An iterative solution approach for a bi-level optimization problem for congestion avoidance on road networks

Andreas Britzelmeier, Alberto De Marchi and Matthias Gerdts

Abstract The paper introduces an iterative solution algorithm for a bi-level optimization problem arising in traffic control. The bi-level problem consists of a shortest path problem on the upper level, which aims to minimize the total path length of a set of cars in a road network. The cost coefficients in the shortest path problem depend on the solutions of lower level optimal control problems taking into account congestion, while the lower level problems depend on the paths. This leads to a strong coupling between upper level problem and lower level problem. This coupling is decomposed by an iterative procedure fixing either the costs or the paths in the upper level and the lower level, respectively. Numerical experiments illustrate the procedure and indicate that the iterative algorithm leads to suitable distribution of cars in the network.

1 Introduction

Increasing traffic loads due to a steadily increasing population and rising commerce, poses a problem especially to urban areas. Neverthless the economical aspect of CO_2 polution is an imminent threat to the health of humans. Reducing traffic seems to be the main idea to solve these problems. However, banning cars from cities or cramming people into public transportations, seems not to be an attractive and productive solution. A different approach would be to reduce the total time a car needs to reach its destination in the sense that the flux of cars is optimized. Considering the introduction of automatic or autonomous cars, this could be achieved by controlling the cars such that their paths and velocity is optimized with respect to avoid traffic jams. In this paper we propose a bi-level optimal control problem and an iterative

Andreas Britzelmeier, Alberto De Marchi, Matthias Gerdts
University of the Federal Armed Forces at Munich, Department of Aerospace Engineering, Werner-Heisenberg-Weg 39, 85577 Neubiberg, Germany, e-mail: andreas.britzelmeier@unibw.de, alberto.demarchi@unibw.de, matthias.gerdts@unibw.de

scheme for optimizing the network-wide traffic flow. The upper level problem controls the overall vehicle distribution with an adaptive shortest path algorithm. The route planning for each car is based on shared costs, derived from coupling single cars behaviour. The lower level is concerned with providing optimal velocity profiles and density updates to the shortest path algorithm, such that speed limits are simulated. [11] proposes two approaches for solving a bi-level optimization problem. Either by treating the lower level problem as a parametric optimization problem, which is solved whenever it is required for the upper level, or by reducing the problem to a single level problem by replacing the lower level through its necessary conditions. A semi-analytic solution approach for minimum time velocity profiles can be found in [1, 7].

The paper is organized as follows. Section 2 provides an overview on the bi-level optimization problem. Sections 3 and 4 formulate and propose numerical methods to solve the upper and lower level optimization problems. Section 5 discusses how the two levels interface with each other and finally in Section 6 we apply the iterative procedure and report numerical results.

2 Problem Formulation and Solution Approach

A road network can be represented by a graph $\mathbb{G} = (V, E)$ consisting of a vertex set V and an edge set E, see [2, 8]. An edge is the topological description of a road segment, and a vertex corresponds to an intersection. Edges have properties like, e.g., length, speed limit, maximum density (i.e. maximum number of vehicles per unit length). A set of cars C move on the graph $\mathbb{G}$, in the sense that cars are initially positioned on a vertex and aim at reaching another vertex by following a suitable path (i.e. a sequence of edges) on the graph $\mathbb{G}$. These agents interact at the microscopic scale, yielding macroscopic effects like congestions, traffic waves and self-organizational phenomena, compare [9, 4, 5].

The problem here is how to plan a route for each and every car, from the initial to the desired point, taking into account traffic jam, driver's behavior, vehicle dynamics and speed limits. Drivers are supposed to aim at the minimum time control of their own car, while obeying speed limits and constraints on the vehicle dynamics. Thus, the overall problem is here formulated as a bi-level optimization problem (BOP), where route planning is represented by the upper level optimization problem and the lower level optimization problem is adopted to predict how drivers will behave, given a certain path. The route planning, also referred to as upper level optimization problem (UL-OP), aims at finding the minimum cost path for each car, given its initial and final position. Instead, the lower level optimization problem (LL-OP) represents an optimal control problem with vehicle and road constraints. These two problems exchange information in the sense that they depend on each other. The UL-OP can be seen as constrained by solutions of the LL-OP, because the cost of each edge depends on the actual traffic jam, in terms of car density, compare [9, 3]. On the other hand, the UL-OP affects the LL-OP, because the minimum time control, and

consequent optimal speed profile, depends on the planned path with corresponding length of edges and speed limits.

Some assumptions and simplifications are adopted throughout the present work: Road geometry is time-invariant; speed limits are considered constant in time and space on each single road segment.

In summary, the traffic control problem results in the following bi-level optimization problem whose details are described in Sections 3 and 4:

The Upper Level Optimization Problem (UL-OP) reads as follows:

$$\begin{aligned} \textit{Minimize} \quad & \sum_{k\in C} c^k (x^k)^\top z^k \\ \textit{with respect to} \quad & (z^k, x^k, v^k, u^k), k \in C, \\ \textit{subject to} \quad & Az^k = b^k, z^k \geq 0, k \in C \\ & (x^k, v^k, u^k) \in \mathcal{M}(z^k), k \in C. \end{aligned} \tag{1}$$

Herein, $\mathcal{M}(z^k)$ denotes the set of minimizers of the following Lower Level Optimization Problem (LL-OP):

$$\begin{aligned} \textit{Minimize} \quad & T \\ \textit{subject to} \quad & \dot{x}(t) = v(t), && \dot{v}(t) = f_k(v(t), u(t)), \\ & x(0) = 0, && x(T) = L_k, \\ & v(0) = v_k^0, && v(t) \in [0, \overline{v}_k(x(t))], \\ & u(t) \in \mathcal{U}_k. \end{aligned} \tag{2}$$

The index k indicates that the corresponding quantities depend on z^k. The function f_k represents the vehicle dynamics, the box $\mathcal{U}_k$ defines control constraints, $\overline{v}_k : \mathbb{R}_+ \to \mathbb{R}_+$ and L_k are speed limits and length of the driving path, respectively.

