) \tag{2.1}$$

$\delta(\lambda)$ *ist der minimale* Sicherheitsabstand *über alle Stationen für den Abfahrtszeitenvektor* λ*.*

$$\delta_{\Sigma}(\lambda) := \sum_{s \in S} \delta(s, \lambda) \tag{2.2}$$

$\delta_{\Sigma}(\lambda)$ *ist die* Summe der Sicherheitsabstände *über alle Stationen für den Abfahrtszeitenvektor* λ*.*

Definition 2.3 *(Sicherheitsabstand-Problem)*
Gegeben ist ein Streckennetzwerk $N(S, C, t, L, T)$*. Finde ein* $\lambda_0 \in \Lambda$ *mit* $\delta(\lambda_0) = \delta_{opt}$*, wobei* δ_{opt} *den optimalen Sicherheitsabstand bezeichnet und wie folgt definiert ist:*

$$\delta_{opt} := \max_{\lambda \in \Lambda} \min_{s \in S} \delta(s, \lambda) = \max_{\lambda \in \Lambda} \delta(\lambda) \tag{2.3}$$

Als Entscheidungsproblem formuliert, betrachten wir einen festen Sicherheitsabstand δ und fragen, ob ein Abfahrtszeitvektor $\lambda \in \Lambda$ existiert, so dass

$$\delta(\lambda) \geq \delta$$

erfüllt ist. Das Entscheidungsproblem ist $\mathcal{NP}$-vollständig [Gen99]. Die $\mathcal{NP}$-Vollständigkeit wird durch Reduktion des $\mathcal{NP}$-vollständigen k-Färbungsproblems [GJ79] auf das Entscheidungsproblem bewiesen.

Das Sicherheitsabstand-Problem taucht in der Literatur in einer anderen Form auf ([Vin89], [Gul80], [Bur86], [CG87], [Hur92]). Dabei werden reguläre Polygone untersucht, man versucht diese so in einen Kreis einzubetten, dass die Abständen zwischen den Ecken der Polygone bestimmte Zielfunktionen maximieren oder minimieren. Dieses Problem ist für einen Spezialfall äquivalent zum Sicherheitsabstand-Problem, falls alle Linien die gleiche Strecke befahren oder wir nur eine Station betrachten.

Das Sicherheitsabstand-Problem haben wir untersucht und konnten gute Ergebnisse erzielen, aber die Untersuchung des minimalen Sicherheitsabstandes zielt im wesentlichen auf die optimale Benutzung stark befahrener Stationen ab, vernachlässigt allerdings wenig befahrene Stationen in Randbezirken. Damit die Stationen in den Randbezirken auch berücksichtigt werden, muss man unter den Lösungen, die den geforderten Sicherheitsabstand einhalten, ein weiteres Kriterium ansetzen. Durch dieses sollte eine möglichst gleichmäßige Verteilung der Ankunftszeiten auf das Taktintervall erreicht werden, d.h. der Sicherheitsabstand sollte möglichst dem optimalen Einzelabstand entsprechen. Aus dem Sicherheitsabstand-Problem ergibt sich das Fahrplan-Problem, das für das ganze Streckennetzwerk geeigneter ist.

Definition 2.4 *(Fahrplan-Problem)*
Gegeben ist ein Streckennetzwerk $N(S, C, t, L, T)$*. Finde ein* $\lambda_0 \in \Lambda$ *mit* $\delta(\lambda_0) =$

δ^{Σ}_{opt}, *wobei* δ^{Σ}_{opt} *den optimalen Sicherheitsabstand bezeichnet und wie folgt definiert ist:*

$$\delta^{\Sigma}_{opt} := \max_{\lambda \in \Lambda : \delta(\lambda) = \delta_{opt}} \sum_{s \in S} \delta(s, \lambda) = \max_{\lambda \in \Lambda : \delta(\lambda) = \delta_{opt}} \delta_{\Sigma}(\lambda) \tag{2.4}$$

wobei δ_{opt} *den optimalen Sicherheitsabstand bezeichnet.*

3 Fragestellungen

3.1 Branch-and-Bound

Für das Fahrplan-Problem haben wir den Branch-and-Bound-Ansatz zur Ermittlung einer exakten Lösung gewählt wie für das Sicherheitsabstand-Problem.

