Mathematical Aspects of Public Transportation Networks

Niels Lindner

June 18, 2018
Chapter 4

Vehicle Scheduling

§ 4.1 Periodic Vehicle Scheduling
§4.1 Periodic Vehicle Scheduling

Public transport planning cycle

- Network Design
- Line Planning
- Timetabling
- Vehicle Scheduling
- Duty Scheduling
- Crew Rostering

strategic planning
operational planning
4.1 Periodic Vehicle Scheduling

Vehicle Scheduling: Overview

Main Question
Given a line network with a timetable, how many vehicles are required to operate the timetable?

Scope

- basic: periodic timetables
- better: aperiodic timetables (e.g., for a day)
- realistic: several depots, vehicle types, capacities, ...

All these versions lead to network flow problems. In some scenarios, the minimal number of vehicles is replaced by a more general cost function.
4.1 Periodic Vehicle Scheduling

Periodic Vehicle Scheduling

Input data

- event-activity network $\mathcal{E} = (V, E)$
- period time $T \in \mathbb{N}$
- periodic timetable $\pi: V \rightarrow [0, T)$
- periodic tensions $x: E \rightarrow \mathbb{R}_{\geq 0}$ such that $x_{ij} \equiv_T \pi_j - \pi_i$ for all $ij \in E$

Question

How many vehicles are required to operate the periodic timetable $\pi$ on $\mathcal{E}$?

Example

![Diagram of event-activity network and driving and turnaround activities]

- driving activity $e$ with tension $x_e$
- turnaround activity $e$ with tension $x_e$
- event $v$ with time $\pi_v$
§4.1 Periodic Vehicle Scheduling

Periodic vehicle schedules

Definition

A **periodic vehicle schedule** is a collection $S$ of directed cycles in $\mathcal{E}$ such that each driving activity is contained in at least (exactly) one cycle of $S$. For a periodic vehicle schedule $S$, its **number of vehicles** $\nu(S)$ is given by

$$\nu(S) := \frac{1}{T} \sum_{e \in \mathcal{E}} \sum_{i=1}^{k} \gamma_{i,e} x_{e},$$

where $\gamma_1, \ldots, \gamma_k \in \{0, 1\}^\mathcal{E}$ are the incidence vectors of the cycles in $S$.

Example

- 30 + 30 minutes $\rightarrow$ 6 vehicles
- 50 minutes $\rightarrow$ 5 vehicles

Remark

By the cycle periodicity property of periodic timetabling, $\nu(S) \in \mathbb{Z}_{\geq 0}$. 
4.1 Periodic Vehicle Scheduling

Minimum tension circulation

Definition
The periodic vehicle scheduling problem is to find a periodic vehicle schedule $S$ with minimal $\nu(S)$.

Tension-based integer programming formulation

Minimize

$$\frac{1}{T} \sum_{e \in E} x_e f_e$$

s.t.

$$\sum_{e \in \delta^+(v)} f_e - \sum_{e \in \delta^-(v)} f_e = 0, \quad v \in V,$$

$$f_e \geq 1, \quad e \in E \text{ driving activity},$$

$$f_e \in \mathbb{Z}_{\geq 0}, \quad e \in E.$$

Observation
The periodic vehicle scheduling problem can be formulated as a minimum cost circulation problem.
Minimum offset circulation

Definition (Recall from periodic timetabling)
For an edge $ij \in E$, define its **periodic offset** as

$$p_{ij} := \frac{x_{ij} - \pi_j + \pi_i}{T} \in \mathbb{Z}_{\geq 0}.$$ 

Cycle periodicity property
For all incidence vectors $\gamma$ of oriented cycles in $\mathcal{E}$ holds $\gamma^t x = T \cdot \gamma^t p$.

Offset-based integer programming formulation

Minimize

$$\sum_{e \in E} p_e f_e$$

s.t.

$$\sum_{e \in \delta^+(v)} f_e - \sum_{e \in \delta^-(v)} f_e = 0, \quad v \in V,$$

$$f_e \geq 1, \quad e \in E \text{ driving activity},$$

$$f_e \in \mathbb{Z}_{\geq 0}, \quad e \in E.$$
§4.1 Periodic Vehicle Scheduling

Perfect turnaround matching

Let $E_d$ and $E_t$ be the set of driving and turnaround activities, respectively.

