Coordination and control of traffic at bottleneck areas

Bart van Gerwen

Oktober 2017

Supervisors: Prof. dr. Rob van der Mei (VU, CWI)
Daphne van Leeuwen (CWI)
Frank Ottenhof (TrafficLink)
Dr. Wieb Bosma (RU)

Contents

Abstract

With more people starting to drive cars more often, we are starting to see more and more traffic on the roads, which results in more traffic jams. Since expanding and redesigning the road network is often not able to keep up with the rapidly increasing demand, we need to look into ways to better use the existing road structure.
In order to accomplish this, we first need to be able to predict the result of our actions. To do this, we introduce a model based on the Fastlane model to predict the state of the road after a certain policy. Since this model can only handle simple road structures, we expand it so that it can be used on more difficult structures as well.
To optimize the usage of the road, we need an objective to minimize. We start by using vehicle loss hours, but later switch to to minimizing the number of vehicles on the road for each timestep. We approach this problem as an integer program, but later we also introduce an heuristic for a faster processing time. As such, we minimized the total number of vehicles on the relevant road network over an arbitrary amount of timesteps.

1 Introduction

With more people starting to drive cars more often, we are starting to see more and more traffic on the roads, which results in more traffic jams. Unfortunately, expanding and redesigning the road network is expensive and time consuming, and thus often not able to keep up with the increasing demand. For this reason, we must look for ways to better optimize the usage of the existing infrastructure.

About TrafficLink

TrafficLink is the market leader in The Netherlands in the field of dynamic traffic management. Its software makes it possible to manage and optimize the flow of traffic in a large area automatically. TrafficLink manages the rush-hour lanes, safety in tunnels, realtime video images of traffic, route information on Dynamic Route Information Panels (DRIPs) and much more. All of this is done in an integral manner, from one interface. TrafficLink allows systems of different suppliers to work together, and makes automatic tuning between road managers from different areas possible. TrafficLink also sells a dynamic traffic management system that automatically controls all kinds of traffic managements tasks. While some basic functionalities on intersection can easily be controlled, their advanced functionalities can also handle more complex networks like the area of Amsterdam.

TrafficLink wanted to produce an app that redirects people in case an incident has happened on their roads. While TrafficLink already has the software to identify incidents and notify people of these, TrafficLink did not yet have the software to calculate an optimal advice to give to drivers. While constructing this software, TrafficLink found out it was also applicable to situations without an incident. Thus, it could also be used in cases like the Ouwehands case (which will be described in more detail later on) where a certain road is overused.

Approach

In this thesis, we will try to address the following questions:

1. How can we model traffic around bottle necks?
2. How can we actively react to peak moments through the use of smart traffic management?
3. How can we improve the use of the road capacity of all road sections?

Organization

- In Chapter 2 we will discuss the basics of traffic flow theory and introduce the Lighthill–Whitham–Richards (LWR) model.
- In Chapter 3 we will present a model that can be used to calculate rerouting policies. We will then expand this model so that it can be used on more complex road structures. We will also introduce algorithms to calculate the policies with this model.
- In Chapter 4 the performance of our algorithm will be tested.

2 Prerequisites

In this chapter the basic definitions will be explained, as well as the relations between these concepts. After that, we will introduce the Wardrop equilibrium, which we will use later on in Chapter 3. Finally, we will describe the Lighthill-Whitman-Richards model which is the base of the used model in this thesis.

2.1 Basics of traffic flow theory

There are several ways one can go about modelling traffic flow. In particular, there are three types of models generally used: *microscopic*, *macroscopic* and *mesoscopic*.

- A *microscopic* model looks at individual drivers. It assume that drivers follow a leading vehicle and adjust their behaviour to that. This means that its characteristics at a certain time only depend on its characteristics in the previous time step, as well as those of its leader.
- A *macroscopic* model views traffic as continuüm flow without distinguishing individual vehicles. Traffic is seen as a continuüm liquid. It uses aggregated variables like the average velocity and average intensity of a certain location and time.
- A *mesoscopic* model lies in between the former two. It does model individual vehicles, but uses aggregated terms like probability distributions.

What type of model is used largely depends on the available data. The data TrafficLink uses suits a macroscopic model more, so that is what we will focus on in this thesis.

2.1.1 Basic definitions

In the models we are going to consider three concepts take center stage: *density*, *intensity* and *velocity*. Density is the number of vehicles occupying a unit of road length, intensity is the number of vehicles that pass a certain point in space per time unit and velocity is the speed of vehicles at a certain time and position. From now on we will write p for the density, q for the intensity and v for the velocity. Since they are dependent on time and space, it becomes $p(x,t)$, $q(x,t)$ and $v(x,t)$. the following more precise definitions are taken from [1].

Definition 2.1 *The density in an area A with spatial length dx is determined by the time s_n that vehicle $n \in \{1,...,N_A\}$ is present in A during time dt:*

$$p_A := \frac{\sum_{n=1}^{N_A} s_n}{dxdt}. \tag{1}$$

Definition 2.2 *The intensity (sometimes also referred to as flow) in a space-time area A with spatial length dx is determined by the number of vehicles N_A that travel through the area during time dt. Denote the distance vehicle $n \in \{1,...,N_A\}$ travels through A by y_n. Then:*

$$q_A := \frac{\sum_{n=1}^{N_A} y_n}{dxdt}. \tag{2}$$

Definition 2.3 *The velocity in area A with spatial length dx during time dt is the total distance travelled by the N_A vehicles divided by the total time they spent in A:*

$$v_A := \frac{\sum_{n=1}^{N_A} y_n}{\sum_{n=1}^{N_A} s_n} = \frac{q_A}{p_A}. \tag{3}$$

Note that this is only defined when there are vehicles in A. Formula (3) gives us a relation between the three.

This gives us a definition for an area in space and time. However, for our models we need a definition for a point in space time. We can get this definition by taking $dt \to 0$ and $dx \to 0$.

Definition 2.4 *Take $A = [x, x+dx] \times [t, t+dt]$ and write $N([x, x+dx], [t, t+dt])$ for the number of vehicles in A (instead of N_A). We can now define*

$$p(x,t) := \lim_{dx\to 0}(\lim_{dt\to 0}[p_A]) = \lim_{dx\to 0}(\frac{1}{dx}\lim_{dt\to 0}(\frac{\sum_{n=1}^{N_A} s_n}{dt})).$$

Since there is a maximum speed at which vehicles can drive, there is also a maximum distance they can travel in a certain period of time. Thus for every dx, it holds that for small enough dt we have $s_n = dt$ for all n. Using this we get

$$= \lim_{dx\to 0}(\frac{1}{dx}\lim_{dt\to 0}\frac{\sum_{n=1}^{N([x,x+dx],[t,t+dt])} dt}{dt}) = \lim_{dx\to 0}(\frac{N([x,x+dx],t)}{dx}).$$

To summarize:

$$p(x,t) := \lim_{dx\to 0}(\frac{N([x,x+dx],t)}{dx}). \tag{4}$$

Definition 2.5 *We define*

$$q(x,t) := \lim_{dt\to 0}(\lim_{dx\to 0}[q_A]) = \lim_{dt\to 0}(\frac{1}{dt}\lim_{dx\to 0}(\frac{\sum_{n=1}^{N_A} y_n}{dx})).$$

Since there are only a finite number of vehicles in A, there is a minimal distance all vehicles have driven in A. If we assume this minimal distance is not zero, then there is a length dx for which it holds that $y_n = dx$ for all n. Thus we have

$$\lim_{dt\to 0}(\frac{1}{dt}\lim_{dx\to 0}(\frac{\sum_{n=1}^{N_A} d_x}{dx})) = \lim_{dt\to 0}(\frac{N(x,[t+dt])}{dt})$$

In short:

$$q(x,t) := \lim_{dt\to 0}(\frac{N(x,[t+dt])}{dt}). \tag{5}$$

Definition 2.6 *Finally, we define the velocity, again as*

$$v(x,t) := \frac{q(x,t)}{p(x,t)}. \tag{6}$$

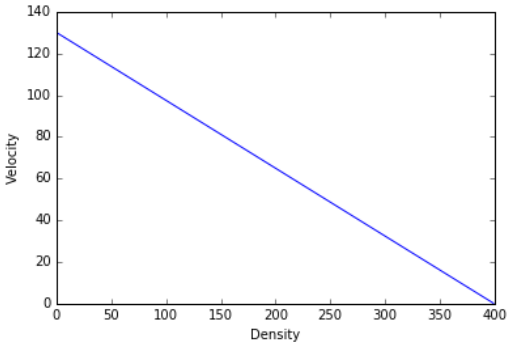
2.1.2 Fundamental diagrams

With relation (6), we can now always determine one of the three variables as long as we know the other two. Next we will assume a relation between the density and the intensity, so that we can determine both the velocity and the intensity as long as we know the density.

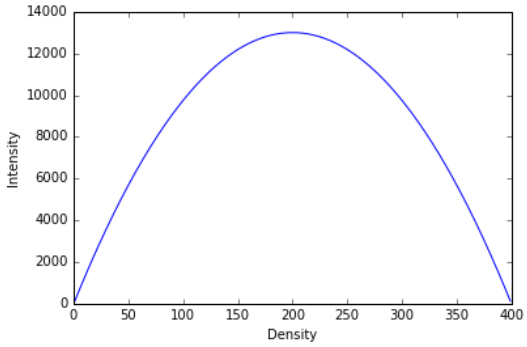
This was first done by Greenshield [2], who assumed that the velocity is linearly dependent on the density. For his relation, he introduced two new parameters, namely the maximum speed $v^{\max}$ and the jam density p^{jam}. The jam density denotes the maximum number of vehicles (per kilometer) that fit on a road. Once this limit is reached, nobody will be able to move anymore.

Greenshield's relation was as follows:

$$v(p) = v^{\max} - \frac{v^{\max}}{p^{\text{jam}}}p. \tag{7}$$



(a) Density-Velocity $v(p)$



(b) Density-Intensity $q(p)$

Figure 2: Fundamental diagrams by Greenshield

We can then derive from Formula's (6) and (7) that

$$q(p) = pv^{\max} - \frac{v^{\max}}{p^{\text{jam}}}p^2. \tag{8}$$

In short, it means that the more vehicles enter a road, the slower they can drive (as can be seen in Figure 2).
After this many other forms were proposed. One of these is the fundamental diagram of Daganzo [3], which assumes the relation between intensity and density consists of two linear functions, as seen in Figure 3.

Daganzo introduces a new parameter, namely p^{crit}. This stands for the critical density, namely the density at which the intensity is maximal (as shown in Figure 3a).

This diagram is given by the relation

$$q(p) = \begin{cases} v^{\max}p & \text{if } 0 \leq p \leq p^{\text{crit}} \\ \frac{v^{\max}p^{\text{crit}}}{p^{\text{jam}}-p^{\text{crit}}}(p^{\text{jam}} - p) & \text{if } p^{\text{crit}} \leq p \leq p^{\text{jam}}. \end{cases}$$

This also gives us

$$v(p) = \begin{cases} v^{\max} & \text{if } 0 \leq p \leq p^{\text{crit}} \\ \frac{p^{\text{crit}}v^{\max}}{p^{\text{jam}}-p^{\text{crit}}}(\frac{p^{\text{jam}}}{p} - 1) & \text{if } p^{\text{crit}} \leq p \leq p^{\text{jam}}. \end{cases}$$

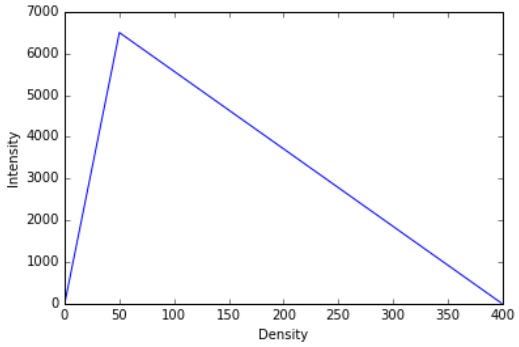
Another interesting variable for this model is the maximum intensity $q^{\max}$. Using formula (6) we can calculate this variable by

$$q^{\max} := v^{\max}p^{\text{crit}}. \tag{9}$$

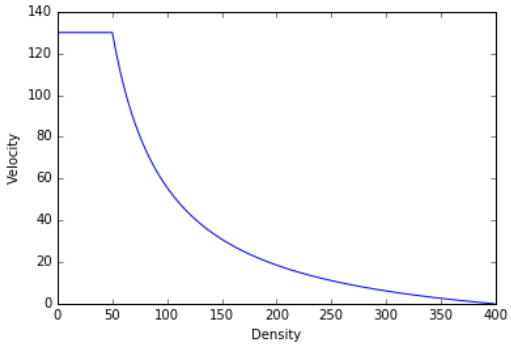
The final fundamental diagram we will look at is the one introduced by Smulders in 1990 [4]. This diagram assumes that the intensity grows quadratically in relation to the density when in free flow, and that it decreases linearly in congestion. To do this, he introduces a new variable, namely the critical velocity v^{crit}. The diagram, which can be seen in Figure 4, is given by

$$q(p) = \begin{cases} v^{\max}p - \frac{v^{\max}-v^{\text{crit}}}{p^{\text{crit}}p^2} & \text{if } 0 \leq p \leq p^{\text{crit}} \\ \frac{p^{\text{crit}}v^{\text{crit}}}{p^{\text{jam}}-p^{\text{crit}}}(p^{\text{jam}} - p) & \text{if } p^{\text{crit}} \leq p \leq p^{\text{jam}}. \end{cases}$$

This gives us the following formula for the velocity:

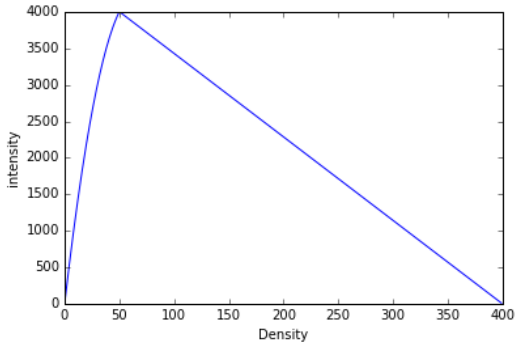


(a) Density-Intensity $q(p)$

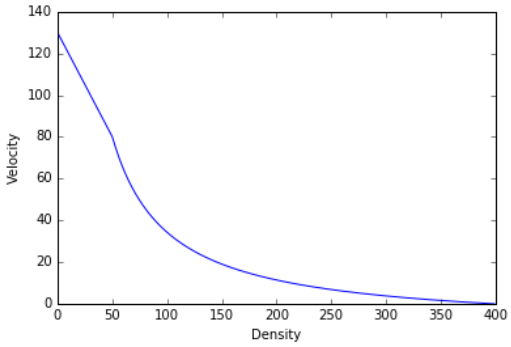


(b) Density-Velocity $v(p)$

Figure 3: Fundamental diagrams by Daganzo



(a) Density-Intensity $q(p)$



(b) Density-Velocity $v(p)$

Figure 4: Fundamental diagrams by Smulders

$$v(p) = \begin{cases} v^{\max} - \frac{v^{\max} - v^{\text{crit}}}{p^{\text{crit}}} & \text{if } 0 \leq p \leq p^{\text{crit}} \\ \frac{p^{\text{crit}} v^{\text{crit}}}{p^{\text{jam}} - p^{\text{crit}}} (\frac{p^{\text{jam}}}{p} - 1) & \text{if } p^{\text{crit}} \leq p \leq p^{\text{jam}}. \end{cases}$$

We can retrieve Daganzo's fundamental diagram if we choose $v^{\max} = v^{\text{crit}}$.

It is shown in [1] that the difference between Smulders and Daganzo in free flow is not too large. For this reason, we will use the simpler Daganzo diagram in our models.

2.2 Wardrop equilibrium

There are several ways to model traffic flow. One of those is with graph theory. The following definitions are taken from [5] and [7]. We will consider a routing network represented by $G = (V, E)$, where V is the set of nodes and E the set of directed arcs.