Bei der Methode Branch-and-Bound wird ein Optimierungsproblem rekursiv in Teilprobleme zerlegt (branch). Zuerst hat man einen Fahrplan, bei dem keine Fahrzeit festgelegt wurde. Man wählt ein Linie aus und setzt alle möglichen Abfahrtszeiten, dadurch entstehen die neuen Teilprobleme.

Man speichert noch zu lösende Teilprobleme in einer Prioritätsschlange ab. Zu Anfang enthält die Prioritätsschlange nur das Anfangsproblem. In jedem Schritt wird ein Teilproblem aus der Prioritätsschlange ausgewählt, dann wird für dieses Teilproblem eine Schranke (bound) berechnet. Es werden zunächst die Teilprobleme bearbeitet, die den größtmöglichen Zielfunktionswert erwarten lassen. Da man hier eine Maximierungsproblem hat, wird eine lokale obere Schranke für das Teilproblem berechnet. Die Schranken werden mit Hilfe der festgelegten Abfahrtszeiten im Teilproblem und globalen oberen Schranken für den Sicherheitsabstand an den einzelnen Stationen berechnet. Wenn diese lokale obere Schranke kleiner als die globale untere Schranke ist, so wird dieses Teilproblem verworfen. Andernfalls wird das Teilproblem weiter zerlegt und in die Prioritätsschlange aufgenommen. Die globale untere Schranke ist der Optimalwert der besten bis dahin gefundene Lösung des Optimierungsproblems.

Die realen Testdaten stammen von der Kölner Verkehrs-Betriebe AG (KVB). Für jede Stunde an Werktagen haben wir aus diesen Daten eine Streckennetzwerk-Instanz für unser Programm erzeugt.

		oberen Schranken		B&B-Alg.			
Inst.	$\|L\|$	$\overline{\delta}$	$\overline{\delta_\Sigma}$	δ	δ_Σ	$\frac{\overline{\delta_\Sigma} - \delta_\Sigma}{\overline{\delta_\Sigma}}$ (%)	Zeit (sec.)
I_1	18	15	20740	15	20204	2.58	9.3
I_2	20	12	24118	12	23535	2.41	84.24
I_4	22	3	8229	3	7934	3.58	41.55
I_9	28	2	4763	2	4596	3.51	37.4
I_{20}	26	3	6714	**2**	6532	2.71	20.78

Tabelle 1: Ergebnisse für das Strassenbahnnetz der KVB nach 100 Sekunden.

Bis jetzt konnte für keine Instanz eine optimale Lösung für das Fahrplan-Problem ermittelt werden. Die Lösung, die der Branch-and-Bound-Alg. nach

einer kurzen Zeit liefert, ist gut, aber verbessert sich nach ein paar Minuten kaum noch. Das Sicherheitsabstand-Problem konnten wir jedoch für fast alle Instanzen in wenigen Minuten exakt lösen.

Wir haben Heuristiken entwickelt, aber diese konnten nach Stunden nicht so gute Ergebnisse für das Fahrplan-Problem liefern wie der Branch-and-Bound-Algorithmus nach einigen Sekunden.

