Calibration of Microscopic Traffic Simulation in an Urban Environment Using GPS-Data

SUMO User Conference 2024
Christopher Stang, Klaus Bogenberger
Agenda

1. Introduction
2. Network
3. Methodology
4. Results
5. Conclusion & Outlook
Introduction
Introduction

Accurate traffic models are important for well-founded traffic engineering: Usage of traffic count and speed measurements of road segments common approach for the calibration of traffic simulation models.

SUMO offers the tools flowrouter and routesampler for generating traffic demand models on the base of traffic count measurements.

Following approach applies a two-step optimization process by using collected GPS-data with information about vehicle count and speed measurements.

- **A priori optimization**
  - of count measurements by adopting Integer Linear Programming

- **A posteriori optimization**
  - of speed (and count) measurements by adopting Integer Linear Programming+Evolutionary Algorithm
Network
Network

Information about dataset provided by TomTom

Network
- City: Friedrichshafen
- Track: Henri-Dunant-Strasse to Löwentaler-Strasse (ca. 3 km)

Time period
2017-2019, 36 months

Kind of information
- Vehicle counts
- Vehicle speed distribution for each detector

Aggregation of dataset for each detector
- One representative working and weekend day of each month
- Each day subdivided into time intervals of 2 hours

What is the best approach to set up a realistic traffic simulation with the existing dataset?

Source: OpenStreetMap. [Map section Friedrichshafen](https://www.openstreetmap.org/#map=15/47.6602/9.4736 (accessed February 13, 2024))

100 "detectors" distributed along the track
Methodology
Methodology: Workflow

1. Set up of network for SUMO simulation
2. Generating random routes with randomtrips.py
3. A priori optimization of vehicle counts for each detector
4. A posteriori optimization of vehicle counts and speed for each detector
5. Comparison of simulated results with real traffic data
Methodology: A priori optimization

Problem Formulation: Integer Linear Programming

\[
\begin{align*}
\text{min } & \quad c^T x \\
\text{subject to: } & \\
& \quad b_i \leq Ax \leq b_u \\
& \quad x \geq 0, \quad x_i \in \mathbb{Z} \\
& \quad b_u = \begin{bmatrix} 
\text{counts, } D_1 \\
\text{counts, } D_2 \\
\vdots \\
\text{counts, } D_m 
\end{bmatrix} \\
& \quad b_l = \begin{bmatrix} 
0 \\
0 \\
\vdots \\
0 
\end{bmatrix}
\end{align*}
\]

1. Create Matrix \( A' \) (dim \( A = m \times n \)) with \( m \) detector edges and \( n \) generated routes
2. Run through each column (route) and check if route contains a detector edge
3. If yes, set the corresponding detector edge to 1, otherwise to 0
Methodology: A priori optimization

- Set up of network for SUMO simulation
- Generating random routes with randomtrips.py
- A priori optimization of vehicle counts for each detector
- A posteriori optimization of vehicle counts and speed for each detector
- Comparison of simulated results with real traffic data

Problem Formulation: Integer Linear Programming

\[
\begin{align*}
\sum & \ multiplicator \\
\end{align*}
\]

\[
\begin{align*}
\mathbf{c}^T &= \sum_{i=1}^{m} a_{i1} \cdot x_1 + \ldots + \sum_{i=1}^{m} a_{in} \cdot x_n = \sum_{j=1}^{n} \sum_{i=1}^{m} a_{ij} \cdot x_j \\
\min \ F(x) &= - \sum_{j=1}^{n} \sum_{i=1}^{m} a_{ij} \cdot x_j \\
\end{align*}
\]
Methodology: A priori optimization

- Set up of network for SUMO simulation
- Generating random routes with randomtrips.py
- A priori optimization of vehicle counts for each detector
- A posteriori optimization of vehicle counts and speed for each detector
- Comparison of simulated results with real traffic data

Simulation results
Methodology: A posteriori optimization

Set up of network for SUMO simulation

Generating random routes with randomtrips.py

A priori optimization of vehicle counts for each detector

A posteriori optimization of vehicle counts and speed for each detector

Comparison of simulated results with real traffic data

Problem Formulation: ILP+Evolutionary Algorithm

Empirical data by TomTom

Initial population by ILP

Calculate objective

Check termination criteria

Calibration done

Reinsertion

Mutation

Recombination

Selection

\( x \): Multiplicator of each route

\( v \): Maximum allowed vehicle speed of each detector

MAE Counts

MAE Speed

not fulfilled

fulfilled
Results
Results

Simulation results: A priori optimization

Set up of network for SUMO simulation

Generating random routes with randomtrips.py

A priori optimization of vehicle counts for each detector

A posteriori optimization of vehicle counts and speed for each detector

Comparison of simulated results with real traffic data
Results

Simulation results: A priori optimization

Simulation results: A posteriori optimization

[Graphs showing simulation results for different models and time periods, with metrics such as MAE counts and MAE speed, highlighting improvements by 10% and 21% respectively.]
Conclusion & Outlook
Conclusion & Outlook

Conclusion
Method of two-step optimization for the calibration of traffic simulations by using vehicle count and speed measurements was presented.

Method was implemented and tested in a subnetwork of Friedrichshafen. Method was compared with the SUMO tools flowrouter and routesampler showing better results than these tools.

Outlook
Method of a posteriori optimization will be extended to a larger time frame.

Method offers the possibility for traffic-based testing for AD/ADAS-development.
Thank You.
Appendix
## Methodology: Overview of methods

<table>
<thead>
<tr>
<th></th>
<th>Flowrouter</th>
<th>Routesampler</th>
<th>ILP approach</th>
<th>ILP+EA approach</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong></td>
<td>Network</td>
<td>Network</td>
<td>Network</td>
<td>Network</td>
</tr>
<tr>
<td></td>
<td>Edge based count data</td>
<td>Edge based count data</td>
<td>Edge based count data</td>
<td>Edge based count data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Initial set of routes</td>
<td>Initial set of routes</td>
<td>Initial set of routes</td>
</tr>
<tr>
<td><strong>Optimization method</strong></td>
<td>Maximum flow problem</td>
<td>Linear Programming</td>
<td>Integer Linear Programming</td>
<td>Integer Linear Programming</td>
</tr>
<tr>
<td></td>
<td>A priori optimization</td>
<td>A priori optimization</td>
<td>A priori optimization</td>
<td>Evolutionary Algorithm</td>
</tr>
<tr>
<td><strong>Optimization objective</strong></td>
<td>Count data</td>
<td>Count data</td>
<td>Count data</td>
<td>Speed data</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>Route-file</td>
<td>Route-file</td>
<td>Route-file</td>
<td>Route-file</td>
</tr>
<tr>
<td></td>
<td>Flow-file</td>
<td></td>
<td></td>
<td>Speed-file</td>
</tr>
</tbody>
</table>

Input: Network, Edge based count data, Initial set of routes.

Optimization methods: Maximum flow problem, Linear Programming, Integer Linear Programming, Evolutionary Algorithm.

Optimization objectives: Count data, Count data, Count data, Speed data.