- $C \subset V \times V$ is a set of origin-destination pairs.
- for every $k \in C$ we have a demand flow of d_k, which denotes the traffic that wants to travel from that origin to that destination.
- For each $k \in C$, we define R_k to be the set of all routes from the origin to the destination.
- We then define $R = \cup R_k$.
- A link flow will be a non-negative vector $f = (f_a)_{a \in E}$ describing the traffic rate in each arc.
- We have a non-negative, non-decreasing and continuous travel cost function $t_a(.)$ for all $a \in E$ that map the flow f_a to the time it takes to travel said arc.
- A route flow is a non-negative vector $h = (h_r)_{r \in R}$ that meets the demand, or $d_k = \sum_{r \in R_k} h_r$ for all $k \in C$. Given a route flow, we can calculate the link flow as $f_a = \sum_{a \in r} h_r$.
- Finally for a flow f, we define the travel cost of that flow along a route r as $c_r = \sum_{a \in r} t_a(f_a)$.

Definition 2.7 *A route flow h is called a Wardrop equilibrium if and only if for all $k \in C$ and all $r \in R_k$*

$$c_r(h) = \min_{q \in R_k} c_q(h).$$

In other words, in a Wardrop equilibrium, every vehicle has no better alternative than its currently decided upon route to travel from its origin to its destination.
While a Wardrop equilibrium does not need to be the optimal distribution to optimize the total travel time, one can argue about how reliable the believe is that drivers will follow a given route knowing that there is a better alternative. We will use the principle of Wardrop equilibriums later on.

2.3 Lighthill-Whitman-Richards model

Now that we have introduced the basics of traffic flow, we will consider one of the first traffic flow models, the Lighthill-Whitman-Richards model (also called the LWR model). It is a model that perceives traffic as a continuum flow, using water as an example.

The basic principle of this model is that traffic does not disappear arbitrarily. This is reflected in the following partial differential equation:

$$\frac{d}{dt} p(x,t) + \frac{d}{dx} q(x,t) = 0 \tag{10}$$

which is called the conservation law. With this law, we can calculate the density of a part of the road as long as we know the flow into that and the flow out of it. A picture is shown in Figure 5.

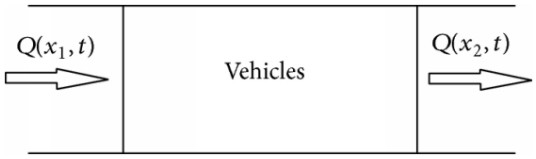


Figure 5: Densities in segments and their boundary flows.

We can derive this conservation law as follows:
Let $A = [x_1, x_2] \times [t_1, t_2]$ be a space time area as before. The conservation law dictates that the amount of vehicles in $[x_1, x_2]$ at time t_1 together with what enters between t_1 and t_2 should be equal to the amount of vehicles in $[x_1, x_2]$ at time t_2 together with what has left between t_1 and t_2. In formula:

$$\int_{x_1}^{x_2} p(x, t_1)dx + \int_{t_1}^{t_2} q(x_1, t)dt = \int_{x_1}^{x_2} p(x, t_2)dx + \int_{t_1}^{t_2} q(x_2, t)dt.$$

For practical use we want a discretisation for A. To obtain this, we will set $x_2 = x_1 + \Delta x$ and $t_2 = t_1 + \Delta t$ and take the limits $\Delta t \to 0$ and $\Delta x \to 0$. Since we are going to take these limits, we will make the approximation of

$$\int_{x_1}^{x_1+\Delta x} p(x, t_1)dx \approx p(x_1, t_1)\Delta x$$

$$\int_{t_1}^{t_1+\Delta t} q(x_1, t)dt \approx q(x_1, t_1)\Delta t$$

$$\int_{x_1}^{x_1+\Delta x} p(x, t_1 + \Delta t)dx \approx p(x_1, t_1 + \Delta t)\Delta x$$

$$\int_{t_1}^{t_1+\Delta t} q(x_1 + \Delta x, t)dt \approx q(x_1 + \Delta x, t_1)\Delta t.$$

If we rewrite our earlier formula with this discretisation, we get

$$p(x_1, t_1)\Delta x + q(x_1, t_1)\Delta t = p(x_1, t_1 + \Delta t)\Delta x + q(x_1 + \Delta x, t_1)\Delta t.$$

Thus

$$p(x_1, t_1)\Delta x + q(x_1, t_1)\Delta t - p(x_1, t_1 + \Delta t)\Delta x - q(x_1 + \Delta x, t_1)\Delta t = 0.$$

If we now divide both sides by $\Delta x \Delta t$ we get

$$\frac{p(x_1, t_1) - p(x_1, t_1 + \Delta t)}{\Delta t} + \frac{q(x_1, t_1) - q(x_1 + \Delta x, t_1)}{\Delta x} = 0.$$

Which (after taking the limits $\Delta t \to 0$ and $\Delta x \to 0$) gives us the conservation law.

To use this model for practical purposes, we will need to make some assumptions. First, we will divide the road in segments of a certain space length Δx. We will assume that the traffic situation is the same at every point in a segment. Thus, if $[x_1, x_1 + \Delta x)$ is a segment, and $y_1, y_2 \in [x_1, x_1 + \Delta x)$, then $p(y_1, t) = p(y_2, t)$ for every t. Furthermore, we will divide the time we are looking at the model in timesteps of a certain time length Δt and assume that the traffic situation remains the same during a timestep. Thus, if $[t_1, t_1 + \Delta t)$ is a timestep and $r_1, r_2 \in [t_1, t_1 + \Delta t)$ then $p(x, r_1) = p(x, r_2)$ for every x.

The number of vehicles in a segment at the new time is the number of vehicles in that segment at the old time plus the number of vehicles that have entered the segment during the time step minus the number of vehicles that have left the segment. To calculate this, we will first define a few new variables, starting with the effective supply σ and the effective demand δ. The effective supply is the amount of traffic the link can take in. We can calculate this as:

$$\sigma_{j,i} = \begin{cases} q_{j,i} & \text{if } p_{j,i} \leq p_i^{\text{crit}} \\ q_i^{\max} & \text{if } p_i^j \geq p_i^{\text{crit}}. \end{cases}$$

The effective demand is the amount of traffic that wants to leave the link. We can calculate this as:

$$\delta_{j,i} = \begin{cases} q_i^{\max} & \text{if } p_{j,i} \leq p_i^{\text{crit}} \\ q_{j,i} & \text{if } p_i^j \geq p_i^{\text{crit}}. \end{cases}$$

We can now define the flow $q_{i-1\to i}^j$ from segment $i-1$ to i at timestep j as

$$q_{i-1\to i}^j = \min(\sigma_{i+1}^j, \delta_i^j). \tag{11}$$

With these definitions, we can now calculate the density in a segment i at timestep $j+1$ by considering the density of the previous timestep. As an equation:

$$p_{j+1,i}\Delta x = p_{j,i}\Delta x + q_j^{i-1\to i}\Delta t - q_j^{i\to i+1}\Delta t$$

By dividing both sides by Δx we get

$$p_{j+1,i} = p_{j,i} + \frac{\Delta t}{\Delta x}(q_j^{i-1\to i} - q_j^{i\to i+1}). \tag{12}$$

With these formulas, we can recursively calculate the traffic state of every segment as long as we know $p_{0,i}$ for all i.

3 Model for rerouting

In this chapter we determine how we can pro-actively route traffic to stop traffic jams from forming, or at least keep them as small as possible. To do this, we will assume we know about the future inflow into our road network. This will allow us to predict the properties of the formation of traffic jams, as explained in the previous chapter. While this lets us predict what would happen when we do nothing, we can do more than just watch. We can also tell the arriving traffic to take a different route to their destination to decrease the inflow in a certain bottleneck. However, while doing this we have to be careful we do not cause a traffic jam in an alternative road, and of course we also don't want to redirect people when there is no need for it. To accomplish all of this, we will use our knowledge from the previous chapter to construct a model that takes rerouting policies into account, and then use this to try to calculate an optimal routing policy. We will first introduce a model constructed in [1] and expand that to a more general model.

3.1 A basic model

Since we need an objective function to determine an optimal policy, we will introduce Vehicle Loss Seconds (VLS). The lost time of a vehicle is the time in seconds lost by traversing a certain road segment relative to the minimum time needed to travel that segment. If this time is multiplied by the number of vehicles, we obtain VLS. For example, if it takes a vehicle 30 seconds to travel some segment, while it should only take 20 seconds if that vehicle was driving the maximum velocity, it has lost 10 seconds. If five vehicles are traversing that segment, then the segment has a VLS of 50 veh $\cdot$ s. The model described in [1] analyses a basic form of a routing structure. There are $R \in \mathbb{N}$ roads leading from point A to B. We assume road 1 to be the fastest if no traffic is present, because it has a higher maximum velocity and/or it is shorter. Note that this assumption is not necessary for the mathematical analysis of the problem. At first glance, it seems that taking road 1 will result in the shortest travel time, but this is of course not always true. When more traffic is present on the road, the velocity of the vehicles decreases and the travel time of the road increases. This is due to the fundamental diagram in velocity v being a decreasing function of the density. When road 1 becomes over-crowded, it is favourable to take a different road, until that road gets too busy, and so on.

We assume the roads are the segments, thus the density over each whole road is constant, and therefore, the velocity is also constant. Point A is the only location in the network where a decision can be taken. There, a vehicle can choose to take road $r \in \{1, ..., R\}$, which it has to follow until point B is reached. However, we do not model the vehicles separately, since we assumed traffic to be a continuum flow. Hence, at every time t we decide what fraction of the flow will take which road. Since we want to make this computable, we have to discretise time. Let $\{0, ..., J\}$ be the set of all time steps of size Δt. Here, $j = 0$ corresponds with the initial situation on the roads.

As we have seen in equation (8), for every timestep j and every road r we can calculate the density of the next timestep by

$$p_{j+1,r} := p_{j,r} + \frac{\Delta t}{X_r}(q^{\text{in}}_{j,r} - q^{\text{out}}_{j,r})$$

where $q^{\text{in}}_{j,r}$ denotes the inflow into road r during timestep j and $q^{\text{out}}_{j,r}$ the outflow out of road r during timestep j. These values need to be calculated as well, of course.

We can calculate the inflow into road r at timestep j by taking the minimum of the supply and demand at that point. The demand would be the amount of traffic that wants to enter road r, which can be formulated by $x_{j,r}\lambda_j$, thus considering the inflow into the model at the time, and multiplying it by the percentage that gets send to road r. The supply is the amount of traffic that can enter road r, which, as we have seen before, can be calculated by

$$\min[v_r^{\max} p_r^{\text{crit}}, \frac{p_r^{\text{jam}} - p_{j,r}}{p_r^{\text{jam}} - p_r^{\text{crit}}}(p_r^{\text{crit}} v_r^{\max})],$$

where it is $v_r^{\max} p_r^{\text{crit}}$ in free flow and $\frac{p_r^{\text{jam}} - p_{j,r}}{p_r^{\text{jam}} - p_r^{\text{crit}}}(p_r^{\text{crit}} v_r^{\max})$ in congested flow.
If we take the minimum of the supply and the demand, we see that

$$q_{j,r}^{\text{in}} := \min[x_{j,r}\lambda_j, v_r^{\max} p_r^{\text{crit}}, \frac{p_r^{\text{jam}} - p_{j,r}}{p_r^{\text{jam}} - p_r^{\text{crit}}}(p_r^{\text{crit}} v_r^{\max})]. \tag{13}$$

For the outflow, we will assume that the road after point B is in free flow and can always take in all the traffic that gets sent there. Thus, we only need to calculate the amount of traffic that wants to leave road r at time j. As seen before, this gives us

$$q_{j,r}^{\text{out}} := v_r^{\max} \min[p_{j,r}, p_r^{\text{crit}}]. \tag{14}$$

So q^{out} is $v_r^{\max} p_{j,r} = q_{j,r}$ when road r is in free flow and $v_r^{\max} p_r^{\text{crit}} = q_r^{\max}$ when it is congested.

With these formulas, we can calculate the density for each road for every timestep step by step. If we now want to know the velocity $v_{j,r}$ on a road, we can calculate this by using the fundamental diagram. The travel time $T_{j,r}$ can then be calculated by dividing the road length by the velocity of vehicles on the road

$$T_{j,r} := \frac{X_r}{v_{j,r}}. \tag{15}$$

If we now define the minimal travel time T_{min} as

$$T_{min} := \min_{0 \le r \le R} [\frac{X_r}{v_r^{\max}}]$$

so the minimal time it takes to get from point A to point B, we can calculate the lost time of a vehicle entering road r at time j as $T_{j+1,r} - T_{min}$. This gives us the following objective function:

$$\sum_j \sum_r q_{j,r}^{\text{in}} (T_{j+1,r} - T_{min}). \tag{16}$$

Thus, for every timestep j, we calculate for every road r how much time a vehicle taking that road would lose in comparison to the minimal travel time (this is $T_{j+1,r} - T_{min}$) and multiply that amount by the inflow in road r, which represents the amount of traffic that is sent to this road. It is shown in [1] that this set of equations can be expressed in an IP program, using $x_{j,r}$ as variables.

3.2 Expanding the model

While the above model can be useful for certain simple situations, we will generally want to look at more complex models. To do this, we first need to expand this model. We will do this step by step. Our goal for constructing these models is to find a relation between the inflow into a model Λ_j, the density on each road $p_{j,r}$ and the routing policy $x_{j,r}$. In particular, we want to be able to determine the density of future timesteps ($p_{1,r}, p_{2,r}$,etc.) based on the current density of each road, the inflow into the model and the routing policy. Once we are able to do this, we can change the formulation into an IP-program to find the optimal routing policy.

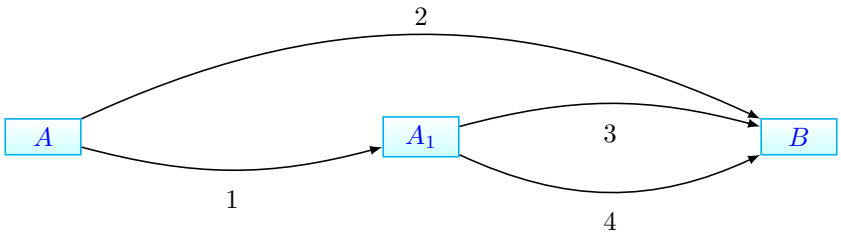


Figure 6: Example of a road network with two crossings.

3.2.1 Two crossings

To start, we want to be able to add multiple crossings. To do this we will first add one more crossing and see how this impacts our model. We will thus consider the situation portrait in Figure 6: We will keep using the same formula for the density, namely

$$p_{j+1,r} := p_{j,r} + \frac{\Delta t}{X_r}(q^{\text{in}}_{j,r} - q^{\text{out}}_{j,r}).$$

Where $r \in R$ and $j \in J$. R is the set of roads and J the set of timesteps. The density at the first time step is part of the input.
In order to define the inflow and outflow used in this formula, however, we will need to make a case distinction:

- **Case 1: $\mathbf{r \leq 2}$**, or when we have R_1 choices at point A, $r \leq R_1$.
  In this case the road left of this one is one assumed to be in free flow, so we can just use formula (13) again

$$q^{\text{in}}_{j,r} := \min[x_{j,r}\lambda_j; p^{\text{crit}}_r v^{\text{max}}_r; p^{\text{crit}}_r v^{\text{max}}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}].$$

- **Case 2: $\mathbf{r > 2}$**, or when we have R_1 choices at point A, $r > R_1$.
  In this case, the inflow comes from road 1, and is thus depended on the amount of traffic on that route. We have calculated with formula (14) that the intensity of traffic wanting to leave road 1 is $v^{\text{max}}_1 \min[p_{j,1}; p^{\text{crit}}_1]$. If we now substitute this in formula (13), we get

$$q^{\text{in}}_{j,r} := \min[x_{j,r} v^{\text{max}}_1 \min[p_{j,1}; p^{\text{crit}}_1]; p^{\text{crit}}_r v^{\text{max}}_r; p^{\text{crit}}_r v^{\text{max}}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}]. \tag{17}$$

  Note that this is basically $min[q^{\text{out}}_{j,1}; q^{\text{in}}_{j,r}]$, if we were to use the old formula's. So we are still taking the minimum of the demand and the supply.