**Theorem**

There is a one-to-one correspondence

$$\left\{ \text{circulations covering all driving activities exactly once} \right\} \leftrightarrow \left\{ \text{perfect matchings in } (V, E_t) \right\},$$

$$(f_e)_{e \in E} \leftrightarrow (f_e)_{e \in E_t}.$$

Moreover, a circulation of cost $c$ w.r.t. $p$ (or $x$) corresponds to a perfect matching of cost $c - \sum_{e \in E_d} p_e$ w.r.t. $p$ (or $c - \sum_{e \in E_d} x_e$ w.r.t. $x$).

**Proof.**

The correspondence has been an exercise. For the cost comparison, note that restricting a circulation to the turnaround activities removes the cost of all driving activities.
§4.1 Periodic Vehicle Scheduling

Perfect turnaround matching: Example

circulation cost w.r.t. $x$: 60

matching cost w.r.t. $x$: 28
driving act. cost w.r.t. $x$: 32

circulation cost w.r.t. $x$: 50

matching cost w.r.t. $x$: 18
driving act. cost w.r.t. $x$: 32
The periodic vehicle scheduling problem has the following interpretations:

- minimum cost circulation w.r.t. periodic tension $x$ covering all driving activities
- minimum cost circulation w.r.t. periodic offset $p$ covering all driving activities
- minimum weight perfect matching w.r.t. periodic tension $x$ of turnaround activities
- minimum weight perfect matching w.r.t. periodic offset $p$ of turnaround activities

The graph $(V, E_t)$ usually decomposes into many small components, so that the perfect matching problem decomposes into smaller problems as well.
§4.2 Aperiodic Vehicle Scheduling

Single-Depot Vehicle Scheduling

Input Data

- set $\mathcal{T}$ of trips $(\tau_{\text{dep}}, \tau_{\text{arr}}) \in \mathbb{R} \times \mathbb{R}$ with $\tau_{\text{dep}} < \tau_{\text{arr}}$
- relation $\preceq$ on $\mathcal{T} \times \mathcal{T}$, where $t_1 \preceq t_2$ holds if a vehicle can use trip $t_2$ after $t_1$

Definition

A vehicle schedule is a collection $S = \{s_1, \ldots, s_k\}$ of chains $s_i = t_{i,1} \preceq t_{i,2} \preceq \cdots \preceq t_{i,r_i}$ such that each trip in $\mathcal{T}$ occurs in at least (exactly) one chain $s_i$. The number $\nu(S) := k$ is the number of vehicles of $S$.

Definition

The (single-depot) vehicle scheduling problem is to find a vehicle schedule $S$ minimizing $\nu(S)$. 
Network flow model

Build an event-activity network $\mathcal{E}$ as follows:

1. Create two events $p$ and $q$.
2. Create activities $(p, d_t), (d_t, a_t), (a_t, q)$ for each trip $t \in \mathcal{T}$ (pull-out, driving, pull-in).
3. For each pair $t_1 \preceq t_2$, add an activity $(a_{t_1}, d_{t_2})$ (turnaround).

The events $p$ and $q$ are depot vertices.

Example
§4.2 Aperiodic Vehicle Scheduling

Network flow model

Build an event-activity network $\mathcal{E}$ as follows:

1. Create two events $p$ and $q$.
2. Create activities $(p, d_t), (d_t, a_t), (a_t, q)$ for each trip $t \in \mathcal{T}$ (pull-out, driving, pull-in).
3. For each pair $t_1 \leq t_2$, add an activity $(a_{t_1}, d_{t_2})$ (turnaround).

The events $p$ and $q$ are depot vertices.

Example
§4.2 Aperiodic Vehicle Scheduling

**Network flow model**

Build an event-activity network \( \mathcal{E} \) as follows:

1. Create two events \( p \) and \( q \).
2. Create activities \((p, d_t), (d_t, a_t), (a_t, q)\) for each trip \( t \in \mathcal{T} \) (*pull-out, driving, pull-in*).
3. For each pair \( t_1 \leq t_2 \), add an activity \((a_{t_1}, d_{t_2})\) (*turnaround*).

The events \( p \) and \( q \) are *depot* vertices.

**Example**
Network flow model

Observation
The single-depot vehicle scheduling problem is solved by finding a minimum value $p$-$q$-flow on $\mathcal{E} = (V, E)$ covering each driving activity at least once:

Minimize

$$\sum_{e \in \delta^+(p)} f_e$$

s.t.

$$\sum_{e \in \delta^+(v)} f_e - \sum_{e \in \delta^-(v)} f_e = 0, \quad v \in V \setminus \{p, q\},$$

$$f_e \geq 1, \quad e \in E \text{ driving activity},$$

$$f_e \in \mathbb{Z}_{\geq 0}, \quad e \in E.$$
This is an optimal $p$-$q$-flow covering each driving activity exactly once. The value of flow (number of vehicles) is 5.