There are basically a few main techniques for solving bi-level optimization problems. The first approach keeps the bi-level structure and treats the LL-OP as a parametric optimization problem, which is being solved whenever the solution algorithm for the UL-OP requires it [11]. The second technique, instead, is based on the formulation of first order necessary optimality conditions for the LL-OP. Then, the LL-OP is replaced by its necessary conditions, which are considered as constraints in the UL-OP. This reduces the bi-level problem into a single-level nonlinear optimization problem, but in general this is not equivalent to the original problem, since necessary conditions might be not sufficient [11]. A third approach is based on the substitution of the LL-OP with its value function. This generates an equivalent single-level optimization problem.

In this paper, we chose to follow an approach that resembles the first one discussed above, but we treat the two levels as coupled optimization problems, while

iteratively solving one after the other. To this end let $z = (z^k)_{k\in C}$ denote the variables of the upper level problem UL-OP and $x = (x^k)_{k\in C}$, $v = (v^k)_{k\in C}$, $u = (u^k)_{k\in C}$ the variables of the lower level problem LL-OP. We iteratively compute $z_{[j]}$, $x_{[j]}$, $v_{[j]}$, $u_{[j]}$ for $j = 0, 1, 2, \ldots$ as follows:

(0)Choose $x_{[0]}, v_{[0]}, u_{[0]}$ and set $j = 0$.
(1)Compute $c_{[j]} := (c^k(x^k_{[j]}))_{k\in C}$.
(2)Solve the UL-OP with $c_{[j]}$ in (2) and neglecting the last constraint. Let the solution be $z_{[j]}$.
(3)Solve LL-OP for $z_{[j]}$. Let the solution be $x_{[j+1]}$, $v_{[j+1]}$, and $u_{[j+1]}$, i.e. $(x^k_{[j+1]}, v^k_{[j+1]}, u^k_{[j+1]}) \in \mathcal{M}(z^k_{[j]})$ for $k \in C$.
(4)Set $j \leftarrow j + 1$ and go to (1) until a stopping criterion has been reached.

Considering this iterative procedure, the LL-OP and UL-OP are solved the same number of times and the levels are treated as uncoupled problems, just coupled at the interface by the procedure itself. Since we are not yet aware of a formal convergence result for such an iterative scheme, one purpose of this paper is to experimentally investigate if the procedure converges or if oscillations can be observed. Please note that the above bi-level problem is a hard problem and also the alternative second and third solution approaches mentioned before are very difficult to realize numerically owing to non-smoothness issues.

3 Upper Level: Route Planning

Let the road network be described through a directed graph $\mathbb{G} = (V, E, c, s, t)$, with vertices $V = \{1, 2, \ldots, n\}$ and edges E. For simplicity we assume that the vertices are numbered such that the initial vertex is given by $s := 1$ whereas the target vertex is $t := n$. The cost c_{ij} of each edge $(i, j) \in E$ is often associated to the length of the corresponding road segment, such that $c : E \to \mathbb{R}_+$ defines a cost function, see [2, 8]. The shortest path problem for an individual car starting at s and moving to t can be formulated mathematically as follows, compare [10, p. 235]:

$$\textit{Minimize} \quad \sum_{(i,j)\in E} c_{ij} z_{ij} \quad \textit{subject to} \quad Az = e^1, z_{ij} \geq 0, (i, j) \in E,$$

where z_{ij} is the load transported along the edge $(i, j) \in E$, e^1 is the canonical unit vector, and A denotes the node-edge incidence matrix of $\mathbb{G}$. Note that A is a totally unimodular matrix and hence the linear optimization problem posseses a binary solution with $z_{ij} \in \{0, 1\}$ for all $(i, j) \in E$. The shortest path then consists of all edges (i, j) with $z_{ij} = 1$. An efficient implementation for solving the above linear program is based on a primal-dual algorithm as described in, e.g. [10, Sect. 4.4.1], and leads to the famous Dijkstra's algorithm [6] in Algorithm 1. Please note that extensions like the A* algorithm exist. After termination of Algorithm 1 $d(i)$ contains the length of a shortest path from s to i and $p(i)$ contains the predecessor of i on such a shortest path.

Algorithm 1: Dijkstra Algorithm.