Note that, in order to make sure that the density never exceeds p^{jam}_r, it is sufficient to have $\frac{\Delta t p^{\text{crit}}_{j,r} v^{\text{max}}_r}{X_r(p^{\text{jam}}_r - p^{\text{crit}}_r)} \leq 1$. This can be achieved by taking Δt small enough.
For the outflow we will need to use a similar case distinction:

- **Case 1: $\mathbf{r > 1}$**. In this case the road to the right of r is assumed to be in free flow, so we can use the formula (14) once more

$$q^{\text{out}}_{j,r} := v^{\text{max}}_r \min[p_{j,r}; p^{\text{crit}}_r].$$

- **Case 2: $\mathbf{r = 1}$**. In this case the road to the right of r is not necessarily in free flow. However, since we have already calculated the inflow for those roads, and this traffic all comes from road 1, we can simply take

$$q^{\text{out}}_{j,1} := q^{\text{in}}_{j,3} + q^{\text{in}}_{j,4}.$$

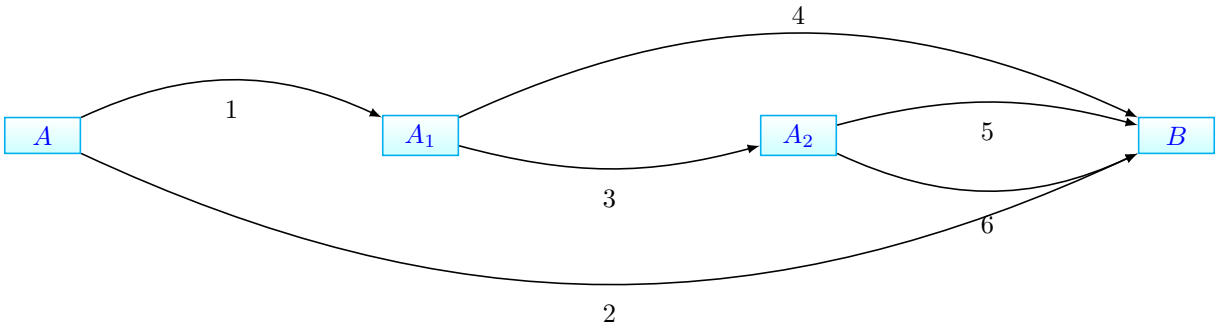


Figure 7: Example of a road network with three crossings.

or

$$q_{j,1}^{\text{out}} := \sum_{s=R_1}^{R_2} q_{j,s}^{\text{in}} \tag{18}$$

if we assume there to be R_1 possible road choices on point A and R_2 at point A_1. Note that the old formula for the outflow of r did not completely disappear. it is still used to calculate the inflow of the roads after it.

To make sure that the density never becomes negative, we need to have $p_{j,r} \geq q_{j,r}^{\text{out}}$. This will always be the case if $v_r^{\max} \frac{\Delta t}{X_r} \leq 1$. This can again be achieved by taking Δt small enough.

3.2.2 n crossings

We can now extend this model to one with an arbitrary number of crossings without much difficulty. We will assume that we have n crossings and R_m road options on crossing m. In Figure 7 we see an example with $n = 3$ and $R_1, R_2, R_3 = 2$

We retain the formula for the density:

$$p_{j+1,r} := p_{j,r} + \frac{\Delta t}{X_r}(q_{j,r}^{\text{in}} - q_{j,r}^{\text{out}}).$$

For the inflow, we have the the same formulas as before:

- **Case 1: $\mathbf{r \leq R_1}$.** In this case, we can use formula (13) again:

$$q_{j,r}^{\text{in}} := \min[\lambda_j x_{j,r}; p_r^{\text{crit}} v_r^{\max}; p_r^{\text{crit}} v_r^{\max} \frac{p_r^{\text{jam}} - p_{j,r}}{p_r^{\text{jam}} - p_r^{\text{crit}}}]$$

- **Case 2: $\mathbf{R_m < r \leq R_{m+1}}$ ($\mathbf{1 \leq m \leq n}$).** In this case we can use a slight variation of formula (17):

$$q_{j,r}^{\text{in}} := \min[x_{j,r} v_{R_{m-1}+1}^{\max} \min[p_{j,R_{m-1}+1}; p_{R_{m-1}+1}^{\text{crit}}]; p_r^{\text{crit}} v_r^{\max}; p_r^{\text{crit}} v_r^{\max} \frac{p_r^{\text{jam}} - p_{j,r}}{p_r^{\text{jam}} - p_r^{\text{crit}}}] \tag{19}$$

Note that this is almost the same formula as before, only now the road to the left is not always road 1.

The outflow also uses the same formula's as the model with only two crossings:

- **Case 1: $\mathbf{r = R_m + 1}$ ($\mathbf{0 \leq m < n}$).** In this case we do the same as in equation (18), we just take the sum over the inflow of all following roads:

$$q_{j,r}^{\text{out}} := \sum_{i=R_m+1}^{R_{m+1}} q_{j,i}^{\text{in}}$$

- **Case 2: $\mathbf{r \neq R_m + 1}$**. In this case we have assumed that the following road is in free flow, thus we can use (14) again:

$$q_{j,r}^{\text{out}} := v_r^{\max} \min[p_{j,r}; p_r^{\text{crit}}].$$

3.2.3 The IP formulation

While we could use the method used in [1] to make the model from our previous section into an IP formulation, this would require calculating the result of all the possible scenarios to have all the constraints. As one might imagine, this takes to much computation time for our more complex model. Thus, we will now construct an IP formulation for our model that is easier to use for more complex road structures.

Definition 3.1 *A Linear Program is a mathematical model where all the requirements are linear. It is a technique to optimize a linear objective function, subject to linear equality and inequality constraints. A linear program can be written in canonical form as*

$$\begin{gathered} \textit{Maximize} \quad \textit{(Minimize)} \quad c^T x \\ \textit{subject} \quad \textit{to} \; Ax \leq b \\ \textit{and} \quad x \geq 0 \end{gathered}$$

where x is the vector of unknown variables, b and c are vectors of known coefficients, and A is a matrix of known coefficients. Linear Programs are very useful, because many problems can be written in this form, and they scale very well. This means that even if the vectors and A become bigger, the time it takes to solve the problem does not grow exponentially.

However, not every problem can written as a LP-problem. One of the restrictions for an LP-program is that every variable can become a real number, where this is not always desirable. It also severely limits the tricks we can use to rewrite a problem. Since several of our formula's are not linear, we will need to be able to put extra restrictions on variables other than equality or inequality relations to translate it.

Definition 3.2 *An Integer Program is an Linear Program where (some of) the variables are integers.*

While IP-problems are more general than LP-problems, they do not scale as well. However, sometimes we need to use them because an LP-problem is just not viable. Using LP or IP formulations on traffic models has been done before, for instance by [6] and [8]. These models do not take into account that roads might become congested however, our model will take that into account.

To make this into an IP-problem we will use the variables $p_{j,r}, q_{j,r}^{\text{in}}, q_{j,r}^{\text{out}}, x_{j,r}$.
We will start with introducing the expression we want to minimize. This is the number of vehicles on the road, over all time steps, so:

$$\sum_{j,r} p_{j,r} X_r. \tag{20}$$

Note that this expression is linear. This might seem quite different from the lost vehicle hours, but this is actually not the case. Minimizing the lost-vehicle-hours is, after all the same as minimizing the total travel time. This sum is:

$$\sum_{j,r} \Delta t X_r \frac{1}{v_{j+1,r}} q_{j,r}^{\text{in}}.$$

Now, if our time steps and roads are small enough, the inflow will be about the same as the flow in the road part. Thus we can write

$$\sum_{j,r} \Delta t X_r \frac{1}{v_{j+1,r}} q_{j,r}.$$

Again, if our time steps are small enough, the speed of timestep j will be about the same as the speed of time step $j+1$. Thus we get:

$$\sum_{j,r} \Delta t X_r \frac{1}{v_{j,r}} q_{j,r}$$

$$= \sum_{j,r} \Delta t X_r p_{j,r}.$$

Since Δt is a constant, we can just leave it out when minimizing. This gives us the first expression. Note that dividing by the speed would give us a non-linear expression.
Now that we know what we want to optimize, we will now look at the formula's in our model and make them into linear constraints.

- The first formula we consider is the one for the density, formula (8):

  $$p_{j+1,r} := p_{j,r} + \frac{\Delta t}{X_r}(q^{\text{in}}_{j,r} - q^{\text{out}}_{j,r}).$$

  Note that this formula is linear in our variables and can thus be used as a constraint.

- Next we will consider the inflow: for $r \leq R_1$ we use formula (13)

  $$q^{\text{in}}_{j,r} := \min[\lambda_j x_{j,r}; p^{\text{crit}}_r v^{\max}_r; p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}].$$

  Note that this is not a linear constraint. However, we can still enforce this by using linear relations. To do this we need to a number $M_{j,r}$ that is larger than $\max[\lambda_j x_{j,r}; p^{\text{crit}}_r v^{\max}_r; p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}]$. We can achieve this by taking

  $$M_{j,r} := \max[\lambda_j; p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r}{p^{\text{jam}}_r - p^{\text{crit}}_r}].$$

  Note that we can calculate $M_{j,r}$ with only constants (and thus no variables). If we now introduce three new binary variables $b_{j,r,1}, b_{j,r,2}$ and $b_{j,r,3}$ we can now translate this formula into the following set of constraints:

  $$q^{\text{in}}_{j,r} \leq \lambda_j x_{j,r}$$

  This constraint forces the inflow to not be higher than the demand on it.

  $$q^{\text{in}}_{j,r} \leq p^{\text{crit}}_r v^{\max}_r$$

  This constraint forces the inflow to not be higher than the supply of the road when it is in free flow.

  $$q^{\text{in}}_{j,r} \leq p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}$$

  This constraint forces the inflow to not be higher than the supply of the road when it is congested.

  $$q^{\text{in}}_{j,r} \geq \lambda_j x_{j,r} - b_{j,r,1} M_{j,r}$$

  When $b_{j,r,1} = 1$ this constraint forces the inflow to be higher than a negative number (Since $M_{j,r}$ is a higher number than λ_j) and when $b_{j,r,1} = 0$ it forces the inflow to be higher than or equal to the demand on the road. Combining this with the first constraint would force the inflow to be equal to the demand.

  $$q^{\text{in}}_{j,r} \geq p^{\text{crit}}_r v^{\max}_r - b_{j,r,2} M_{j,r}$$

This constraint works the same as the previous one, only now it forces the inflow to be higher than or equal to the supply of the road when in free flow if $b_{j,r,2} = 0$, and combined with the second constraint, it would force equality.

$$q^{\text{in}}_{j,r} \geq p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r} - b_{j,r,3} M_{j,r}$$

This constraint is similar to the two previous ones, only now with the supply when the road is congested.

$$b_{j,r,1} + b_{j,r,2} + b_{j,r,3} = 2$$

Finally, this constraint forces exactly one of $b_{j,r,1}, b_{j,r,2}$ and $b_{j,r,3}$ to be zero. Because of this, equality is reached in exactly one of the previous three cases. Together with the first three constraints, this forces the inflow to be equal to the minimum of the three.

- For $R_m < r \leq R_{m+1}$ $(1 \leq m \leq n)$ we use equation (19)

$$q^{\text{in}}_{j,r} := \min[x_{j,r} v^{\max}_{R_{m-1}+1} \min[p_{j,R_{m-1}+1}; p^{\text{crit}}_{R_{m-1}+1}]; p^{\text{crit}}_r v^{\max}_r; p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}]$$

Here we can do a similar thing as before, but we have to be a bit more careful because the term $x_{j,r} v^{\max}_{R_{m-1}+1} p_{j,R_{m-1}+1}$ is not linear (since both $x_{j,r}$ and $p_{j,R_{m-1}+1}$ are variables). We can solve this by first assuming $x_{j,r} = 1$ and then slightly altering the constraints to accommodate for when $x_{j,r} = 0$. We start again by taking a large number:

$$M_{r,j} := \max[v^{\max}_{R_{m-1}+1} p^{\text{jam}}_{R_{m-1}+1}; p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r}{p^{\text{jam}}_r - p^{\text{crit}}_r}]$$

The construction of the following eight constraints follows the same procedure as before when we assume $x_{j,r} = 1$:

$$q^{\text{in}}_{j,r} \leq v^{\max}_{R_{m-1}+1} p_{j,R_{m-1}+1}$$

Forces that the inflow is lower than the demand on the road when the previous road is in free flow.

$$q^{\text{in}}_{j,r} \leq v^{\max}_{R_{m-1}+1} p^{\text{crit}}_{R_{m-1}+1}$$

Forces the inflow to be lower than the demand on the road when the previous road is congested.

$$q^{\text{in}}_{j,r} \leq p^{\text{crit}}_r v^{\max}_r$$

Forces that the inflow is lower than the supply of the road when it is in free flow.

$$q^{\text{in}}_{j,r} \leq p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}$$

Forces that the inflow is lower than the supply of the road when it is congested.

$$q^{\text{in}}_{j,r} \geq v^{\max}_{R_{m-1}+1} p_{R_{m-1}+1} - b_{j,r,1} M_{j,r}$$

Forces the inflow to be equal to the demand on the road when the previous road is in free flow and $b_{j,r,1} = 1$.

$$q^{\text{in}}_{r,j} \geq v^{\max}_{R_{m-1}+1} p^{\text{crit}}_{R_{m-1}+1} - b_{r,j,2} M_{j,r}$$

Forces the inflow to be equal to the demand of the road when the previous road is congested and $b_{j,r,2} = 1$.

$$q^{\text{in}}_{j,r} \geq p^{\text{crit}}_r v^{\max}_r - b_{j,r,3} M_{j,r}$$

Forces the inflow to be equal to the supply of the road when the road is in free flow and $b_{j,r,3} = 1$

$$q^{\text{in}}_{j,r} \geq p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r} - b_{j,r,4} M_{j,r}$$

Forces the inflow to be equal to the supply of the road when the road is congested and $b_{j,r,4} = 1$. We will now add a restraint to accommodate for the case that $x_{j,r} = 0$.

$$q_{j,r}^{\text{in}} \leq x_{j,r} M_{j,r}$$

This equation will force the inflow to be equal to zero when $x_{j,r} = 0$.

$$b_{r,j,1} + b_{r,j,2} + b_{r,j,3} + b_{r,j,4} = 3 + (1 - x_{r,j})$$

this last constraint is almost the same as before, only now we add $(1 - x_{j,r})$ to the right hand side, so that the inflow can be equal to zero when $x_{j,r} = 0$.

Now we will look at the outflow.

- For $r = R_m + 1$ $(0 \leq m < n)$ we use formula (18):

$$q_{j,r}^{\text{out}} := \sum_{i=R_m+1}^{R_{m+1}} q_{j,i}^{\text{in}}$$

Since this is a linear expression, we can simply use it as a constraint.