3.2 Reduktion des Streckennetzwerks

Im Branch-and-Bound-Algorithmus wird sehr oft der Sicherheitsabstand und die Summe der Sicherheitsabstände berechnet. Da diese oft die meiste Rechenleistung benötigen, wollen wir durch Reduktion die Branch-and-Bound-Schritte schneller machen. Wie wir an der Definition des Sicherheitsabstands und der Summe der Sicherheitsabstände sehen, hängen diese sehr stark von der Stationsmenge ab. Viele Stationen eines Streckennetzwerkes sind aber überflüssig für die Berechnung der Sicherheitsabstände, weil sie nur von einer Linie oder von gleichen Linien befahren werden oder deren Informationen sind in anderen Stationen enthalten. Unser Anliegen wird es sein, die Menge der Stationen so zu reduzieren, dass mit der reduzierten Stationsmenge gearbeitet werden kann ohne irgendwelche Information zu verlieren.

Es stellt sich die Frage, welche Stationen sind wirklich nötig, um den Sicherheitsabstand (bzw. die Summe der Sicherheitsabstände) für einen gegebenen Abfahrtszeitenvektor zu berechnen. Die Reduktion der Stationsmenge basiert auf (Äquivalenz-)Relationen über die Menge der Stationen.

Definition 3.1 *Sei S die Menge der Stationen eines Streckennetzwerkes. Definiere eine Äquivalenz-Relation R_Σ und eine Relation R_δ auf der Menge S wie folgt:*

$$\begin{array}{c|c} s_1 \sim_{R_\Sigma} s_2 \Leftrightarrow & s_1 \leq_{R_\delta} s_2 \Leftrightarrow \\ L(s_1) = L(s_2) \text{ und} & L(s_1) \subseteq L(s_2) \text{ und} \\ \multicolumn{2}{c}{\exists c \in \mathbb{Z}\ \forall l \in L(s_1) : \text{Fahrzeit von Station } s_1 \text{ nach } s_2 \text{ der Linie } l{=}c} \end{array}$$

Instanzen	$\lvert L\rvert$	ohne Red. $\lvert S\rvert$	Relation $\lvert S_\Sigma\rvert$	$\lvert S_\delta\rvert$
I_1	18	393	22	14
I_2	20	461	28	16
I_4	22	496	32	14
I_9	28	543	48	28
I_{20}	26	525	44	22

Tabelle 2: Ergebnisse der Stationsreduktion für das Strassenbahnnetz der KVB an Werktagen.

Wie man an der Tabelle 2 sieht, konnte mit Hilfe dieser Relationen die Anzahl der Stationen, die für die Berechnung des Sicherheitsabstandes($|S_\delta|$) (bzw. Summe der Sicherheitsabstände ($|S_\Sigma|$)) benötigt werden, um mindestens 95% (bzw. 90%) reduziert werden.

Wenn man mit Hilfe der Relationen im Branch-and-Bound-Algorithmus den Sicherheitsabstand und die Summe der Sicherheitsabstände berechnet, so kann man etwa drei- bis vierfach schneller zu der selben Lösung gelangen.

3.3 Unterteilung des Netzwerks

Wir haben die Frage untersucht, wie wir das Streckennetzwerk aufteilen können, so dass aus optimalen Lösungen der Teilnetze eine optimale Lösung für das gesamte Netz entsteht.

Zu diesem Zweck untersuchen wir den Linienkonflikt-Graphen (LK-Graph).

Definition 3.2 *Der* Linienkonflikt-Graph *(LK-Graph) zum Streckennetzwerk $N(S, C, t, L, T)$ ist definiert als LK-$G(N) = (V, E)$ mit der Knotenmenge $V = L$ und eine Kante zwischen zwei Linien (=Knoten) existiert, wenn sich die Linien in einer Station kreuzen. D.h. es gilt:*

$$e = (l_i, l_j) \in E \Leftrightarrow S(l_i) \cap S(l_j) \neq \emptyset$$

Wir konnten zeigen, dass aus den optimalen Lösungen der Teilnetze, die durch die zweifachen Zusammenhangskomponenten von LK-Graph gebildet werden, eine optimale Lösung für das Gesamtennetzwerk in linearer Zeit bzgl. Anzahl der Linien konstruieren lässt.