Input: Set $W = \{s\}, d(s) = \{0\}$ and $d(i) = \infty$ for all $i \in V \setminus \{s\}$.
forall $i \in V \setminus \{s\}$ *and* $(s, i) \in E$ **do**
  $\longrightarrow$ set $d(i) = c_{1i}, p(i) = s$
while $W \neq V$ **do**
  $\longrightarrow$ find $k \in V \setminus W$ where $d(k) = min\{d(i) : i \in V \setminus W\}$
  $\longrightarrow$ $W = W \cup k$
  forall $i \in V \setminus W$ *with* $(k, i) \in E$ **do**
    if $d(i) > (d(k) + c_{ki})$ **then**
      $\longrightarrow$ $d(i) = d(k) + c_{ki}$
      $\longrightarrow$ $p(i) = k$

Now we are interested in minimizing the total path length, which is obtained by summing up the lengths of all individual shortest paths of the cars in the road network. To this end let $c^k = (c^k_{ij})_{(i,j)\in E} > 0$ denote the cost vector of car $k \in C$, $z^k = (z^k_{ij})_{(i,j)\in E}$ the corresponding path indicator variables, and $b^k = (b^k_i)_{i\in V}$ the unit vector that indicates the starting node of car $k \in C$. With this notation, the task to minimize the total path length for all cars in C yields the following upper level problem UL-OP:

$$\begin{aligned} \textit{Minimize} \quad & \sum_{k\in C} (c^k)^\top z^k = \sum_{k\in C} \sum_{(i,j)\in E} c^k_{ij} z^k_{ij} \\ \textit{subject to} \quad & Az^k = b^k, z^k \geq 0, k \in C. \end{aligned}$$

Please note that UL-OP is a separable optimization problem and its solution can be obtained by solving individual shortest path problems for all cars in C and summing up the lengths.

So far, we assumed that the cost vectors c^k, $k \in C$, are given vectors. This assumption will be dropped in the sequel by taking into account individual trajectories for each car on the shortest paths. To this end, the costs of each edge follow an evolution, depending on the congestion of the roads and therefore on the speed of the vehicles on the same edge e. Thus, the cost vectors will depend on the solution of lower level optimal control problems, which will be discussed in the following Section 4.

4 Lower Level: Minimum Time Driving

We aim to compute time minimal trajectories on a given path in the road network. The vehicle dynamics are described by a second-order time-invariant linear system

for simplicity. We take into account a linear drag force. The validity of this assumption significantly depends on the velocity regime, but it simplifies the derivation of a semi-analytical solution to the LL-OP. There exist results also accounting for both, linear and quadratic drag forces, see [1, 7]. We point out that this simplification is not necessary for the proposed iterative scheme, but it reduces the computational time required for solving the lower level problem LL-OP.

Each individual car minimizes the time required to arrive at the destination subject to acceleration and speed limits. It is noticeable that at this level agents do not interact, in fact, no coupling between cars is present in LL-OP (2). This inaccuracy is more negligible as density gets lower and traffic congestions are avoided. In this section we focus on a single car. Each vehicle is characterized by its mass $m_D > 0$, its linear drag coefficient $c_D \geq 0$, its initial speed $v_0 \geq 0$ and its maximum braking and pushing forces $F_{\text{brake}} \in (-\infty, 0)$ and $F_{\text{push}} \in (0, +\infty)$. Let us introduce the drag parameter $c := c_D/m_D \geq 0$ and control bounds $\underline{u} := F_{\text{brake}}/m_D$ and $\overline{u} := F_{\text{push}}/m_D$. Let a path $p = (p_0, \dots, p_N)$ with vertices $p_j \in V$, $j = 0, \dots, N$, be given. With each edge $e_j = (p_j, p_{j+1})$ on the path we associate a (physical) distance ℓ_j, $j \in \{0, \dots, N-1\}$. The total length of the path is then $L_p = \sum_{j=0}^{N-1} \ell_j$. We assume that a piecewise constant speed limit function $\overline{v} : [0, L_p] \to \mathbb{R}$ is given with $\overline{v}(x) := \overline{v}_j > 0$ for $x \in [a_j, a_j + \ell_j)$, $j \in \{0, \dots, N-1\}$, and $a_j := \sum_{k=0}^{j-1} \ell_k$.

Each vehicle aims at solving the following path minimum-time optimization problem:

$$
\begin{aligned}
&\textit{Minimize} && T && \\
&\textit{subject to} && \dot{x}(t) = v(t), && \dot{v}(t) = u(t) - cv(t), \\
&&& x(0) = 0, && x(T) = L_p, \\
&&& v(0) = v_0, && v(t) \in [0, \overline{v}(x(t))], \\
&&& u(t) \in [\underline{u}, \overline{u}]. &&
\end{aligned}
\tag{3}
$$

Because of its particular structure, mostly the time cost and the edge-wise constant speed limit, it is possible to reduce Problem (3) to an ordered sequence of simpler edge minimum-time optimization problems. These have to be solved starting from the first edge and iterating until the end of path p. Let us consider edge $e = e_j$ with length $L := \ell_j > 0$, speed limit $\overline{v} := \overline{v}_j > 0$ and end-point speed limit $\overline{v}_T := \min\left(\overline{v}_j, \overline{v}_{j+1}\right) > 0$. On edge e we have to solve the following optimal control problem:

$$
\begin{aligned}
&\textit{Minimize} && T && \\
&\textit{subject to} && \dot{x}(t) = v(t), && \dot{v}(t) = u(t) - cv(t), \\
&&& x(0) = 0, && x(T) = L, \\
&&& v(0) = v_0, && v(T) \in [0, \overline{v}_T], \\
&&& v(t) \in [0, \overline{v}], && u(t) \in [\underline{u}, \overline{u}].
\end{aligned}
\tag{4}
$$

Problem (4) resembles the minimum-time optimal control problem subject to velocity constraints and limited acceleration discussed in [1, 7]. However, an additional constraint is present, that is the final speed constraint. In the following we focus on the solution of Problem (4) for the case $v_0 < \overline{v} > \overline{v}_T$, which is the most crucial case. An analogous derivation for the other cases is straightforward.