- For $r \neq R_m + 1$ we use formula (14):

$$q_{j,r}^{\text{out}} := v_r^{\max} \min[p_{j,r}; p_r^{\text{crit}}].$$

We will need to change this expression, but fortunately we can just use the same method as with the inflow. If we pick a large number

$$N_{j,r} := v_r^{\max} p_r^{\text{jam}}$$

we can translate the formula into the following constraints:

$$q_{j,r}^{\text{out}} \leq v_r^{\max} p_{j,r}$$

$$q_{j,r}^{\text{out}} \leq p_r^{\text{crit}} v_r^{\max}$$

$$q_{j,r}^{\text{out}} \geq v_r^{\max} p_{j,r} - a_{j,r,1} N_{j,r}$$

$$q_{j,r}^{\text{out}} \geq p_r^{\text{crit}} v_r^{\max} - a_{j,r,2} N_{j,r}$$

$$a_{j,r,1} + a_{j,r,2} = 1$$

- Finally we also have the constraint that for all $j \in J$ and all $0 \leq m < M$

$$\sum_{r=R_m+1}^{R_{m+1}} x_{j,r} = 1.$$

All of these constraints together serve to enforce our model. To recap, our program comes down to

$$\textit{Minimize} \quad \sum_{j,r} p_{j,r} X_r$$

under the constraints

$$\forall_{j,r} : \; p_{j+1,r} = p_{j,r} + \frac{\Delta t}{X_r} (q_{j,r}^{\text{in}} - q_{j,r}^{\text{out}})$$

$$\forall_{j,r\leq R_1}: \begin{cases} q^{\text{in}}_{j,r} \leq \lambda_j x_{j,r} \\ q^{\text{in}}_{j,r} \leq p^{\text{crit}}_r v^{\max}_r \\ q^{\text{in}}_{j,r} \leq p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r} \\ q^{\text{in}}_{j,r} \geq \lambda_j x_{j,r} - b_{j,r,1} M_{j,r} \\ q^{\text{in}}_{j,r} \geq p^{\text{crit}}_r v^{\max}_r - b_{j,r,2} M_{j,r} \\ q^{\text{in}}_{j,r} \geq p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r} - b_{j,r,3} M_{j,r} \\ b_{j,r,1} + b_{j,r,2} + b_{j,r,3} = 2 \end{cases}$$

$$\forall_{j,r>R_1}: \begin{cases} q^{\text{in}}_{j,r} \leq v^{\max}_{R_{m-1}+1} p_{j,R_{m-1}+1} \\ q^{\text{in}}_{j,r} \leq v^{\max}_{R_{m-1}+1} p^{\text{crit}}_{R_{m-1}+1} \\ q^{\text{in}}_{j,r} \leq p^{\text{crit}}_r v^{\max}_r \\ q^{\text{in}}_{j,r} \leq p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r} \\ q^{\text{in}}_{j,r} \geq v^{\max}_{R_{m-1}+1} p_{R_{m-1}+1} - b_{j,r,1} M_{j,r} \\ q^{\text{in}}_{r,j} \geq v^{\max}_{R_{m-1}+1} p^{\text{crit}}_{R_{m-1}+1} - b_{r,j,2} M_{j,r} \\ q^{\text{in}}_{j,r} \geq p^{\text{crit}}_r v^{\max}_r - b_{j,r,3} M_{j,r} \\ q^{\text{in}}_{j,r} \geq p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r} - b_{j,r,4} M_{j,r} \\ q^{\text{in}}_{j,r} \leq x_{j,r} M_{j,r} \\ b_{r,j,1} + b_{r,j,2} + b_{r,j,3} + b_{r,j,4} = 3 + (1 - x_{r,j}) \end{cases}$$

$$\forall_{j,0\leq m<n,r=R_{m+1}}: q^{\text{out}}_{j,r} := \sum_{i=R_m+1}^{R_{m+1}} q^{\text{in}}_{j,i}$$

$$\forall_j \forall_{r\neq R_m+1}: \begin{cases} q^{\text{out}}_{j,r} \leq v^{\max}_r p_{j,r} \\ q^{\text{out}}_{j,r} \leq p^{\text{crit}}_r v^{\max}_r \\ q^{\text{out}}_{j,r} \geq v^{\max}_r p_{j,r} - a_{j,r,1} N_{j,r} \\ q^{\text{out}}_{j,r} \geq p^{\text{crit}}_r v^{\max}_r - a_{j,r,2} N_{j,r} \\ a_{j,r,1} + a_{j,r,2} = 1 \end{cases}$$

$$\sum_{r=R_m+1}^{R_{m+1}} x_{j,r} = 1.$$

3.3 Overlapping routes

We will now extend our model yet again by allowing alternative routes to overlap. See the Figure 8 for an example. Unfortunately, solving larger problems for these complex networks results in to much computation time. For this reason, we will not include an IP formulation of this model in this section, but we will introduce an heuristic for the model in the next section.

We will first need to change our notation. First off, for every road r we will create a set $P(r)$ of its predecessors and a set $S(r)$ of its successors. Note that we did not need these sets before, because it was easy to determine them based on the number of alternative routes. Next, to determine the inflow, we will partition our roads in three sets, to determine how each road works. We will call these sets $\text{In}_1, \text{In}_2, \text{In}_3$.

- In_1 is the set of roads that start in A, so the roads without predecessors. This consists of roads 1 and 2 in the above example.
- In_2 is the set of roads that have only one predecessor. This consists of roads 3 and 4 in the example.

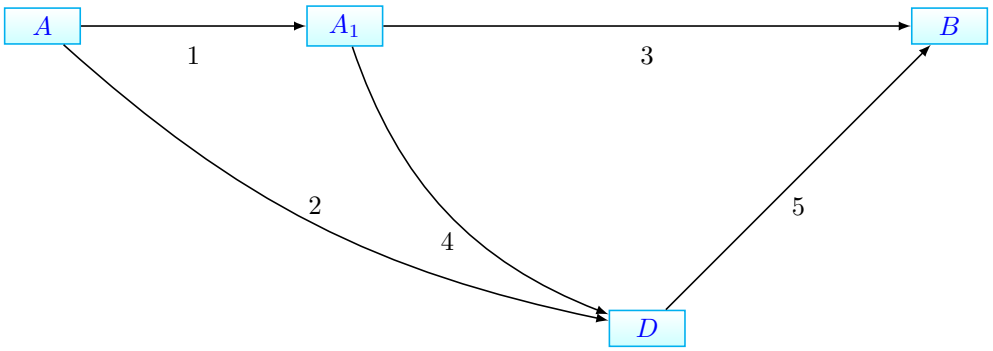


Figure 8: Example of a road network where alternative roads overlap.

- In_3 is the set of roads that have two predecessors. Note that we assume a road will not have more than two predecessors. This consists of road 5.

For the outflow we will partition the roads into three sets:

- Out_1 is the set of roads that end in B, so the roads with no successors. We see that this set would consist of road 3 and 5 in the example.
- Out_2 is the set of roads whose successor is an element of In_3. This consists of roads 2 and 4 in the example.
- Out_3 is the set of roads not in $\text{Out}_1, \text{Out}_2$. In our example, this consists of road 1.

Note that we assume that each element in Out_2 only has one successor. We will still calculate the the density for every road with:

$$p_{j+1,r} := p_{j,r} + \frac{\Delta t}{X_r}(q^{\text{in}}_{j,r} - q^{\text{out}}_{j,r}).$$

The inflow also remains mostly the same, except for one small change:

- **Case 1:** r has no predecessors. In this case we can just use formula (13) again:

$$q^{\text{in}}_{j,r} := \min[x_{j,r}\lambda_j; p^{\text{crit}}_r v^{\max}_r; p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}]$$

- **Case 2:** r has only one predecessor. In this case we can just use formula (19):

$$q^{\text{in}}_{j,r} := \min[x_{j,r} v^{\max}_s \min[p_{j,s}; p^{\text{crit}}_s]; p^{\text{crit}}_r v^{\max}_r; p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}],$$

  where s is the predecessor of r. Note that if s has only one successor, we can leave $x_{j,r}$ out (since it will always need to be 1).

- **Case 3:** r has multiple predecessors. In this case we can use a slight variation of Formula (19):

$$q^{\text{in}}_{j,r} := \min[\sum_{s \in P(r)} v^{\max}_s \min[p_{j,s}; p^{\text{crit}}_s]; p^{\text{crit}}_r v^{\max}_r; p^{\text{crit}}_r v^{\max}_r \frac{p^{\text{jam}}_r - p_{j,r}}{p^{\text{jam}}_r - p^{\text{crit}}_r}]. \quad (21)$$

  Note that the only difference between this expression and Fromula (19) is that we now we have a sum as the first component. Furthermore, since we assume that each element in $Out2$ only has one successor, we do not need an x-variable.

Now we will consider the outflow:

- **Case 1:** r has no successors. In this case we simply use formula (14) yet again:
$$q_{j,r}^{\text{out}} := v_r^{\max} \min[p_{j,r}; p_r^{\text{crit}}],$$
just as before.
- **Case 2:** the successor of r has multiple predecessors. In this case we can use the following formula:
$$q_{j,r}^{\text{out}} := \min[v_r^{\max} \min[p_{j,r}; p_r^{\text{crit}}]; a_r q_{j,u}^{\text{in}}]$$
$$+ \max[\min[a_s q_{j,u}^{\text{in}} - v_s^{\max} \min[p_{j,s}; p_s^{\text{crit}}]; v_r^{\max} \min[p_{j,r}; p_r^{\text{crit}}] - a_r q_{j,u}^{\text{in}}]; 0]$$
where u is the successor of r and $s \in P(u)$ with $s \neq r$. Note that the expression simply divides the supply of road u between roads r and s, while making sure that as much of this supply is used (so if the demand of road s is smaller than the part of the supply it is allowed, we give this to road r). a_r and a_s are how we indicate the supply, so $a_r + a_s = 1$, and $a_r, a_s \geq 0$.
- In all other cases we can just sum over the inflow of all successors:
$$q_{j,r}^{\text{out}} := \sum_{u \in S(r)} q_{j,s}^{\text{in}}$$

3.3.1 Multiple starting points and background traffic

We will now make the final extension for our model by including background traffic and accept there to be inflow at multiple points. In order to not make things to complicated, we will in turn assume that every starting point (so points where no roads lead to) only has a single road going out of it. We now need to introduce several new variables. First off, we introduce the background percentage $\phi_{j,r}$ that tells us what percentage of the traffic on road j is background traffic at time step j. We will also assume that the background percentage of the inflow is always the same and define this as ϕ. Background traffic is the traffic that we cannot direct. Because of this we also need a separate variable that tells us how this background traffic distributes itself over the roads. For this we will use $\overline{x_{j,r}}$.

Furthermore, instead of using λ_j for the inflow in the model, we will now have to write $\lambda_{j,r}$, since there is an inflow for each road (not to be confused with $q_{j,r}^{\text{in}}$, which also takes into account the cars already in the model). Since not all of this background traffic needs to go to the endpoint of the model, it is possible they will leave the model at some point. For this reason we will also add a variable to represent the percentage of exiting cars, namely $\psi_{j,r}$. With these we can once again calculate the density for multiple timesteps.

The formula for the density now changes slightly, it becomes:

$$p_{j+1,r} = p_{j,r} - \frac{\Delta t}{X_r}(q_{j,r}^{\text{in}} - q_{j,r}^{\text{out}} - v_r^{\max} \min[\psi_{j,r} p_{j,r} \phi_{j,r}, p_{j,r}^{\text{crit}}]). \tag{22}$$

Note that it is almost the same formula as before, apart from the fact that we also have to consider the disappearing vehicles. The amount of disappearing vehicles is calculated by multiplying the percentage of disappearing cars $\psi_{j,r}$ with the density $p_{j,r}$ and the maximum speed $v_r^{\max}$. However, the percentage of exiting vehicles is taken from the background traffic, so we also need to multiply with the background percentage. This gives us $\psi_{j,r} p_{j,r} \phi_{j,r}$. Since the amount of traffic that can leave road r is limited as well, this gives us

$$v_r^{\max} \min[\psi_{j,r} p_{j,r} \phi_{j,r}, p_{j,r}^{\text{crit}}].$$

For the calculation of the inflow and outflow we will once again use case distinctions. The formulas will be almost the same, but with background traffic taken into account.

- **Case 1:** r has no predecessors:

$$q_{j,r}^{\text{in}} := \min[\lambda_{j,r}; p_r^{\text{crit}} v_r^{\max}; p_r^{\text{crit}} v_r^{\max} \frac{p_r^{\text{jam}} - p_{j,r}}{p_r^{\text{jam}} - p_r^{\text{crit}}}]. \tag{23}$$

Since we assume that there is no choice for the inflow, the formula remains mostly unchanged. The only difference is that we do not need a choice variable $x_{j,r}$ anymore. For the purpose of calculating $\phi_{j,r}$ more easily, we will keep track of what percentage of the inflow is background traffic. We will name this percentage $\phi_{j,r}^{\text{in}}$. In this case it is easy to see that

$$\phi_{j,r}^{\text{in}} = \phi. \tag{24}$$

- **Case 1:** r has one predecessor s, and s has only one successor (namely r):

$$q_{j,r}^{\text{in}} := \min[\lambda_{j,r} + v_s^{\max} \min[(1 - \phi_{j,s}\psi_{j,s})p_{j,s}; p_s^{\text{crit}}]; p_r^{\text{crit}} v_r^{\max}; p_r^{\text{crit}} v_r^{\max} \frac{p_r^{\text{jam}} - p_{j,r}}{p_r^{\text{jam}} - p_r^{\text{crit}}}], \tag{25}$$

where s is the predecessor of r. Again we notice that the formula remains mostly unchanged. The only differences are that there is now an inflow into r other than the vehicles from road s (namely $\lambda_{j,r}$) and a certain percentage of the vehicles on road s do not want to drive into road r (namely $\phi_{j,s}\psi_{j,s}$, the percentage of cars that want to leave the model).
Calculating $\phi_{j,r}^{\text{in}}$ is bit more difficult this time. We will need to calculate the demand on road r and what part of that demand is background traffic. We already know the demand on road r, namely

$$d_{j,r} := \lambda_{j,r} + v_s^{\max} \min[(1 - \phi_{j,s}\psi_{j,s})p_{j,s}; p_s^{\text{crit}}].$$

The amount of background traffic flowing into r from outside the model is simply $\lambda_{j,r}\phi$. The inflow of traffic from road s to road r is $v_s^{\max} \min[(1 - \phi_{j,s}\psi_{j,s})p_{j,s}; p_s^{\text{crit}}]$. Not all of this is background traffic of course, the percentage that is background is

$$\frac{\phi_{j,s}(1 - \psi_{j,s})p_{j,s}}{(\phi_{j,s}(1 - \psi_{j,s}) + 1 - \phi_{j,s})p_{j,s}} =: \phi_{j,r,s}^{\text{in}}.$$

This means we can calculate the background percentage of the inflow with

$$\phi_{j,r}^{\text{in}} := \frac{\lambda_{j,r}\phi + v_s^{\max}\phi_{j,r,s}^{\text{in}} \min[(\phi_{j,s}(1 - \psi_{j,s}) + 1 - \phi_{j,s})p_{j,s}, p_s^{\text{crit}}]}{d_{j,r}}. \tag{26}$$

- **Case 3:** r has one predecessor s, and s has multiple successors (including r). We can calculate the demand on road r by

$$d_{j,r} := \lambda_{j,r} + v_s^{\max} \min[((1 - \phi_{j,s})x_{j,r} + \phi_{j,s}(1 - \psi_{j,s})\overline{x_{j,s}})p_{j,s}; p_s^{\text{crit}}] \tag{27}$$

where s is the predecessor of r. It may seem like a lot has changed from before, but that is not entirely the case. The only things that have changed is that we again take into account that traffic can flow into the road from outside the model and that we cannot direct all the traffic. The traffic from road s that wants to flow into r consists of two parts: there is $(1-\phi_{j,s})x_{j,r}p_{j,s}$, which is the traffic we have directed to road r and there is $\phi_{j,s}(1-\psi_{j,s})\overline{x_{j,s}}$, which consists of the background traffic headed to road r. We can then calculate the inflow as we are used to:

$$q_{j,r}^{\text{in}} := \min[d_{j,r}; p_r^{\text{crit}} v_r^{\max}; p_r^{\text{crit}} v_r^{\max} \frac{p_r^{\text{jam}} - p_{j,r}}{p_r^{\text{jam}} - p_r^{\text{crit}}}].$$

To calculate the background percentage of the inflow, it helps to first calculate the background percentage of the inflow from just route s into road r. We can do this as follows:

$$\phi_{j,r,s} := \frac{\phi_{j,s}(1 - \psi_{j,s})\overline{x_{j,r}}p_{j,s}}{(\phi_{j,s}(1 - \psi_{j,s})\overline{x_{j,r}} + (1 - \phi_{j,s})x_{j,r})p_{j,s}}.$$

We can also calculate the demand of road s on road r (so the amount of traffic from road s that wants to go to road r), which is

$$d_{j,r,s} := v_s^{\max} \min[(\phi_{j,s}(1-\psi_{j,s})\overline{x_{j,r}} + (1-\phi_{j,s})x_{j,r})p_{j,s}, p_s^{\text{crit}}].$$

We can calculate $\phi_{j,r}^{\text{in}}$ by:

$$\phi_{j,r}^{\text{in}} := \frac{\lambda_{j,r}\phi + \phi_{j,r,s}d_{j,r,s}}{d_{j,r}} \tag{28}$$

- **Case 4:** r has multiple predecessors. first, we define

$$d_{j,r} := (\sum_{s\in P(r)} v_s^{\max} \min[(1-\psi_{j,s}\phi_{j,s})p_{j,s}, p_s^{\text{crit}}]) + \lambda_{j,r}.$$

We can now define

$$q_{j,r}^{\text{in}} := \min[d_{j,r}, p_r^{\text{crit}}v_r^{\max}; p_r^{\text{crit}}v_r^{\max}\frac{p_r^{\text{jam}} - p_{j,r}}{p_r^{\text{jam}} - p_r^{\text{crit}}}]. \tag{29}$$

Before we calculate the full background percentage, we first note that we can use the same formula's for the background percentage for the flow from a predecessor s to r, and of the demand of s on r:

$$\phi_{j,r,s} := \frac{\phi_{j,s}(1-\psi_{j,s})p_{j,s}}{(\phi_{j,s}(1-\psi_{j,s}) + (1-\phi_{j,s})x_{j,r})p_{j,s}},$$

$$d_{j,r,s} := v_s^{\max} \min[(\phi_{j,s}(1-\psi_{j,s}) + (1-\phi_{j,s})x_{j,r})p_{j,s}, p_s^{\text{crit}}].$$

Note that we do not use $\overline{x_{j,r}}$ here, because we assume that all predecessors of r don't have a choice in where to go anyway. We can calculate $\phi_{j,r}^{\text{in}}$ by:

$$\phi_{j,r}^{\text{in}} := \frac{\lambda_{j,r}\phi + \sum_{s\in P(r)} \phi_{j,r,s}d_{j,r,s}}{d_{j,r}}. \tag{30}$$

Similarly, we will also use $Out1, Out2$ and $Out3$ for the calculation of the outflow and keep track of a new variable $\psi_{j,r}^{\text{out}}$.