**Notation**
Let $E_d$ and $E_t$ denote the set of driving and turnaround activities, respectively.
Matching interpretation

Lemma
The following numbers are equal:

(a) The minimum value of an $p$-$q$-flow covering each $e \in E_d$ exactly once.

(b) $|E_d| - |M|$, where $M$ is a maximum cardinality matching of $(V, E_t)$.

Proof.
A feasible flow with value $\nu$ decomposes into $\nu$ edge-disjoint $p$-$q$-paths, where the activity types along each path have the pattern (pull-out, driving, turnaround, driving, turnaround, . . . , driving, pull-in). So each path with $k$ driving activities uses $k - 1$ turnaround activities. Summing over all paths yields a matching $M$ of the turnaround activities with $|M| = |E_d| - \nu$.

Conversely, let $M$ be a matching of $(V, E_t)$. Consider the flow of value $|E_d|$ using the $|E_d|$ paths $(p, d_t, a_t, q)$ for all trips $t$. For each edge in $M$, connect the corresponding trips, thereby reducing the flow value by 1. Repeating this process yields a feasible flow of value $|E_d| - |M|$.

June 18, 2018
In \((V, E_t)\), only the 7 arrival vertices \(a_1, a_2, a_4, a_5, a_7, a_8, a_9, a_{10}\) are non-isolated. All of these vertices are matched, so that we obtain a maximum cardinality (in fact, even perfect) matching. As there are 12 driving activities, the minimal number of vehicles equals \(12 - 7 = 5\).
In $(V, E_t)$, only the 7 arrival vertices $a_1, a_2, a_4, a_5, a_7, a_8, a_9, a_{10}$ are non-isolated. All of these vertices are matched, so that we obtain a maximum cardinality (in fact, even perfect) matching. As there are 12 driving activities, the minimal number of vehicles equals $12 - 7 = 5$. 
4.2 Aperiodic Vehicle Scheduling

Single-depot case: Summary

Summary

The single-depot vehicle scheduling can be solved by computing a . . .

- minimum value network flow covering all driving activities
- maximum cardinality matching of the turnaround activities

Remarks

- The actual timetable and the actual travel times are not important, only the feasible sequences of trips matter.
- When costs for trips or turnarounds come into play, this generalizes to a minimum cost network flow or a weighted matching problem.
4.2 Aperiodic Vehicle Scheduling

Multi-Depot Vehicle Scheduling

Input Data

- set $\mathcal{T}$ of trips $(\tau_{\text{dep}}, \tau_{\text{arr}}) \in \mathbb{R} \times \mathbb{R}$ with $\tau_{\text{dep}} < \tau_{\text{arr}}$
- relation $\preceq$ on $\mathcal{T} \times \mathcal{T}$, where $t_1 \preceq t_2$ holds if a vehicle can use trip $t_2$ after $t_1$
- set $\mathcal{D}$ of depots
- assignment $D : \mathcal{T} \rightarrow \mathcal{P}(\mathcal{D})$ of feasible depots for each trip

Definition

The multi-depot vehicle scheduling problem is to find a vehicle schedule $S$ minimizing $\nu(S)$ such that for each chain $t_1 \preceq \cdots \preceq t_r$ in $S$ holds $D(t_1) \cap \cdots \cap D(t_r) \neq \emptyset$.

I.e., the trips served by a vehicle must be feasible for a common depot. In particular, we can assume that the pull-out and pull-in depots of each vehicle are the same.
4.2 Aperiodic Vehicle Scheduling

Network flow model

Build an event-activity network $\mathcal{E}$ as follows:

1. Create two depot vertices $p_d$ and $q_d$ for each depot $d \in D$.
2. Add driving activities $(d_t, a_t)$ for each trip $t \in T$.
3. Add pull-out activities $(p_d, d_t)$ for each trip $t \in T$ and each $d \in D(t)$.
4. Add pull-in activities $(a_t, q_d)$ for each trip $t \in T$ and each $d \in D(t)$.
5. For each pair $t_1 \preceq t_2$, add a turnaround activity $(a_{t_1}, d_{t_2})$.

Example (2 depots)
Network flow model

Build an event-activity network $\mathcal{E}$ as follows:

1. Create two depot vertices $p_d$ and $q_d$ for each depot $d \in \mathcal{D}$.
2. Add driving activities $(d_t, a_t)$ for each trip $t \in \mathcal{T}$.
3. Add pull-out activities $(p_d, d_t)$ for each trip $t \in \mathcal{T}$ and each $d \in D(t)$.
4. Add pull-in activities $(a_t, q_d)$ for each trip $t \in \mathcal{T}$ and each $d \in D(t)$.
5. For each pair $t_1 \preceq t_2$, add a turnaround activity $(a_{t_1}, d_{t_2})$.