Leider haben viele Streckennetzwerke im ÖPNV eine sehr große zweifache Zusammenhangskomponente, darum bringt der obige Ansatz nicht viel.

4 Zusammenfassung und Ausblick

Der Branch-and-Bound-Ansatz hat sich als ein sehr gutes Verfahren herausgestellt. Gegenüber Heuristiken lieferte das Branch-and-Bound-Verfahren sehr gute Ergebnisse bei gleicher Laufzeit. Leider konnte für keine Instanz eine optimale Lösung für das Fahrplan-Problem ermittelt werden, obwohl das Branch-and-Bound-Verfahren bei dem Sicherheitsabstand-Problem nach kurzer Zeit eine optimal Lösung gefunden hat. Ursache dafür ist, dass die Schranken, die wir für das Sicherheitsabstand-Problem ermittelt haben und mit denen auch beim Fahrplan-Problem gearbeitet wurde, nicht ausreichen, darum können nicht genügend große Teilbereiche des Suchbaums ausgeschlossen werden. In diesem Kontext muss noch einiges geleistet werden.

Durch die Reduktion des Streckennetzwerkes konnten wir die Laufzeit der Branch-and-Bound-Schritte um einiges verbessern. Die Reduktion verdeutlicht, dass nur wenige Stationen für das Fahrplan-Problem eine wichtige Rolle spielen.

Literatur

[Bur86] Rainer E. Burkard. Optimal schedules for periodically recurring events. *Discrete Applied Mathematics*, 15:167–180, 1986.

[CG87] Jan Cerny and Filip Guldan. Location of polygon vertices on circles and its application in transport studies. *Aplikace Matematiky*, 32:81–95, 1987.

[DFV92] W. Domschke, P. Forst, and S. Voß. Tabu search techniques for the quadratic semi-assignment problem. In G. Fandel, T. Gullegde, and A. Jones, editors, *New Directions for Operations Research in Manufacturing*, pages 389–405. Springer, 1992.

[Dom89] W. Domschke. Schedule synchronization for public transit networks. *OR Spektrum 11*, pages 17–24, Mar 1989.

[DV95] Joachim R. Daduna and S. Voß. Practical experiences in schedule synchronization. In Joachim R. Daduna, Isabel Branco, and Jose M. Pinto Paixao, editors, *Computer-Aided Transit Scheduling*, pages 39–55. Springer, 1995.

[Gen99] Z. Genc. *Ein Aspekt der Fahrplanoptimierung im ÖPNV : Maximierung minimaler Sicherheitsabstände*. Diplomarbeit, Universität zu Köln, 1999.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability*. Freeman, 1979.

[GLV96] Martin Grötschel, Andreas Löbel, and Manfred Völker. Optimierung des Fahrzeugumlaufs im öffentlichen Nahverkehr. *Heureka '96: Optimierung in Verkehr und Transport, Tagungsbericht*, pages 341–355, Aug 1996.

[Gul80] F. Guldan. Maximization of distances of regular polygons on a circle. *Aplikace Matematiky*, 25:182–195, 1980.

[Hur92] Johann Hurink. *Polygon scheduling*. Dissertation, Fachbereich Mathematik/Informatik, Universität Osnabrück, 1992.

[KM00] Ioannis Kontas and Jörg Mühlenkamp. *Fahrplanoptimierung im öffentlichen Nahverkehr*. Diplomarbeit, Universität Dortmund, 2000.

[NV95] Karl Nachtigall and Stefan Voget. Optimierung von periodischen Netzplänen mit genetischen Algorithmen. Jan 1995.

[Vin89] A. Vince. Scheduling periodic events. *Discrete Appl. Math.*, 25(3):299–310, 1989.

[Voß92] S. Voß. Network design formulations in schedule synchronization. In M. Desrochers and J.M. Rousseau, editors, *Computer-Aided Transit Scheduling*, volume 386, pages 137–152. Springer, 1992.