- **Case 1:** r has no successors:

$$q_{j,r}^{\text{out}} := v_r^{\max} \min[(1-\phi_{j,r}\psi_{j,r})p_{j,r}; p_r^{\text{crit}}]. \tag{31}$$

We note that the definition is mostly the same, apart from the fact that some traffic disappears elsewhere. The definition of $\psi_{j,r}^{\text{out}}$ is also relatively easy now:

$$\psi_{j,r}^{\text{out}} := \frac{\phi_{j,r}(1-\psi_{j,r})}{1-\phi_{j,r}\psi_{j,r}} \tag{32}$$

- **Case 2:** r has one successor s:

$$q_{j,r}^{\text{out}} := \frac{v_r^{\max} \min[(1-\phi_{j,r}\psi_{j,r})p_{j,r}; p_r^{\text{crit}}]}{d_{j,s}} q_{j,s}^{\text{in}} \tag{33}$$

Note that in this case we calculate things a bit different from the previous model. We now divide the overload over all the roads, instead of giving each road a certain percentage of s. The background percentage of this outflow can be calculated as follows:

$$\psi_{j,r}^{\text{out}} := \frac{\phi_{j,r}(1-\psi_{j,r})}{1-\phi_{j,r}\psi_{j,r}} \tag{34}$$

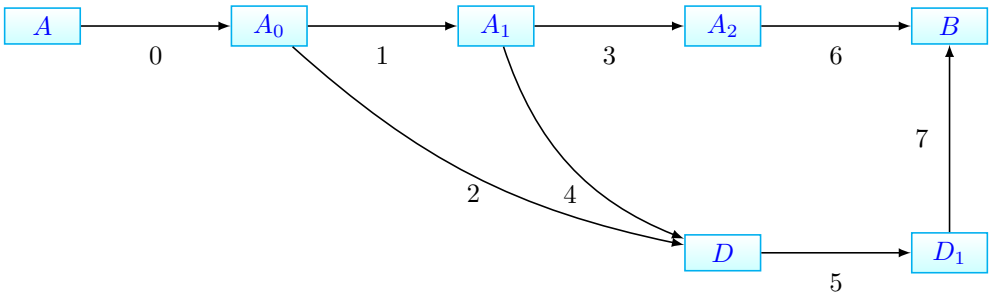


Figure 9: Example road network used for heuristic.

- **Case 3:** r has multiple successors:

$$q_{j,r}^{\text{out}} := \sum_{s \in S(r)} \frac{d_{j,s,r}}{d_{j,s}} q_{j,s}^{\text{in}}. \tag{35}$$

Before we could simply take the sum over all inflows of the successors of r. However, now that these roads also have an inflow from outside the graph, we cannot do this. Thus we now calculate what percentage of the inflow in each of these roads comes from road r and multiply this with the inflow. We can calculate $\phi_{j,r}^{\text{out}}$ as follows:

$$\phi_{j,r}^{\text{out}} := \frac{\sum_{s \in S(r)} \phi_{j,s,r} \frac{d_{j,s,r}}{d_{j,s}} q_{j,s}^{\text{in}}}{q_{j,r}^{\text{out}}} \tag{36}$$

Finally, we need to update the background percentage. We do this simply by calculating how much background traffic comes in and how much goes out in a time step. Thus, the formula is quite similar to the formula for the density.

$$\phi_{j+1,r} := \frac{\phi_{j,r} p_{j,r} + \frac{\Delta t}{X_r}(\phi_{j,r}^{\text{in}} q_{j,r}^{\text{in}} - \phi_{j,r}^{\text{out}} q_{j,r}^{\text{out}} - v_r^{\max} \min[\psi_{j,r} p_{j,r} \phi_{j,r}, p_r^{\text{crit}}])}{p_{j+1,r}}. \tag{37}$$

3.4 Heuristic

As stated before, solving larger and more complex problems requires to much computation time when we try to solve it with an IP formulation. Because of this we have derived a heuristic to calculate a good solution instead of the optimal solution (although it will still give the optimal solution in certain cases). Our plan for this is to calculate the Wardrop equilibrium for a model and try to approximate this as well as possible. We will now list the steps our heuristic goes through, and calculate an example. Our example will be as shown in Figure 9.

With the parameters for the road shown in Figure 10:

- **Step 1:** Distribute roads over categories
  We distribute the roads $1..R$ over the categories $In1, In3$,etc. as described in paragraph 2.3. For our example, this gives us:

$$In1 = \{0\}$$
$$In2 = \{1, 2, 3, 4, 6, 7\}$$
$$In3 = \{5\}$$
$$Out1 = \{6, 7\}$$
$$Out2 = \{2, 4\}$$
$$Out3 = \{0, 1, 3, 5\}$$

Roads	Road length	max speed	critical density	jam density	initial density
road 0	1 km	130 km/h	130 veh/km	260 veh/km	130 veh/km
road 1	2 km	130 km/h	130 veh/km	260 veh/km	130 veh/km
road 2	3 km	100 km/h	130 veh/km	260 veh/km	120 veh/km
road 3	2 km	130 km/h	130 veh/km	260 veh/km	120 veh/km
road 4	1 km	80 km/h	100 veh/km	200 veh/km	90 veh/km
road 5	2 km	130 km/h	130 veh/km	260 veh/km	20 veh/km
road 6	2 km	100 km/h	120 veh/km	240 veh/km	20 veh/km
road 7	2 km	130 km/h	130 veh/km	260 veh/km	20 veh/km

Figure 10: Parameter values for the example.

- **Step 2:** Find troubleroads
  We look at each road and consider whether it might get jammed or not. There are two cases in which this might happen: either if $r \notin Out2$ and

$$(\sum_{s \in S(r)} v_s^{\max} p_s^{\text{crit}} - max_j[\lambda_{j,s}]) < v_r^{\max} p_r^{\text{crit}},$$

  or if $r \in Out2$ and

$$(\sum_{u \in P(s)} v_u^{\max} p_u^{\text{crit}}) + \max_j[\lambda_{j,s}] > v_s^{\max} p_s^{\text{crit}},$$

  where s is the successor of r.
  So if the successors of r have a higher combined capacity than r, or if r's successor s has multiple predecessors and the capacity of those predecessors is higher than the capacity of s. We will call these roads the *trouble roads* of our network.
  In our example, we see that the trouble roads are roads $2, 3$ and 4.

- **Step 3:** Define routes
  We will now build the list named AllRoutes, which consists of, for each road, the routes drivers can take to the endpoint after that road, the traveltime of that route, whether it contains trouble roads or not and if so, what those troubleroads are. We do this as follows:

```
Allroutes=[[ ]*NumberofRoads]
curRoads=Out1
finRoads=[]
while curRoads is not empty:
  for r in curRoads:
    if all successors of r are in finRoads:
          if r has successors:
            add all routes of its successors to Allroutes[r] (+successor)
                add the traveltime of the successor to the total traveltime
                keep track of troubleroads in the route
                if a successor has no routes, make a route consisting of:
                  -the successor
                  -the traveltime over  this successor
                  -wheter it is a troubleroad (0 for no, 1 for yes)
          add r to finRoads
          add all predecessors of r to curRoads
          remove r from curRoads
```

  We will now follow this procedure for our example:

- We start of by setting curRoads=$[6, 7]$.
- We now take $r = 6$. Since 6 has no successors, we proceed to add 6 to finRoads and add all of its predecessors to curRoads. After that, we remove 6 from curRoads.
- We do the same as above, but now for $r = 7$. We now have

$$\text{finRoads} = [6, 7]$$

$$\text{curRoads} = [3, 5]$$

  Nothing has happened to the set AllRoutes yet.
- Since curRoutes is not empty yet, we will loop over it again. We will now start with $r = 3$.
- We see that all of r's successors are in finRoads. Since the successors have no routes, we have to make one. This route will consist of the successor (6), the travel time of this successor ($\frac{2}{100} = 0.02h$) and whether or not it's a trouble road (it's not). This gives us the route $[[6], 0.02, 0]$. we add this route in the second entry of our list. We now add r to the list finRoads, add all of it's predecessors to the list curRoads.
- We do the same for $r = 5$. This gives us the route $[[7], \frac{2}{130}, 0]$.
- Our sets now look like this:

$$\text{finRoads} = [3, 5, 6, 7]$$

$$\text{curRoads} = [1, 2, 4]$$

$$\text{AllRoutes} = [[\,], [\,], [\,], [[[6], 0.02, 0]], [\,], [[[7], \frac{2}{130}, 0]], [\,], [\,]]$$

- Since the list curRoads is not empty, we will loop over it again.
- We start with $r = 1$. Not all of r's successors are contained in finRoads yet, so we will skip this road for now.
- We continue with $r = 2$. r only has one successor (5) and this successor already has a route. This means we copy this route and add the successor to it. This gives us the route $[[5, 7], \frac{2}{130} + \frac{2}{130} = \frac{4}{130}, 0]$. We now add r to finRoads, add its predecessors to curRoads and remove r from curRoads.
- We continue with $r = 4$. This gives us the same route as before, namely $[[5, 7], \frac{4}{130}, 0]$. We now add r to finRoads, add its predecessors to curRoads and remove r from curRoads.
- Our sets now look like this:

$$\text{finRoads} = [2, 3, 4, 5, 6, 7]$$

$$\text{curRoads} = [0, 1]$$

$$\text{AllRoutes} = [[\,], [\,], [[[5, 7], \frac{4}{130}, 0]], [[[6], 0.02, 0]], [[[5, 7], \frac{4}{130}, 0]], [[[7], \frac{2}{130}, 0]], [\,], [\,]]$$

- Since curRoads is still not empty, we will loop over it once more. Since not all of 0's successors are in finRoads, we will continue with $r = 1$.
- Since r has two successors, we will consider both of them. We will start with 3.
- 3 only has the route $[[6], 0.02, 0]$, so we will just add 3 to it. However, we must also note that 3 is a troubleroad. this means our new route will now look like this: $[[3, 6], 0.02 + \frac{2}{130} = \frac{46}{1300}, 1, [3]]$.
- In a similar way, the successor 4 gives us the route $[[4, 5, 7], \frac{4}{130} + \frac{1}{80} = \frac{40}{1040}, 1, [4]]$. We now add r to finRoads, add its predecessor to curRoads and remove r from curRoads.

– Our sets now look like this:

$$\text{finRoads} = [1, 2, 3, 4, 5, 6, 7]$$

$$\text{curRoads} = [0]$$

$$\text{AllRoutes} = [[\,], [[[3, 6], \frac{46}{1300}, 1, [3]], [[4, 5, 7], \frac{40}{1040}, 1, [4]]], [[[5, 7], \frac{4}{130}, 0]], [[[6], 0.02, 0]],$$
$$[[[5, 7], \frac{4}{130}, 0]], [[[7], \frac{2}{130}, 0]], [\,], [\,]]$$

– We will now loop over curRoads for the last time. It has only one element now, so we take $r = 0$.

– r has two successors. We will first consider 1. Since 1 has two routes, we use both of these. This gives us the routes $[[1, 3, 6], \frac{46}{1300} + \frac{2}{130} = \frac{66}{1300}, 1, [3]]$ and $[[1, 4, 5, 7], \frac{40}{1040} + \frac{2}{130} = \frac{56}{1040}, 1, [4]]$.

– The other successor is 2. This road gives us the route $[[2, 5, 7], \frac{4}{130} + \frac{3}{100} = \frac{79}{1300}, 1, [2]]$. We now add r to finRoads, add its predecessors to curRoads and finally we remove r from curRoads.

– Our sets now look like this:

$$\text{finRoads} = [0, 1, 2, 3, 4, 5, 6, 7]$$

$$\text{curRoads} = [\,]$$

$$\text{AllRoutes} = [[[[1, 3, 6], \frac{66}{1300}, 1, [3]], [[1, 4, 5, 7], \frac{56}{1040}, 1, [4]], [[2, 5, 7], \frac{79}{1300}, 1, [2]]],$$
$$[[[3, 6], \frac{46}{1300}, 1, [3]],$$
$$[[4, 5, 7], \frac{40}{1040}, 1, [4]]], [[[5, 7], \frac{4}{130}, 0]], [[[6], 0.02, 0]],$$
$$[[[5, 7], \frac{4}{130}, 0]], [[[7], \frac{2}{130}, 0]], [\,], [\,]]$$

– Since curRoads is now empty, we are done.