As suggested in [1], let us introduce the following auxiliary functions, both for numerical stability and notational clarity:

$$\mathcal{E}(t,w) := \frac{1-e^{wt}}{w}, \qquad \mathcal{E}_2(t,w) := \frac{e^{wt}-1-wt}{w^2}. \tag{5}$$

Then, analogously to [1, 7], we claim there exist two distinct time instants, denoted τ_1 and τ_2 and such that $0 < \tau_1 < \tau_2 < T$, that are switching times for the optimal control, whose expression reads

$$u(t) = \begin{cases} \overline{u}, & 0 < t < \tau_1, \\ c\overline{v}, & \tau_1 < t < \tau_2, \\ \underline{u}, & \tau_2 < t < T, \end{cases} \tag{6}$$

for a.e. $t \in [0,T]$. We like to emphasize that $u : [0,T] \to \mathbb{R}$ is uniquely identified by switching times and final time T. The optimal control (6) consists of an initial pushing phase, up to the maximum allowed speed, a second phase where speed is kept constant at the speed limit until the final braking phase. The structure of (6) resembles a bang-bang control, but it shows an intermediate phase due to the velocity constraint. Problem (4) is transformed into a boundary value problem (BVP) collecting optimal control (6) and differential-algebraic constraints in (4). The unknowns of this BVP are switching times τ_1 and τ_2 and final time T. We remark that Problem (4) and the BVP are equivalent if and only if control (6) locally minimizes the Hamiltonian function of Problem (4), as claimed above (the proof is left to the reader). By considering the vehicle model and initial conditions in (4) along with the optimal control (6), it is possible to compute the time evolution of vehicle velocity and position, for $t \in [0,T]$, i.e.

$$v(t) = \begin{cases} v_0 e^{-ct} + \overline{u}\mathcal{E}(-t,c), & 0 \le t \le \tau_1, \\ v(\tau_1^-), & \tau_1 \le t \le \tau_2, \\ v(\tau_2^-)e^{-c(t-\tau_2)} + \underline{u}\mathcal{E}(\tau_2 - t, c), & \tau_2 \le t \le T, \end{cases} \tag{7}$$

$$x(t) = \begin{cases} x_0 + v_0\mathcal{E}(-t,c) + \overline{u}\mathcal{E}_2(-t,c), & 0 \le t \le \tau_1, \\ x(\tau_1^-) + v(\tau_1^-)(t-\tau_1), & \tau_1 \le t \le \tau_2, \\ x(\tau_1^-) + v(\tau_1^-)(\tau_2-\tau_1) + v(\tau_1^-)\mathcal{E}(\tau_2 - t, c) + \underline{u}\mathcal{E}_2(\tau_2 - t, c), & \tau_2 \le t \le T. \end{cases} \tag{8}$$

The analytical solution of this Cauchy problem greatly simplifies the solution of the aforementioned BVP, transforming it into an equivalent non-linear system. This task can be achieved by enforcing boundary conditions and state constraints in (4)

to speed profile and trajectory (7)-(8). In particular, the following conditions must be satisfied by the solution of Problem (4):

$$v(\tau_1) = \overline{v}, \qquad v(T) = \overline{v}_T, \qquad x(T) = L. \tag{9}$$

The first makes the pushing phase to stop when the speed limit is reached; similarly, the second constraint means that, at the final time T, the vehicle speed has to be as high as possible, otherwise it would not be a minimum-time speed profile. Finally, the third condition ensures that the final position is reached at the final time T. Conditions (9) can be rewritten by using (7)-(8), yielding the non-linear system $\varphi(z) = 0$, where $z = (\tau_1, \delta, T)^\top$, $\delta := T - \tau_2$, and $\varphi : \mathbb{R}^3 \to \mathbb{R}^3$ is defined by

$$\varphi(z) := \begin{pmatrix} v_0 e^{-c\tau_1} + \overline{u}\mathcal{E}(-\tau_1, c) - \overline{v} \\ \overline{v}e^{-c\delta} + \underline{u}\mathcal{E}(-\delta, c) - \overline{v}_T \\ x_0 + v_0\mathcal{E}(-\tau_1, c) + \overline{u}\mathcal{E}_2(-\tau_1, c) + \overline{v}(T - \delta - \tau_1) + \overline{v}\mathcal{E}(-\delta, c) + \underline{u}\mathcal{E}_2(-\delta, c) - L \end{pmatrix}. \tag{10}$$

It is possible to explicitly write the Jacobian φ' and then to take advantage of it by using Newton-type solvers to find $z^\star$ such that $\varphi(z^\star) = 0$, where

$$\varphi'(z) = \begin{bmatrix} (\overline{u} - cv_0)e^{-c\tau_1} & 0 & 0 \\ 0 & (\underline{u} - c\overline{v})e^{-c\delta} & 0 \\ v_0 e^{-c\tau_1} - \overline{v} + \overline{u}\mathcal{E}(-\tau_1, c) & \overline{v}(e^{-c\delta} - 1) + \underline{u}\mathcal{E}(-\delta, c) & \overline{v} \end{bmatrix}. \tag{11}$$

Non-linear solvers typically require an initial guess. A reasonable and easy-to-compute initial guess can be estimated by considering the limit $c \to 0^+$; in fact, typically the parameter c is small. Let us define $\varphi_0 : \mathbb{R}^3 \to \mathbb{R}^3$, such that $\varphi_0(z) := \lim_{c\to 0^+} \varphi(z)$ for any $z \in \mathbb{R}^3$. Then, a reasonable initial guess is given by the solution of $\varphi_0(z^\star) = 0$, that is

$$z^\star = \left(\frac{\overline{v} - v_0}{\overline{u}}, \ \frac{\overline{v}_T - \overline{v}}{\underline{u}}, \ \frac{L - x_0}{\overline{v}} + \frac{(\overline{v} - v_0)^2}{2\,\overline{v}\,\overline{u}} - \frac{(\overline{v}_T - \overline{v})^2}{2\,\overline{v}\,\underline{u}} \right)^\top. \tag{12}$$

The following Section 5 describes how the lower level optimal control problems are coupled with the upper level shortest path problem in Section 3.