Example (2 depots)
4.2 Aperiodic Vehicle Scheduling

Network flow model

Build an event-activity network $\mathcal{E}$ as follows:

1. Create two depot vertices $p_d$ and $q_d$ for each depot $d \in \mathcal{D}$.
2. Add driving activities $(d_t, a_t)$ for each trip $t \in \mathcal{T}$.
3. Add pull-out activities $(p_d, d_t)$ for each trip $t \in \mathcal{T}$ and each $d \in D(t)$.
4. Add pull-in activities $(a_t, q_d)$ for each trip $t \in \mathcal{T}$ and each $d \in D(t)$.
5. For each pair $t_1 \leq t_2$, add a turnaround activity $(a_{t_1}, d_{t_2})$.

Example (2 depots)
Observations

- This is not a normal network flow problem: In a flow with the $p_d$ as sources and the $q_d$ as sinks, a vehicle might pull out from depot 1 and pull in to depot 2.
- Instead, our flow covering all driving activities needs to decompose into $p_d$-$q_d$-flows for each depot $d \in D$.
- This leads to a *multi-commodity flow*. 
§4.2 Aperiodic Vehicle Scheduling

**Multi-commodity flow model**

Minimize

\[
\sum_{d \in D} \sum_{e \in \delta^+(p_d)} f^d_e
\]

s.t.

\[
\sum_{e \in \delta^+(v)} f^d_e - \sum_{e \in \delta^-(v)} f^d_e = 0, \quad d \in D, v \in V \setminus \{p_d, q_d\},
\]

\[
\sum_{d \in D(t)} f^d_e = 1, \quad e \in E \text{ driving activity of trip } t,
\]

\[
\sum_{d \not\in D(t)} f^d_e = 0, \quad e \in E \text{ driving activity of trip } t,
\]

\[
f^d_e \in \{0, 1\}, \quad d \in D, e \in E.
\]

This defines a \(p_d\)-\(q_d\) flow \(f^d\) for each depot \(d \in D\). Each driving activity is covered by exactly one such \(f^d\), and \(d\) is feasible for the corresponding trip.
This optimal 2-commodity flow decomposes into 3 \( p_1-q_1 \)-paths and 3 \( p_2-q_2 \)-paths (→ 6 vehicles required).
4.2 Aperiodic Vehicle Scheduling

General multi-commodity flow

Let $G = (V, E)$ be a digraph with cost functions $c^1, \ldots, c^k : E \to \mathbb{R}$, balances $b^1, \ldots, b^k : V \to \mathbb{R}$, and capacities $u, u^1, \ldots, u^k : E \to \mathbb{R}_{\geq 0}$. The problem

Minimize $\sum_{i=1}^{k} \sum_{e \in E} c^i_e f^i_e$

s.t. $\sum_{e \in \delta^+(v)} f^i_e - \sum_{e \in \delta^-(v)} f^i_e = b^i_v$, $i = 1, \ldots, k, v \in V$,

$\sum_{i=1}^{k} f^i_e \leq u_e$, $e \in E$,

$f^i_e \in \{0, 1, \ldots, u^i_e\}$, $i = 1, \ldots, k, e \in E$.

is called an integer minimum cost $k$-commodity flow problem.
Remarks

- When the flows $f_i$ can be relaxed to rational numbers in $[0, u^i]$, then there are polynomial-time algorithms (linear programming).
- In fact, for rational $f_i$, there are strongly polynomial-time algorithms. I.e., the running time does not depend on cost, balance or capacities.
- However, for $k \geq 2$ commodities, the total unimodularity property of single-commodity flows gets lost. In particular, we cannot use linear programming to obtain integer flows.
- There are non-integral minimum cost 2-commodity flows with integer costs, balances and capacities.
- Finding an integer $k$-commodity flow is NP-hard for every fixed $k \geq 2$ (Even/Itai/Shamir 1974: SAT $\leq$ integer 2-commodity flow).
§4.2 Aperiodic Vehicle Scheduling

Path-based multi-commodity flow/Set partitioning

---

Observation

A feasible multi-commodity flow consists certain $p_d$-$q_d$-paths. Let $\mathcal{P}_d$ denote the set of all $p_d$-$q_d$-paths for a depot $d$. Then

- Every driving activity of some trip $t$ must be covered by exactly one path $p \in \mathcal{P}_d$ with $d \in D(t)$.
- We want to minimize the number of required paths.
§4.2 Aperiodic Vehicle Scheduling

**Path-based multi-commodity flow model**

For a driving activity $e$ of trip $t$, let $P_e := \bigcup_{d \in D(t)} \{ p \in P_d \mid e \in p \}$ denote the set of $p_d$-$q_d$-paths using $e$ coming from a feasible depot $d \in D(t)$ for $t$.