- **Step 4:** Calculate rerouting density
Now we can calculate at which point routes become too slow and different routes should be chosen. The calculation is as follows:
let there be two routes, named route 1 and route 2. Assume route 1 is the fastest. Let r be the first trouble road in route 1, let T_1 be the travel time over route 1, T_2 the travel time over route 2, and t_r the travel time over road r. Then, the density we are looking for is

$$p^{reroute} := \frac{(T_2 - T_1 + t_r)p_r^{\text{jam}} v_r^{\max} p_r^{\text{crit}}}{X_r(p_r^{\text{jam}} - p_r^{\text{crit}}) + (T_2 - T_1 + t_r)v_r^{\max} p_r^{\text{crit}}}. \tag{38}$$

Using this formula, we go over the routes as follows:

```
for r in [1..R]:
  sort Allroutes[r] on traveltime (fastest rout on 0, etc.)
  for l in AllRoutes[r]:
    if l contains troubleroads:
          for each route k after l:
            calculate the rerouting density and add it to l
```

We will also calculate this in our example. Note that it is only relevant for the roads that have multiple routes in the set AllRoutes. This happens only in entry 0 and 1. Thus, we will calculate the rerouting densities for these routes:

- We will begin with road 0. It has routes $[[1,3,6],\frac{66}{1300},1,[3]]$, $[[1,4,5,7],\frac{56}{1040},1,[4]]$ and $[[2,5,7],\frac{79}{1300},1,[2]]$. Coincidentally, this is also how we'd order the routes from fastest to slowest (route $[1,3,6]$ is the fastest route to take after road 0, followed by route $[1,4,5,6]$ and finally route $[2,5,7]$ is the slowest). We also note that all of these routes contain a troubleroad. We will first calculate the rerouting densities for the first route:
- We start by comparing the first and the second route. We know that road 3 is the first troubleroad for the first route, so if we take $r=3$ and substitute in formula 38, we see that

$$\begin{aligned} p^{reroute} &:= \frac{(T_2 - T_1 + t_r)p_r^{\text{jam}} v_r^{\max} p_r^{\text{crit}}}{X_r(p_r^{\text{jam}} - p_r^{\text{crit}}) + (T_2 - T_1 + t_r)v_r^{\max} p_r^{\text{crit}}} \\ &= \frac{(\frac{56}{1040} - \frac{66}{1300} + \frac{2}{130})260 * 130 * 130}{2(260-130) + (\frac{56}{1040} - \frac{66}{1300} + \frac{2}{130})130 * 130} \\ &= \frac{1560}{11} \end{aligned}$$

  we now add this number to the entry of the first route.
- We do the same thing when we compare the first route and the third route. This gives us

$$\begin{aligned} p^{reroute} &= \frac{(\frac{79}{1300} - \frac{66}{1300} + \frac{2}{130})260 * 130 * 130}{2(260-130) + (\frac{56}{1040} - \frac{66}{1300} + \frac{2}{130})130 * 130} \\ &= \frac{8580}{53}. \end{aligned}$$

  We now also add this number to the entry of the first route.
- We now compare the second and third route with one another. Since the first troubleroad of of the second route is 4, we take $r=4$. This gives us:

$$\begin{aligned} p^{reroute} &:= \frac{(T_2 - T_1 + t_r)p_r^{\text{jam}} v_r^{\max} p_r^{\text{crit}}}{X_r(p_r^{\text{jam}} - p_r^{\text{crit}}) + (T_2 - T_1 + t_r)v_r^{\max} p_r^{\text{crit}}} \\ &= \frac{(\frac{79}{1300} - \frac{56}{1040} + \frac{1}{80})200 * 80 * 100}{1(200-100) + (\frac{79}{1300} - \frac{56}{1040} + \frac{1}{80})80 * 100} \\ &= \frac{10100}{83}. \end{aligned}$$

  We add this number to the entry of the second route.
- Our set AllRoutes now looks like this:

$$\begin{aligned} \text{AllRoutes} = [&[[[1,3,6],\frac{66}{1300},1,[3],\frac{1560}{11},\frac{8580}{53}], \\ &[[1,4,5,7],\frac{56}{1040},1,[4],\frac{10100}{83}],[[2,5,7],\frac{79}{1300},1,[2]]], \\ &[[[3,6],\frac{46}{1300},1,[3]],[[4,5,7],\frac{40}{1040},1,[4]]],[[[5,7],\frac{4}{130},0]],[[[6],0.02,0]], \\ &[[[5,7],\frac{4}{130},0]],[[[7],\frac{2}{130},0]],[\,],[\,]] \end{aligned}$$

– We will now do the same for road 1. This road contains the routes $[[[3,6], \frac{46}{1300}, 1, [3]]$ and $[[4,5,7], \frac{40}{1040}, 1, [4]]]$. We notice that the first route is the fastest one this time as well. We also see that the faster route has a trouble road, so we need to take $r = 3$. This means we get:

$$p^{reroute} := \frac{(T_2 - T_1 + t_r)p_r^{\text{jam}} v_r^{\max} p_r^{\text{crit}}}{X_r(p_r^{\text{jam}} - p_r^{\text{crit}}) + (T_2 - T_1 + t_r)v_r^{\max} p_r^{\text{crit}}}$$

$$= \frac{(\frac{40}{1040} - \frac{46}{1300} + \frac{2}{130}) * 260 * 130 * 130}{2 * (260 - 130) + (\frac{40}{1040} - \frac{46}{1300} + \frac{2}{130}) * 130 * 130}$$

$$= \frac{1560}{11}.$$

We will now add this number to the entry of the fastest route.

– The set AllRoutes now looks like this:

$$\text{AllRoutes} = [[[[1,3,6], \frac{66}{1300}, 1, [3], \frac{1560}{11}, \frac{8580}{53}],$$

$$[[1,4,5,7], \frac{56}{1040}, 1, [4], \frac{10100}{83}], [[2,5,7], \frac{79}{1300}, 1, [2]]],$$

$$[[[3,6], \frac{46}{1300}, 1, [3], \frac{1560}{11}], [[4,5,7], \frac{40}{1040}, 1, [4]]], [[[5,7], \frac{4}{130}, 0]], [[[6], 0.02, 0]],$$

$$[[[5,7], \frac{4}{130}, 0]], [[[7], \frac{2}{130}, 0]], [\,], [\,]]$$

- **Step 5:** Calculate overload
Before we start calculating the policy, we also want to calculate how much vehicles are already headed to a troubleroad (but not there yet). For this we make a list called overload to keep track of it. We calculate the overload as follows:

```
for r in Troubleroad:
  if the predecessor of r is not a choice point:
    k= sum (s in S(r), critdensity[s]*maxspeed[s])
    curRoutes=[[t,k/|P(r)|]]
    while curRoutes not empty:
      for m in curRoutes:
            overload[r]+=max[m[1]-(flow in m[0]),0]
            if predecessor of m[0] is not a choice point:
              for t predecessor of m[0]
                    add [t,m[1]/|P(m[0])|] to curRoutes
              remove m from curRoutes
```

For this step we only need to go over the the troubleroads of our network. These were roads $2, 3$ and 4. We see that all of these roads are preceded by a choice point. This means we do not need to calculate the overload for any of these roads, since the traffic could be sent over a different road.

- **Step 6:** Calculate policy
Finally we can calculate the policy. This is the only step that iterates over the timesteps as well as the number of roads, but since all ingredients for the computations have already been calculated, it is still quite fast.

```
for j in 1..J:
  for r in 1..R:
    if AllRoutes[r] has more than 1 element:
      for l in AllRoutes[r]:
            for l1 in TroubleRoutes:
              add as much traffic to l1 as is needed for l to becomes faster
            add as much to l until it would become jammed
            if l contains a troubleroad:
              add l to Troubleroutes
  calculate effect of policy
  recalculate the overload
```

Note that calculating the effect of the policy comes down to following the calculations in the previous section and recalculating the overload means repeating the last step.
We will now calculate a policy for our example for one timestep. For this example, we will consider an inflow into road 0 of 16900 veh/h. We will set $\Delta t = \frac{1}{60}$h (so one minute). The reason we only calculate one timestep is because the other timesteps will just be a repeat of the first with different numbers.

- We will now loop over all roads, starting with $r = 0$. We notice that AllRoutes[r] has multiple entries.
- We start by calculating, using formula (14), that the maximal outflow of road 0 is 16900 veh/h for this timestep. That means that this is the amount of traffic we want to distribute over the routes.
- We will first send as much traffic into the fastest route without it becoming congested. To do this, we consider the first troubleroad of that route, road 3. We see that the road(s) following road 3 can only take up a flow of $100 * 120 = 12000$ veh/h. thus, we will start by sending a flow of 12000 veh/h into the fastest route. A quick check tells us that the first road of that route (road 1) can indeed take up that much traffic. We now still need to distribute $16900 - 12000 = 4900$ veh/h.
- We now consider the second fastest route. We will first send as much traffic as possible into the fastest route until it becomes slower than the second fastest route. By adapting (8) we can calculate how much traffic we can send on this route before it gets too congested. After all, we see that

  $$q^{\text{in}}_{j,3} = (p_{j+1,3} - p_{j,3})\frac{X_3}{\Delta t} + q^{\text{out}}_{j,3}.$$

  We will assume that the outflow of this troubleroad is equal to the maximal supply of the following road(s). In this case, that is $v_6^{\max} p_6^{\text{crit}} = 120 * 100 = 12000$. All the other values can be found in tabel 10. Thus, we see that we can send

  $$(\frac{1560}{11} - 120)\frac{2}{\frac{1}{60}} + 100 * 120$$

  $$= \frac{160800}{11}$$

  flow into the fastest route (instead of the earlier calculated 12000). We see that road 1 can still take up this amount of traffic. This leaves us with

  $$16900 - \frac{160800}{11} = \frac{25100}{11} veh/h$$

  to distribute.

– We will now look at how much traffic we can send into the second fastest route before it becomes congested. We see that the first troubleroad of this route is road 4. Since its successor has multiple predecessors, we need to be a bit more careful here. We assume that we can send

$$\frac{v_4^{\max} p_4^{\text{crit}}}{\sum_{s \in P(5)} v_s^{\max} p_s^{\text{crit}}} v_5^{\max} p_5^{\text{crit}}$$

$$= \frac{80 * 100}{100 * 130 + 80 * 100} * 130 * 130$$

$$= \frac{135200}{21}.$$

into this route before it becomes congested (see Chapter 2.3.1). Since this number is larger than the amount of traffic we still need to distribute, we are done if we can send all of this over road 1 (the first road of the second fastest route). We see that this is indeed possible.

– In the end, we sent all of the outflow of road 0 to road 1.

– We now look at $r = 1$. We see that AllRoutes[1] also has multiple entries. We see that the maximum outflow of road 1 will be $130 * 130 = 16900$ this timestep. Thus, we will attempt to distribute this amount of traffic.

– We will first consider the fastest route after road 1. This is route $[[3, 6], \frac{46}{1300}, 1, [3], \frac{1560}{11}]$. Its first troubleroad is road 3. As seen before, this means we can send a flow of 12000 veh/h into this road before it will become congested. Road 3 can indeed take up this amount of traffic. This means we still need to distribute a flow of 4900 veh/h.

– We now consider the second fastest route. We will first send as much traffic as possible into the fastest route until it becomes slower than the second fastest route. With the same calculation as before, we see that we can send

$$(\frac{1560}{11} - 120)\frac{2}{\frac{1}{60}} + 100 * 120$$

$$= \frac{160800}{11}$$

flow into the fastest route (instead of the earlier calculated 12000). We see that road 3 can still take up this amount of traffic. This leaves us with

$$16900 - \frac{160800}{11} = \frac{25100}{11} \text{ veh/h}$$

to distribute.

– We will now look at how much traffic we can send into the second fastest route before it becomes congested. We see that the first troubleroad of this route is road 4. As before, we assume that we can send

$$\frac{v_4^{\max} p_4^{\text{crit}}}{\sum_{s \in P(5)} v_s^{\max} p_s^{\text{crit}}} v_5^{\max} p_5^{\text{crit}}$$

$$= \frac{80 * 100}{100 * 130 + 80 * 100} * 130 * 130$$

$$= \frac{135200}{21} \text{ veh/h}$$

into this route before it becomes congested. Since this number is larger than the amount of traffic we still need to distribute, we are done if we can send all of this over road 4 (the first road of the second fastest route). We see that this is indeed possible.

- Thus, we end up sending about 13.5% of the outflow of road 1 to road 4, and the remaining 86.5% to road 3.
- Since all other roads don't have more than one successor, we are done with calculating the policy for this step.
- We can now calculate the effect of our policy on each road. This is done as described in the paragraphs preceding this one.

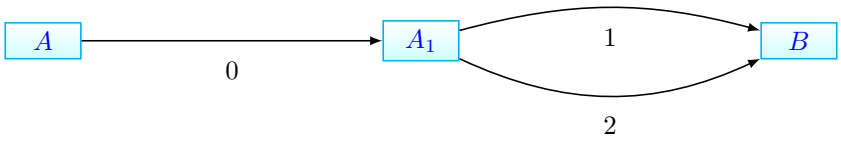


Figure 11: road network used for first and second test.

Roads	Road length	max speed	critical density	jam density	initial density
road 0	1 km	130 km/h	130 veh/km	260 veh/km	130 veh/km
road 1	1 km	130 km/h	130 veh/km	260 veh/km	120 veh/km
road 2	1 km	120 km/h	120 veh/km	240 veh/km	20 veh/km

Figure 12: Parameter values for the first test.

4 Tests and Results

4.1 First tests

In this chapter we will consider several scenario's and see what routing policies our program gives in all of these. We will start with some basic scenario's to get a better idea of how the program works and to verify whether it gives the optimal policy.
Our first scenario will feature the road setup of Figure 11.

In this example, A_1 is the relevant switch point where we have to choose between road 1 and road 2. The parameters used for our first test are described in Figure 12. With these values we see that road 1 is faster than road 2. After all, they have the same length, but one can drive faster on road 1. First of, we see that with an inflow of 15000 veh/h, the program will send all traffic over road 1. This is also what we want, since road 1 can handle that amount of traffic and is the fastest route to take.
Now that we have tested the easiest case, we will consider what happens when we increase the inflow. To do this, we need to change the values so that road 0 can now handle twice as much traffic. See Figure 13 for the new parameters. The outflow of road 0 can now be more than road 1 can take in. We'd expect the program to start using road 2 only when there is to much traffic for road 1. We can easily calculate using formula (5) that the maximal supply of road 1 is

$$v_1^{\max} p_1^{\text{crit}} = 130 * 130 = 16900.$$

Since the maximal outflow of road 0 is exactly twice this amount, namely

$$v_0^{\max} p_0^{\text{crit}} = 130 * 260 = 33800,$$

We expect the program to send exactly half the traffic to road 1 after a while, and the other half to road 2.

As we can see in Figure 14, our assumption is indeed correct. While it takes road 0 a little while to fill up, the program eventually starts sending exactly half the traffic to road 1, and the other half to road 2.

Roads	Road length	max speed	critical density	jam density	initial density
road 0	1 km	130 km/h	260 veh/km	520 veh/km	150 veh/km
road 1	1 km	130 km/h	130 veh/km	260 veh/km	120 veh/km
road 2	1 km	120 km/h	120 veh/km	240 veh/km	20 veh/km

Figure 13: Parameter values for the second test.

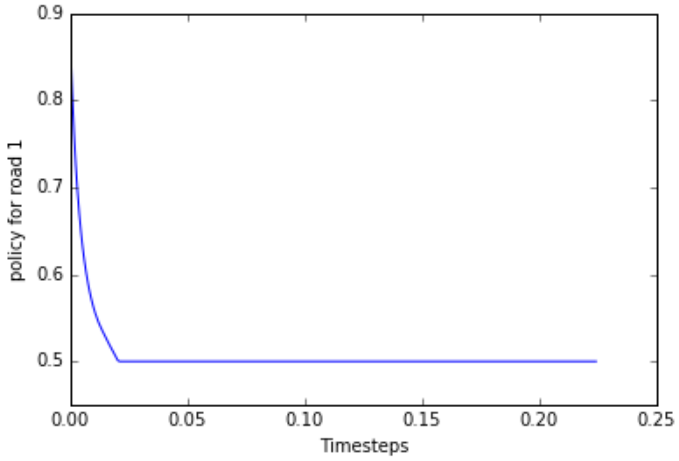


Figure 14: Percentage of traffic sent to road 1 over time when there is an overload of traffic.

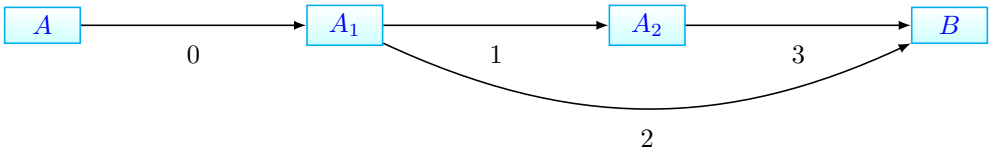


Figure 15: Road network used in test with bottleneck.

4.1.1 Adding a bottleneck

In both of the examples above, there is no traffic jam occurring, because there is no bottleneck. While this can still give some insight in the program, it doesn't account for any interesting case. We will thus create a bottleneck in the model for our next test. The setup is shown in Figure 15.

The initial parameters a listed in Figure 16. The idea is that road 3 cannot take in as much traffic as road 1, so if road 1 is used to much, a traffic jam will appear, making it a slower route than road 2 (which is slower if all roads are in free flow). Thus, we expect the program to start using road 2 earlier than it did before. For instance, even a flow of 15000 veh/h would already be more than the capacity of road 3 which we can calculate, as done before, to be 11310 veh/h. Thus, we expect only $\frac{11310}{15000} = 0.745 = 74.5\%$ of the traffic to be sent to road 1.

Looking at the results, we see that our expectation was indeed met. The most interesting thing

Roads	Road length	max speed	critical density	jam density	initial density
road 0	1 km	130 km/h	260 veh/km	520 veh/km	120 veh/km
road 1	1 km	130 km/h	130 veh/km	260 veh/km	120 veh/km
road 2	1.2 km	120 km/h	120 veh/km	240 veh/km	20 veh/km
road 3	0.2 km	130 km/h	87 veh/km	174 veh/km	10 veh/km

Figure 16: Parameter values for the first test with bottleneck.