5 Levels Coupling

The interface between levels, namely UL-OP and LL-OP, plays a key role in the solution process of the bi-level optimization problem. In fact, this crucially affects the exchange of information among levels.

Considering the k-th car, the information flow from UL-OP to LL-OP consists of the ordered sequence of edge lengths and speed limits uniquely identified by the planned path p^k, that is the solution of UL-OP. These values constrain the LL-OP, both as boundary conditions and state constraints.

On the other hand, given optimal speed profiles $v^k(\cdot)$ and a trajectories $x^k(\cdot)$ for every car $k \in C$, an edge cost c^k, compare Section 3, has to be defined, based on an estimate of travel time, accounting for possible traffic jam and driver's behavior. Given the solutions to LL-OP for every car, one can reconstruct the number of cars $n_e(t)$ in any edge $e \in E$ as a function of time, $n_e : \mathbb{R}_+ \to \mathbb{R}_+$ with

$$n_e(t) := \text{card}\{k \mid x^k(t) \in e\} \tag{13}$$

(herein, we identified the edge e with its physical distance range for notational simplicity). For any edge $e \in E$, having length $L_e > 0$ and speed limit $\overline{v}_e > 0$, the edge density function $\rho_e : \mathbb{R}_+ \to \mathbb{R}_+$ is defined, such that $\rho_e(t) := n_e(t)/L_e$ for any t. Inspired by the LWR model in [9], that is a first-order PDE-based macroscopic model widely used for traffic flow, let us introduce also the edge speed function $v_e : \mathbb{R}_+ \to \mathbb{R}_+$, such that

$$v_e(t) := \overline{v}_e \left(1 - \frac{\rho_e(t)}{\overline{\rho}_e}\right) \tag{14}$$

for any t, where $\overline{\rho}_e > 0$ is the maximum edge density. Note that the edge speed v_e does not reflect vehicles speed along this edge, but it is just an estimate accounting for traffic jam (v_e is a non-increasing function of n_e and ρ_e). We notice also that for $n_e(t) = 1$, using Eq. (14), the edge speed $v_e(t)$ is lower than the speed limit $\overline{v}_e$, which is not what we want to achieve. One possible way to fix this inaccuracy is to replace n_e with $\max(n_e - 1, 0)$, in order to make the driver not to interact with itself.

As an edge cost we consider an estimate of the time needed to run across the edge itself. To evaluate this time duration, a representative edge speed value is needed, here denoted by $\hat{v}_e$ and chosen to be

$$\hat{v}_e := (1 - \theta)\frac{1}{T_\text{h}} \int_0^{T_\text{h}} v_e(t)\, \mathrm{d}t + \theta \min_{t \in [0, T_\text{h}]} v_e(t) \tag{15}$$

given hyper-parameter $\theta \in [0, 1]$ and time horizon $T_\text{h} > 0$. With this definition it always holds

$$0 \leq \min_{t \in [0, T_\text{h}]} v_e(t) \leq \hat{v}_e \leq \frac{1}{T_\text{h}} \int_0^{T_\text{h}} v_e(t)\, \mathrm{d}t \leq \overline{v}_e$$

for any edge speed function v_e, in any edge $e \in E$. The hyper-parameter θ has been introduced to estimate an edge speed $\hat{v}_e$ representative of the predicted evolution of vehicle trajectories and their interactions. Note that this estimate may be really rough and in general it leads to sub-optimal solutions, especially when long edges are present.

The edge cost c_e, for $e \in E$, is expressed in terms of the time needed to travel along edge e, based on estimate $\hat{v}_e$. This cost is defined as the minimum time run, plus an augmentation of the traffic-related time, to possibly give more importance to congestions, through a parameter $\lambda \geq 0$:

$$c_e := \frac{L_e}{\overline{v}_e} + \lambda \left(\frac{L_e}{\hat{v}_e} - \frac{L_e}{\overline{v}_e}\right) \tag{16}$$

Using (16) in the shortest path problem in Section 3 leads to a nonlinear coupling with the lower level problem LL-OP in Section 4. This coupling acts in both directions and the resulting bi-level optimization problem is very hard to solve in general. As a first approach towards its solution we propose the iterative procedure in Section 2, which results in the following Algorithm 2.

Algorithm 2: Iterative procedure as a method to solve BOP.

Input: Road network $\mathbb{G} = (V, E, c, s, t)$, with cars position, speed and target, $\{s_j, v_j^0, t_j\}_{j\in C}$, parameters $\{c_j, \underline{u}_j, \overline{u}_j\}_{j\in C}$, hyper-parameters $\theta \in [0,1]$, $\lambda \geq 0$.