**Integer program**

$$\text{Minimize } \sum_{d \in D} \sum_{p \in P_d} f_p$$

s.t. $\sum_{p \in P_e} f_p = 1, \quad e \in E$ driving activity of trip $t$,

$$f_p \in \{0, 1\}, \quad p \in \bigcup_{d \in D} P_d.$$

**Remarks**

- Any multi-commodity flow problem has a path-based formulation.
- The number of paths is enormous. $\rightarrow$ column generation (pricing: shortest path problems).
4.2 Aperiodic Vehicle Scheduling

Extensions

- depot capacities $\kappa_d$: In the path-based multi-commodity flow formulation, these are modeled as

$$\sum_{p \in \mathcal{P}_d} f_p \leq \kappa_d, \quad d \in \mathcal{D}.$$

- operational costs (e.g., fuel)
- fixed costs (e.g., maintenance, investment)
- multiple vehicle types: one commodity for each feasible combination of a depot and a vehicle type
- time windows: regular trips (e.g., according to line frequency) vs. irregular trips (e.g., school trips)
- route constraints (e.g., battery vehicles)
## Real-world examples

<table>
<thead>
<tr>
<th></th>
<th>Berlin (BVG)</th>
<th>Hamburg (HHA)</th>
<th>Hamburg (VHH)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depots</td>
<td>10</td>
<td>14</td>
<td>10</td>
</tr>
<tr>
<td>Vehicle types</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Combinations</td>
<td>44</td>
<td>40</td>
<td>19</td>
</tr>
<tr>
<td>Trips</td>
<td>25,000</td>
<td>16,000</td>
<td>5,500</td>
</tr>
<tr>
<td>Turnarounds</td>
<td>70,000,000</td>
<td>15,100,000</td>
<td>10,000,000</td>
</tr>
</tbody>
</table>

Löbel, Optimal Vehicle Scheduling in Public Transit, 1997
Chapter 4

Vehicle Scheduling

§4.3 Railway Stock Rotation Planning
Find an optimal periodic vehicle schedule for a standard week (rotation).

A vehicle configuration is a multiset of vehicles.

For each trip, there is a feasible set of vehicle configurations.

Before or after a trip, a vehicle configuration can be changed by coupling.

A train is a set of at most seven trips haven the same departure and arrival stops and the same departure and arrival times, but on different days.
§4.3 Railway Stock Rotation Planning

Hypergraph model

Define a hypergraph $G = (V, H, A)$ as follows:

- A node $v \in V$ is a tuple $(t, c, f, m)$, where $t$ is a trip, $c$ is a feasible vehicle configuration for $t$, and $f$ is a vehicle used $m$ times by $c$.

- A hypernode $h \in H$ is a collection $V(t, c)$ of all nodes $(t, c, f, m)$ for a given trip $t$ and a given configuration $c$.

- A hyperarc $a \in A$ is a non-empty set of pairs $(v, w) \in V \times V$, constructed as follows:
  - Configuration conserving arcs: If a vehicle can go from trip $t_1$ to trip $t_2$ with the same configuration, then add an hyperarc consisting of $|V(t_1, c)| = |V(t_2, c)|$ arcs connecting them.
  - Coupling arcs: Connect trips with different configurations.
  - Regularity hyperarcs: Let $T_1, T_2$ be trains, let $c$ be a configuration and let $o \in \{0, \ldots, 6\}$. Let $a$ be the set of all arcs connecting any trip from $T_1$ with any trip of $T_2$ with configuration $c$ such that midnight is passed $o$ times between the arrival of $t_1$ and the departure of $t_2$. If $|a| \geq 2$, then add a hyperarc $\{ (v, w) \in V \times V | \exists a \in a : (v, w) \in a \}$. 
§4.3 Railway Stock Rotation Planning

Hypergraph model: Conservation and coupling

Borndörfer et. al., A Hypergraph Model for Railway Vehicle Rotation Planning, 2011
§4.3 Railway Stock Rotation Planning

Hypergraph model: Regularity

Borndörfer et. al., A Hypergraph Model for Railway Vehicle Rotation Planning, 2011
§4.3 Railway Stock Rotation Planning

Hypergraph model: Torus

ICE-A network, HyDraw output