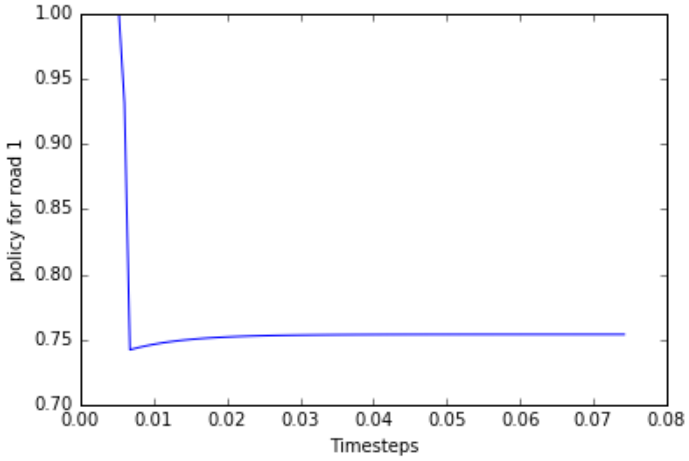


Figure 17: Percentage of traffic sent to road 1 over time when there is a bottleneck.

about this is that it seems to approach a certain limit distribution. We will now try to figure out the how changing the parameters affects this limit distribution.
As explained in the previous chapter, the program tries to maintain a certain density in road 1, by sending in exactly as much traffic as there is leaving road 1. Recall that the density it is trying to maintain can be calculated by using formula (38):

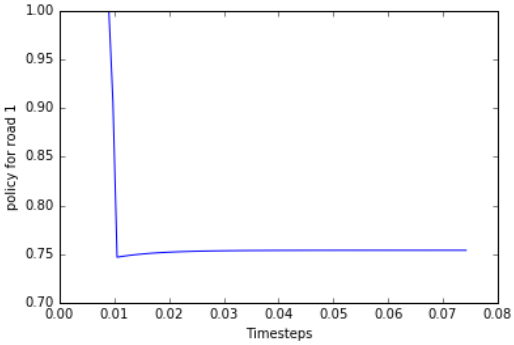
$$p^{reroute} := \frac{(T_2 - T_1 + t_r)p_r^{\text{jam}} v_r^{\max} p_r^{\text{crit}}}{X_r(p_r^{\text{jam}} - p_r^{\text{crit}}) + (T_2 - T_1 + t_r)v_r^{\max} p_r^{\text{crit}}}.$$

Considering this, increasing the maximum speed or critical density on road 1 will increase the rerouting density, but it will not affect the limit distribution. This is because the limit distribution is dependent on how much traffic can leave road 1, not on how congested road 1 can become. It will however take longer to reach the limit distribution.
Figure 18 shows the result of trying to change several parameters. As we can see, only Figure 18d seems to have significantly changed from the original test in Figure 17. Thus, we can conclude that the limit distribution is indeed dependent only on the capacity of road 3. We're also interested in how the amount of inflow affects the distribution of traffic. In Figure 19 we see the result of taking the previous set up and gradually increasing the inflow.
As we can see, it starts by using the fastest route (road 1+3) until road 1 becomes congested. It then uses road 2 more and more, until we reach the capacity of road 2. After that it starts using the first route again since the traffic needs to go somewhere, but at this point we have already reached the capacity of the network.

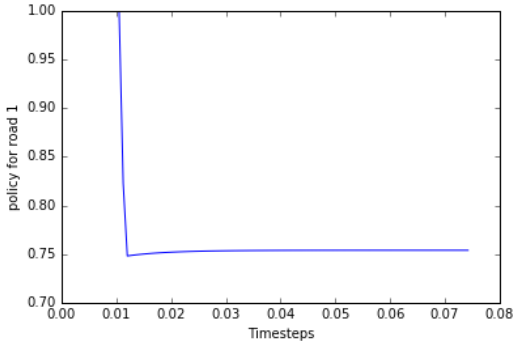
4.1.2 More choice options

Now that we have a good idea about how the program works when there is only one crossing, it is time to consider a case with multiple crossings. For this, we will consider the road structure from Figure 20. While the second crossing will just act like we have seen before, it is of interest how the program makes decisions for the first crossing, now that there will be a second decision following it. Using the parameters given in Figure 21 and an inflow of 20000 veh/h, we get the results shown in Figure 22.

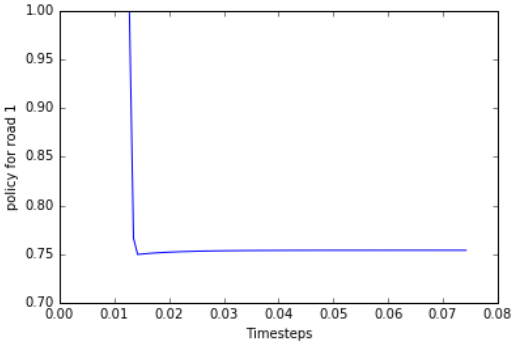
We see a difference appearing now. The program still uses alternative routes, but unlike before it does not seem to converge to a certain distribution. The reason for this is that it does not



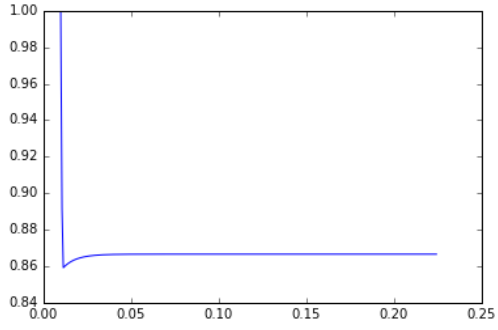
(a) Percentage of traffic sent to road 1 over time after increasing maximum speed of road 1.



(b) Percentage of traffic sent to road 1 over time after increasing critical density of road 1.



(c) Percentage of traffic sent to road 1 over time after increasing the length of road 1.



(d) Percentage of traffic sent to road 1 over time after increasing capacity of road 3.

Figure 18: Testing different parameters.

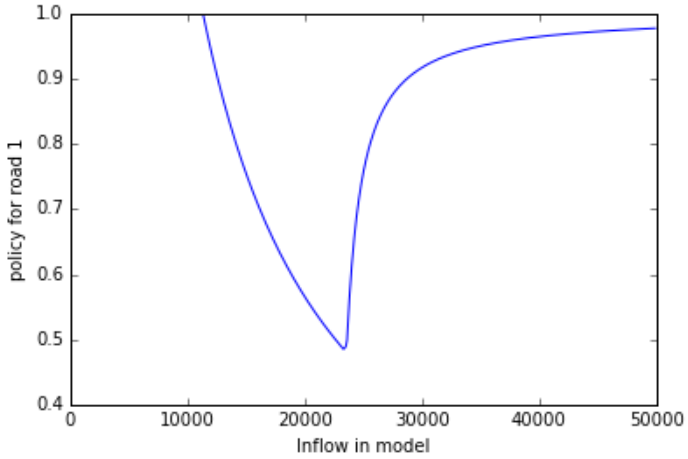


Figure 19: Testing the limit distribution against the inflow.

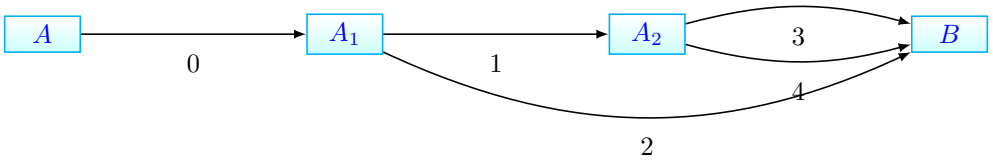


Figure 20: Road network used in test with multiple crossings.

Roads	Road length	max speed	critical density	jam density	initial density
road 0	1 km	130 km/h	130 veh/km	260 veh/km	120 veh/km
road 1	1 km	130 km/h	130 veh/km	260 veh/km	120 veh/km
road 2	2.2 km	120 km/h	130 veh/km	260 veh/km	20 veh/km
road 3	1 km	130 km/h	130 veh/km	260 veh/km	120 veh/km
road 4	1.2 km	100 km/h	120 veh/km	240 veh/km	20 veh/km
road 5	0.2 km	130 km/h	87 veh/km	174 veh/km	10 veh/km

Figure 21: Parameters for the third model with multiple crossings.

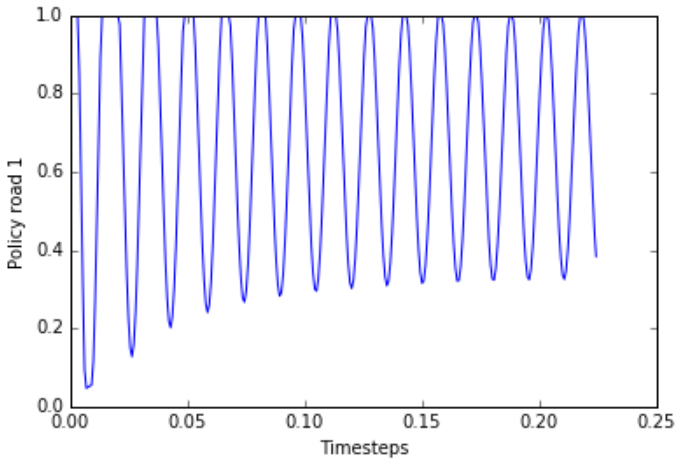
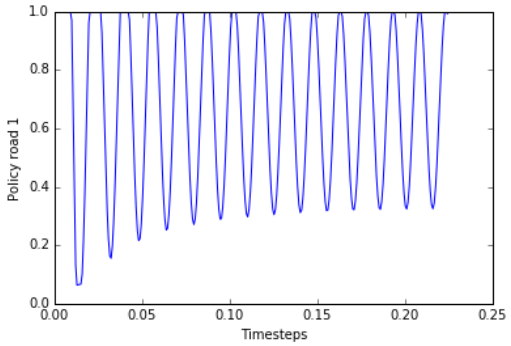
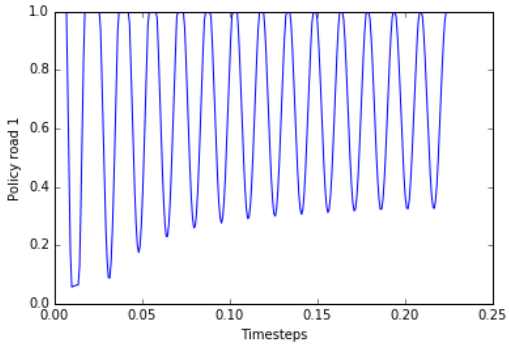


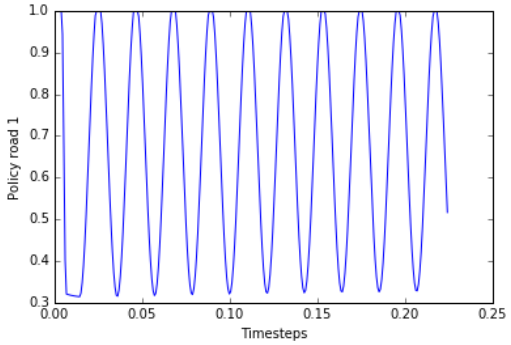
Figure 22: Percentage of traffic sent to road 1 over time when there are multiple crossings.



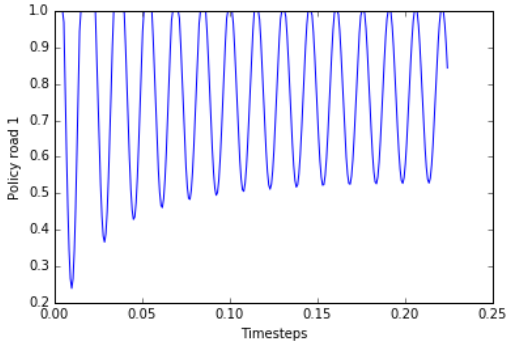
(a) Percentage of traffic sent to road 1 over time after increasing maximum speed of road 3.



(b) Percentage of traffic sent to road 1 over time after increasing the length of road 3.



(c) Percentage of traffic sent to road 1 over time after increasing the length of road 1.



(d) Percentage of traffic sent to road 1 over time after increasing critical density of road 5.

Figure 23: Testing different parameters for multiple switch points.

take road 1 into account. Thus, it will only start rerouting when road 3 starts getting jammed, but there is still a lot of traffic that was routed to road 3. If the traffic jam gets too bad, road 4 will of course be used, but it is still not the optimal distribution. Instead of reaching a certain limit distribution, the distribution seems to be oscillating. Figure 23 shows the result of several tests we conducted to get a better feeling for this oscillation.

As can be seen from these tests, changing the parameters of road 3 doesn't impact the oscillation significantly, but changing the length of road 1 (and road 2 to not suddenly make it faster) does. After all, if road 1 is longer, it means that there will be even more time between when choices are made in the first crossing and when we see the result of those choices. This results in slower and heavier actions. We also see that increasing the capacity of road 5 does not affect the oscillation rate, although it does mean less traffic has to be rerouted. All in all, we can conclude that the heaviness of the oscillation is connected with the length too the congested road. Thus, the program will become less reliable with more crossings. One should however note that the distribution right before the congested road will be optimal, which limits how bad the given solution can become.

5 Test Case Ouwehands Dierenpark

Until now, we have only used very basic examples to show how the program works. We will now show a test case to give an example of how it can be used in a real life situation.

This test case was performed for TrafficLink, after they got a request from Ouwehands Dierenpark to think about an app to regulate the traffic around their facility. The traffic situation in the area

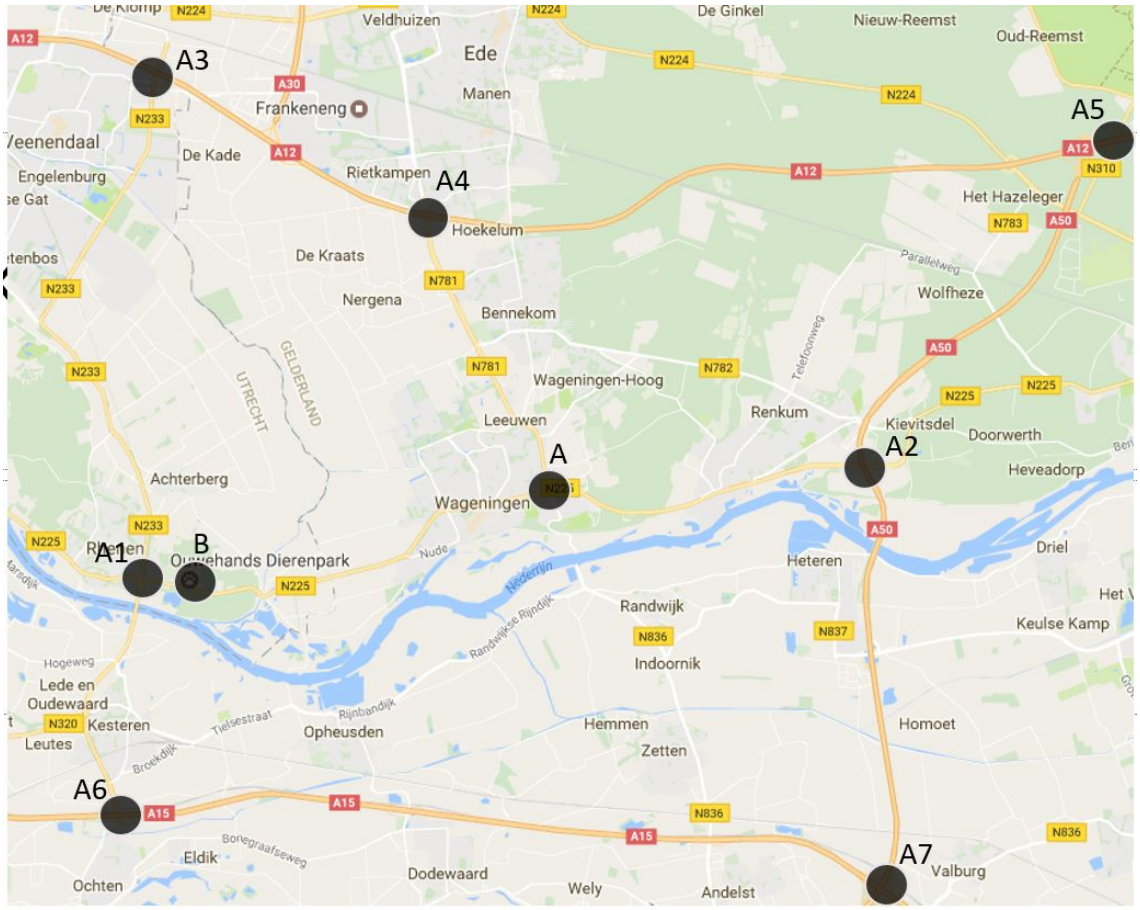


Figure 24: Picture of the road structure surrounding Ouwehand Dierenpark.

is shown in Figure 24, where point B is the location of Ouwehands Dierenpark.