$k \leftarrow 0$;
for $e \leftarrow E$ **do**
  $c_e^k \leftarrow L_e/\overline{v}_e$; // edge cost initialization
while *not converged* **do**
  for $j \leftarrow C$ **do**
    $p_j^k \leftarrow \texttt{shortestPath}\left(\{c_e^k\}_{e\in E}, s_j, t_j\right)$; // UL-OP
  for $j \leftarrow C$ **do**
    $l_j^k \leftarrow \left\{L_e \mid e \in p_j^k\right\}$; // upper → lower
    $\overline{v}_j^k \leftarrow \left\{\overline{v}_e \mid e \in p_j^k\right\}$;
    $\left(x_j^k, v_j^k, u_j^k\right) \leftarrow \texttt{minTime}\left(l_j^k, \overline{v}_j^k, v_j^0, c_j, \underline{u}_j, \overline{u}_j\right)$; // LL-OP
  for $e \leftarrow E$ **do**
    $c_e^{k+1} \leftarrow \texttt{edgeCost}\left(\{x_j^k\}_{j\in C}, \theta, \lambda\right)$; // lower → upper
  $k \leftarrow k + 1$;

Numerical experiments are documented in the following Section 6.

6 Numerical Results

In the previous Sections we presented the algorithms for solving the upper and lower level of the proposed bi-level optimization problem, the coupling of those levels, especially the cost function, was discussed in Section 5.
First we want to test the overall functionality of the proposed iterative bi-level algorithm. Thereafter, regarding the proposed parameters θ and λ in the cost function, which implies the connection from the lower to the upper level, we want to analyze the impingement of these parameters on the numerical results as well as the convergence. Therefore we vary one parameter while fixing the other one and vice versa. Finally we take a closer look at the behaviour of a single car.

6.1 General evaluation of the Bi-level Algorithm

The algorithms discussed above are implemented in a MATLAB program. For a first test we set the number of cars $n_c = 500$, $\theta = 0.5$ and $\lambda = 1000$, the drag is neglected ($c = 0$). The road network (Fig.1) is randomly generated on a 2000×2000

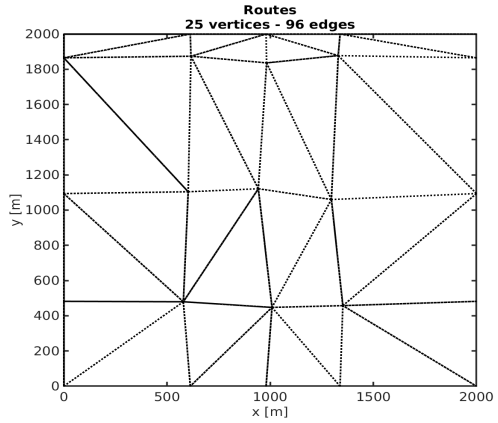
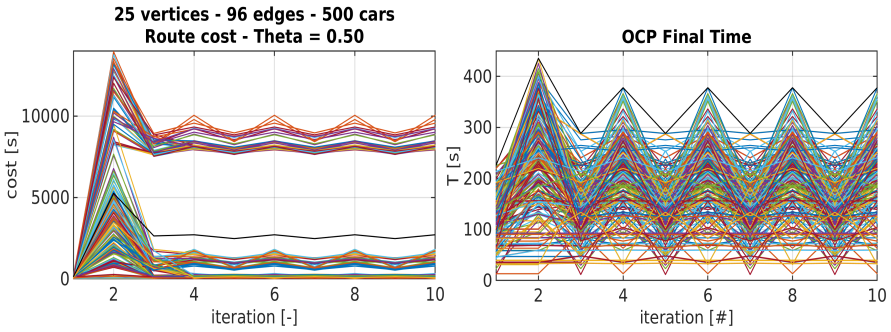


Fig. 1: Randomly generated road network, connections through Delaunay triangulation.

[m] grid, the connections between the chosen gridpoints are derived through applying a Delaunay triangulation. The limits on the acceleration for the LL-OP is set to $u \in [-3, 2]$ [m/s^2], the maximum velocity therefore is chosen randomly for each car from a set $[10, 20]$ $[m/s]$, as well as the initial speed $v_0 \in [6, 10]$ $[m/s]$, and the number of iterations $N_{iter} = 10$. Fig. 2 shows the result of the bi-level algorithm as in the behaviour of the cost function and the evolution of the final time of every car. Considering the cost function, due to the weighing with λ, the meaning of the values is negligible. Nevertheless we notice a reduction in the cost for every car during the first 3 steps. The algorithm converges to different optimal solutions for sets of cars with the same costs. This can be explained such that to avoid congestions the algorithm distributes the cars over the road network with respect to keeping the costs low. This leads to sets of vehicles with the same minimal cost to pass from start to their destination. However we also notice that there remains an oscillating behaviour, which seems to resemble two equally good solutions regarding the overall distribution of the vehicles. One solution however yields higher costs. This oscillating characteristic is also mirrored in the final time. In the first three steps the final time decreases. After that, the jumping between two solutions occurs.
Concluding, the algorithm finds optimal paths as well as velocity profiles for every car, while avoiding congestions, through consideration of the vehicle density



(a) Evolution of cost function for every car, $\theta = 0.5$ and $\lambda = 1000$.

(b) of final time for every car, $\theta = 0.5$ and $\lambda = 1000$.

Fig. 2: Numerical results for the cost function and final time over 10 iterations, $n_c = 500$ cars

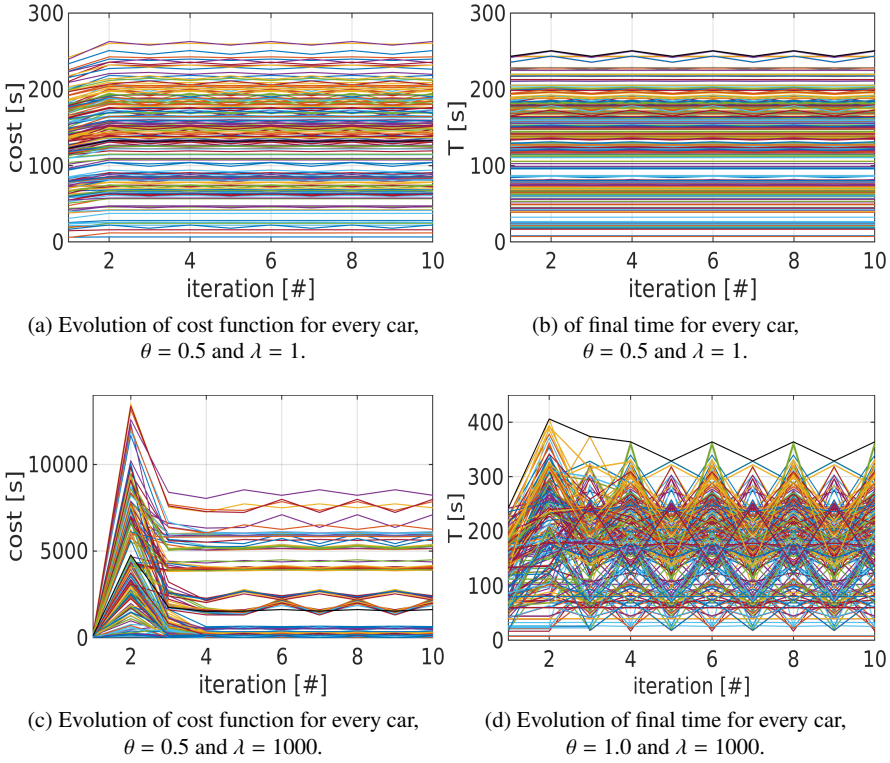
on every edge which is taken into account as an update on the edge cost in every iteration.

6.2 Influence of the parameters θ and λ

Considering the path planning in the UL-OP, which highly depends on the cost of the edges, the parameters θ and λ, which control the cost function, impact the result of the upper level path planning algorithm. Therefore we compare different parameter settings and analyze their effect on the cost function and the final time. Note that especially in the case of the cost function the values are not directly comparable, due to the different scaling factors. Hence we are more interested on the trend of the cost function itself.

Initially we examine λ, while fixing $\theta = 0.5$. The number of cars $n_c = 400$ is slightly reduced to speed up the computation. The other values remain as they were set in Section 6.1. Fig. 3 shows the comparison of the progression of the final time and the cost function for $\lambda = 1$ and $\lambda = 1000$ over the iterations.

Comparing the cost profiles, the increase of λ and therefore emphasizing the congestion as an increase in the cost of certain edges, leads to a convergence in the cost function. Thus the algorithm generates bundles of cars with the same cost, meaning multiple optima are achieved for such car bundles, and more important with a drastic decrease in the cost. Considering the final time, we notice an increase in the final time along side the increase in λ. For $\lambda = 1$ the cost functions as well as the final time remain almost constant, this is due to the underestimation of the traffic load. The traffic gets almost neglected, since the addition to the density on an edge is in the range of 0.1. Hence the increase in time for $\lambda = 1000$ is justified, since some cars get redirected on longer routes to their destination to avoid congestions.



(a) Evolution of cost function for every car, $\theta = 0.5$ and $\lambda = 1$.

(b) of final time for every car, $\theta = 0.5$ and $\lambda = 1$.

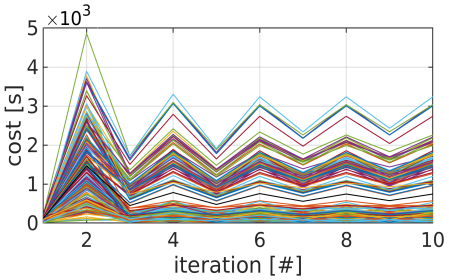
(c) Evolution of cost function for every car, $\theta = 0.5$ and $\lambda = 1000$.

(d) Evolution of final time for every car, $\theta = 1.0$ and $\lambda = 1000$.

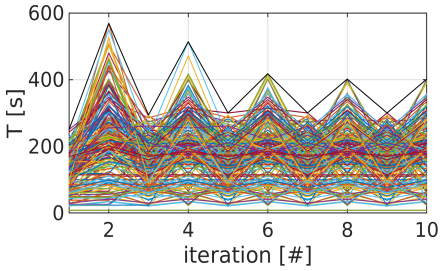
Fig. 3: Implication of the weight factor $\lambda \in [1; 1000]$ on the cost function and final times, with fixed $\theta = 0.5$, $n_c = 400$ and 25 vertices - 96 edges.

Through the stronger weight the traffic jam becomes emphasized. As a result we can draw the conclusion that a higher weight factor λ is recommended to achieve convergence and for a better distribution of the cars on the network.