The problem was that the N233 was getting congested, so Ouwehands wanted to solve this by either having some visitors use a different route, or managing when each visitor leaves. We will now focus on how this program can be used to distribute visitors on their way to Ouwehands Dierenpark.

First off, we will have to make the road structure into a graph. To do this, we must consider the most prominent routes to Ouwehands. For visitors from the west of the country, there aren't to many options apart from the N233, but visitors from the east can also take the N225. Taking all of this into consideration, we get a graph as shown in Figure 25. We see that this graph contains three crossings: the choice between road 4 and 5 at point A_4, the choice between road 6 and 7 at point A_5 and finally the choice between road 9 and 10 at point A_7. It is important to note that this graph does not contain loops, as that would make the program unable to find all routes to the endpoint. We also see that we can introduce multiple starting points, although we cannot use multiple endpoints. In reality, each road will be split up in several segments of 25m for more accurate calculations, but since this would make for a very chaotic picture, we have decided to include the above one instead.

Now that we have the overall road structure, we need to obtain values for all of our variables. We will go over them in order.

- The length and maximum speed of each road (part) can simply be looked up. Thus, these

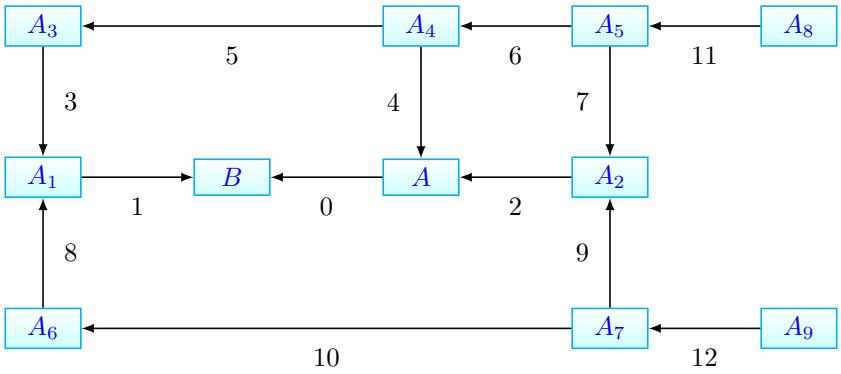
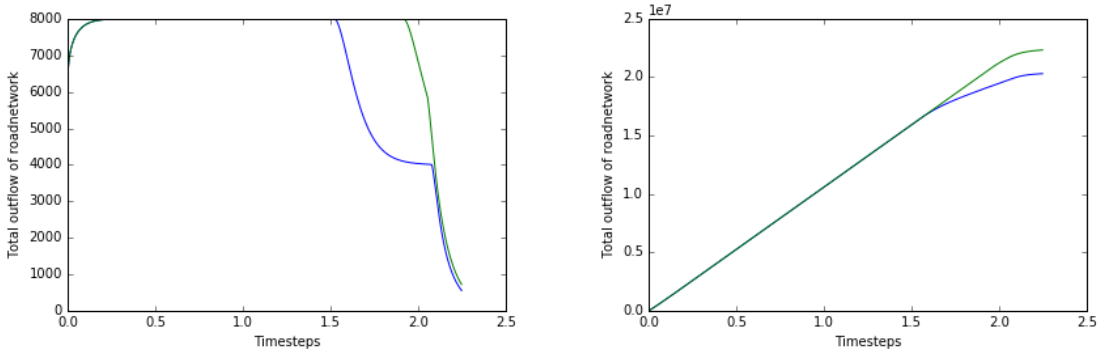


Figure 25: Graph representation of the road structure surrounding Ouwehands Dierenpark.



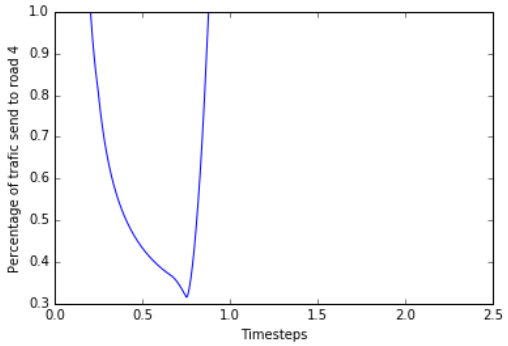
(a) Total outflow of the network per timestep. Green is full control, blue is no control

(b) Total cumulative outflow of the network over time. Full control is green, no control is blue.

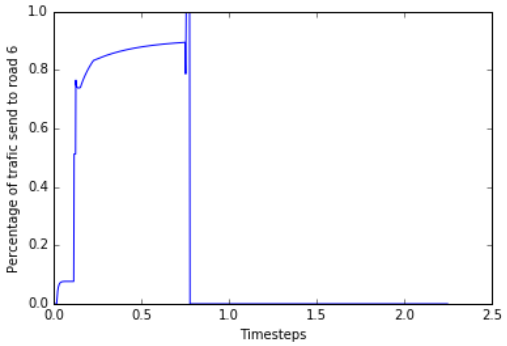
Figure 26: Results of testing on Ouwehands model with high overload.

variables are easy to obtain.

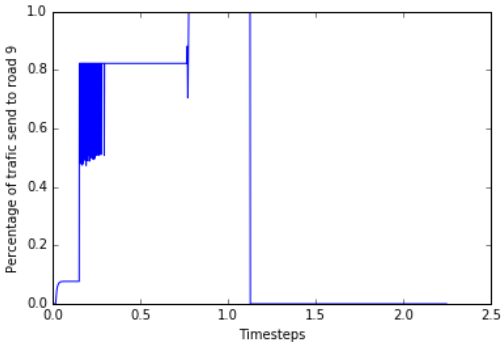
- The critical density and jam density have to be calculated. This can either be done by predicting the values based on the number of lanes of the road and the maximum speed, or by predicting the values based on a large number of pairs of velocity and density. While the second method is more accurate, it requires a large amount of data to be used, while the first method can be done with data that can easily be looked up.
- When we're only interested in calculating policies to be immediately used (say, for the next half hour), we can calculate the inflow into starting points by considering the current flow. If we want to calculate policies further ahead though, we will need to make a historic profile of this inflow to obtain a good prediction.
- For the initial density, we will need access to real time information about the traffic situation in our area of interest. fortunately, TrafficLink does have this access.
- The split percentages will have to be calculated using historical data of the flow at the relevant switch points.
- Finally, the background percentage can be calculated by considering the number of cars on a given road part and the number of app-users on that road part.



(a) Percentage of traffic sent to road 4 over time.



(b) Percentage of traffic sent to road 6 over time.



(c) Percentage of traffic sent to road 9 over time.

Figure 27: Routing policies of the Ouwehands test case with high overload.

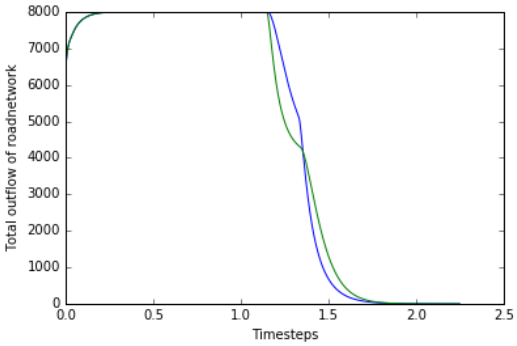


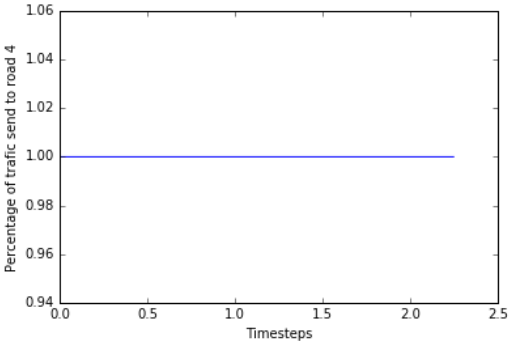
Figure 28: Total outflow of the network per timestep with a low overload. Green is full control, blue is no control

Roads	inflow
road 0	0 veh/h
road 1	5000 veh/h
road 2	5000 veh/h
road 3	10000 veh/h
road 4	0 veh/h
road 5	0 veh/h
road 6	0 veh/h
road 7	0 veh/h
road 8	5000 veh/h
road 9	0 veh/h
road 10	0 veh/h
road 11	20000 veh/h
road 12	20000 veh/h

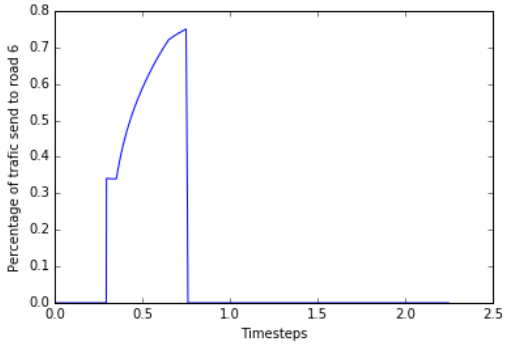
Figure 29: Inflow for the Ouwehands test case with high overload.

Roads	inflow
road 0	0 veh/h
road 1	1000 veh/h
road 2	1000 veh/h
road 3	1000 veh/h
road 4	0 veh/h
road 5	0 veh/h
road 6	0 veh/h
road 7	0 veh/h
road 8	1000 veh/h
road 9	0 veh/h
road 10	0 veh/h
road 11	5000 veh/h
road 12	5000 veh/h

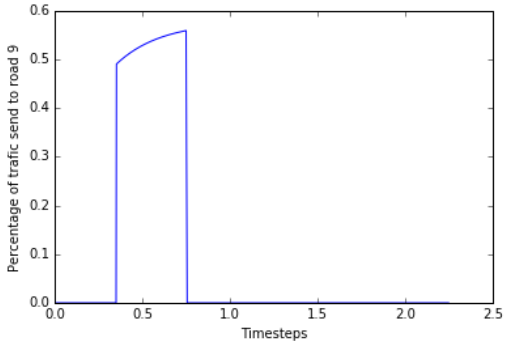
Figure 30: Inflow for the Ouwehands test case with low overload.



(a) Percentage of traffic sent to road 4 over time.



(b) Percentage of traffic sent to road 6 over time.



(c) Percentage of traffic sent to road 9 over time.

Figure 31: Routing policies of the Ouwehands test case for a low overload.

Roads	Road length	max speed	critical density	jam density	initial density
road 0	5.5 km	80 km/h	50 veh/km	260 veh/km	40 veh/km
road 1	1 km	80 km/h	50 veh/km	260 veh/km	40 veh/km
road 2	5 km	80 km/h	50 veh/km	260 veh/km	20 veh/km
road 3	8 km	80 km/h	50 veh/km	260 veh/km	40 veh/km
road 4	6 km	80 km/h	50 veh/km	260 veh/km	20 veh/km
road 5	5 km	130 km/h	60 veh/km	400 veh/km	10 veh/km
road 6	12 km	130 km/h	60 veh/km	400 veh/km	10 veh/km
road 7	6 km	120 km/h	60 veh/km	260 veh/km	10 veh/km
road 8	4 km	80 km/h	50 veh/km	400 veh/km	40 veh/km
road 9	8 km	120 km/h	60 veh/km	400 veh/km	10 veh/km
road 10	12 km	120 km/h	60 veh/km	400 veh/km	10 veh/km
road 11	1 km	130 km/h	60 veh/km	400 veh/km	10 veh/km
road 12	1 km	130 km/h	60 veh/km	400 veh/km	10 veh/km

Figure 32: Parameters used for the Ouwehands test case.

In Figure 26 we see the combined outflow of roads 0 and 1 set out against the time. The parameters used for this test can be found in Figure 32. We see that in the case of full control, the output is higher at the end. This means there are more vehicles leaving the model, which means more drivers are arriving at their destination. The reason that the outflow stays lower in the case of no control is because there are fewer vehicles entering the model overall. After all, because everyone tries to take the fastest route, these roads become congested and fewer vehicles can enter a congested road. Because the algorithm distributes the vehicles better over the whole road network, the full control case can take up more vehicles, resulting in a higher outflow. Figure 27 shows the routing policies used for this situation. Since there are only three crossings in the network, we can show the entire policy by only showing the policies of road $4, 6$ and 9. The used inflow can be found in Figure 29

In Figure 28 we see the results of another test on the same model but with a lower overload. As we can see, there's not much of a difference between the outflow of the no control case and the full control case. The reason for this is that everyone taking the fastest route is already very close to the optimal policy in this case. If we look at the policies calculated for this case in Figure 31, we see that the actions taken by the algorithm are much less severe than in the previous case. The used inflow can be found in Figure 30

Since the difference between using the algorithm and not interfering is much larger in the case of a large overload, we can conclude that the algorithm is best used when there's a large overload, instead of a low one.

6 Conclusion

6.1 Conclusions

To do a short recap, we have first looked at a model presented in [1] to redirect traffic. After that, we greatly expanded this model step by step. We also rewrote our model into an IP-formulation, but after finding out that solving the model was very slow, we derived a heuristic. After that, we did several tests with this heuristic to see how it worked and how much better it would be compared to not doing anything at all. Our results show that our algorithm is very effective compared to no intervention when there is a high overload on the network. Since our model is very general, it can also be used on most road networks. It can be used to solve routing problems on networks all over the world and can also be used as an effective tool for analytic research.

We have seen that the algorithm still has some problems when there are multiple switchpoints in a row, with no bottlenecks in between. However, even in these cases it still gives a better policy than no intervention would give, and because of the abundance of inflow points in most networks, it is not a very common occurrence. Thus, while the algorithm can still be improved upon, we can recommend it for use anyway.

6.2 Future research

In this section, we will give recommendations for future research.

- First off, we have to mention that a heuristic is never a perfect solution. While it is possible to have an IP-Solver solve the model, it is to slow to be put to practical use. Ideally, one would find a way to calculate the ideal solution fast enough for practical use.
- Other than that, the heuristic presented in this thesis has some flaws on it own. As we have seen, it starts oscillating when there are multiple switch points before a bottleneck. Thus, further research is required to find a way to remedy this.
- Finally, the heuristic in this thesis will always sent drivers to their fastest route, even if this causes a traffic jam. One could also try to use the model presented here to make a heuristic that never goes above the capacity of a route so long as this is possible. This will certainly be a good way to stop congestion from happening on the road, but it can be argued whether sending drivers over a slower route than what is available is preferable or not. It is of course worth looking into nonetheless.

References

[1] Jan Klein, *Modelling Incident Management*, 2016

[2] Greenshield B.D. *The photographic method of studying trac behavior*, in: Proceedings of the 13th Annual Meeting of the Highway Research Board, pp. 382399, 1934

[3] Daganzo, C. F. *The cell transmission model: A dynamic representation of highway trac consistent with the hydrodynamic theory*, Transportation Research Part B: Methodological, 28(4), pp. 269287. , 1994

[4] smulders S. *Control of freeway trac ow by variable speed signs*, Transportation Research Part B: Methodological, 24(2), pp. 111132, 1990

[5] Jochem Braakman, *Evacuation Policies for Events*, 2016

[6] S. Travis Waller, David Fajardo, Melissa Duell, Vinayak Dixit *Linear Programming Formulation for Strategic Dynamic Traffic Assignment*, 2013

[7] José R. Correa, Nicolás E. Stier-Moses *Wardrop Equilibria*

[8] Athanasios K. Ziliaskopoulos
A Linear Programming Model for the Single Destination System Optimum Dynamic Traffic Assignment Problem