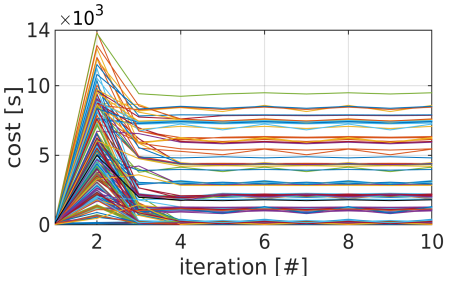
Considering the hyper-parameter θ, which influences the estimated representative edge speed $\hat{v}_e$, a higher value of θ shifts the representative edge speed in the direction of the minimum edge velocity, whereas a lower θ emphasizes the mean velocity along the edge over time, see eq. (15). The influence of θ on the cost function as well as the final time is shown in Fig. 4. Comparing the evolution of the cost function, we notice that the convergence and bundling effect grows with rising θ. However the magnitude of aberrations simultaneously rises, this effect can be countered by introducing additional constraints such that not only the average majority improves while others pay the price for it. Considering the evolution of the final time, the average final time decreases with increasing θ. With $\theta = 1$ the representative edge velocity is given through the minimum velocity value, which represents the worst case. The vehicles velocity on the same edge becomes devalued. This way



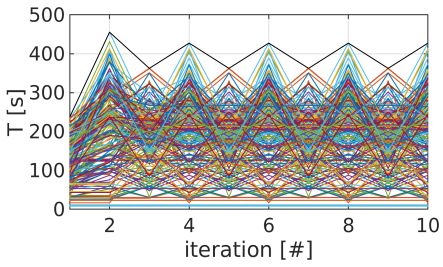
(a) Evolution of cost function for every car, $\theta = 0.0$ and $\lambda = 1000$.



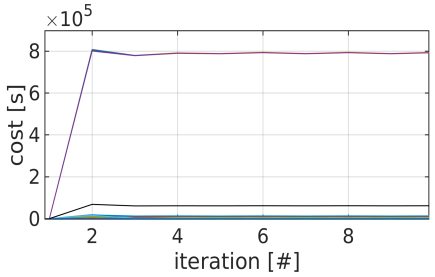
(b) of final time for every car, $\theta = 0.0$ and $\lambda = 1000$.



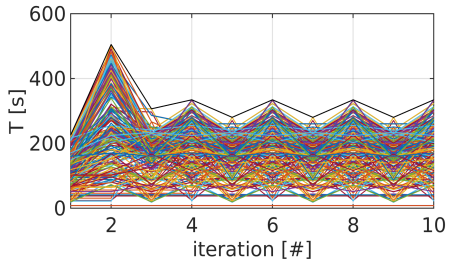
(c) Evolution of cost function for every car, $\theta = 0.5$ and $\lambda = 1000$.



(d) Evolution of cost function for every car, $\theta = 0.5$ and $\lambda = 1000$.



(e) Evolution of cost function for every car, $\theta = 1.0$ and $\lambda = 1000$.



(f) Evolution of final time for every car, $\theta = 1.0$ and $\lambda = 1000$.

Fig. 4: Implication of the hyper-parameter $\theta \in [0.0; 0.5; 1.0]$ on the cost function and final times, with fixed $\lambda = 1000$, $n_c = 400$ and 25 vertices - 96 edges.

the algorithm strives for a better distribution of the cars on the network, with the result that the vehicles on average reach their destination faster. We conclude that a higher value, respectively closer to $\theta_{max} = 1.0$ is recommended.

7 Conclusions

In this paper we presented an iterative algorithm for solving a bi-level optimal control problem. Furthermore we presented a model for a combined single car and network control through density updates and optimal time control. Considering the numerical results we could show that an increase in the hyper-parameters θ, λ affect the optimal solution and emphasize the convergence. The upper level control leads to an optimal distribution of cars among the edges of the network, such that in the lower level OCP an optimal speed profile for each car can be computed with the upper level solution as a constraint. Despite the increase in the final time, which results from longer paths due to a compromise for congestion avoidance, we showed that the density update on the edge cost affects the solution of each car and as a result to bundling of cars with the same cost. The increase of the magnitude of aberrations can be neglected. A simple fix would be the introduction of additional constraints to prevent such runaways.

References

1. E. Bertolazzi and M. Frego. Semi-analytical minimum time solution for the optimal control of a vehicle subject to limited acceleration, arXiv:1603.06245 [math.NA], 2016.
2. A. Bressan, S. Čanić, M. Garavello, M. Herty, and B. Piccoli. Flows on networks: recent results and perspectives. *EMS Surveys in Mathematical Sciences*, 1(1):47–111, 2014.
3. A. Bressan and K. T. Nguyen. Conservation law models for traffic flow on a network of roads. *Networks and Heterogeneous Media*, 10(2):255–293, 2015.
4. E. Cristiani, B. Piccoli, and A. Tosin. Multiscale Modeling of Granular Flows with Application to Crowd Dynamics. *Multiscale Modeling & Simulation*, 9(1):155, 2011.
5. E. Cristiani, B. Piccoli, and A. Tosin. How can macroscopic models reveal self-organization in traffic flow? In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 6989–6994, 12 2012.
6. E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.
7. M. Frego, E. Bertolazzi, F. Biral, D. Fontanelli, and L. Palopoli. Semi-analytical minimum time solutions for a vehicle following clothoid-based trajectory subject to velocity constraints. In *2016 European Control Conference (ECC)*, pages 2221–2227, June 2016.
8. P. Goatin, S. Göttlich, and O. Kolb. Speed limit and ramp meter control for traffic flow networks. *Engineering Optimization*, 48(7):1121–1144, 2016.
9. M. Lighthill and J. Whitham. On kinematic waves. *Proc. R. Soc. Lond.*, 229(A):281–345, 1955.
10. M. Gerdts and F. Lempio. *Mathematische Optimierungsverfahren des Operations Research*. DeGruyter, Berlin, Boston, 2011.
11. K. D. Palagachev and M. Gerdts. Numerical approaches towards bi-level optimal control problems with scheduling tasks. in: Math for the Digital Factory, Editors: L. Ghezzi, D. Hömberg, C. Landry, Springer, Berlin, to appear 2017.