TRR
JOURNAL OF THE TRANSPORTATION RESEARCH BOARD

Research Article

Transportation Research Record
1–10

DOI: 10.1177/03611981211000348
journals.sagepub.com/home/trr
SAGE

Online and Proactive Vehicle Rerouting with Uppaal Stratego

Alexander Bilgram[1], Emil Ernstsen[1], Peter Greve[1], Harry Lahrmann[2], Kim G. Larsen[1], Marco Muñiz[1], Peter Taankvist[1], and Thomas Pedersen[1]

Abstract

Modern navigation systems warn the user of traffic jams ahead and suggest alternative routes. However, a lemming effect can cause the alternative routes also to become congested, as the system suggests the same route to all users. As such, in an attempt to optimize for the individual driver, the welfare of the traffic network is punished. In this paper we introduce an online and proactive method for collective rerouting recommendations based on real-time data and stochastic optimization. Our system periodically monitors the status of the network to identify potentially congested roads together with vehicles affected by them. The system then uses Uppaal Stratego to perform machine learning and approximate the best rerouting scenarios. As a proof of concept, we build a SUMO model of a representative traffic network. We perform exhaustive experiments considering different traffic loads and different traffic light controllers. Our results are promising, showing considerable improvement in travel times, queue lengths, and CO_2 emissions.

As cities and populations grow, so do traffic and traffic congestion. Congestion can be caused by road works, malfunctioning traffic lights, accidents, and so forth, but also by too much traffic in road networks ill equipped to handle it. Modern navigation systems might propose alternative routes to users, but may suggest the same route to several vehicles, causing congestion on the alternative route. Congestion in urban traffic networks has several negative effects on the environment and society. Thus, reducing congestion is an important goal. There have been several initiatives to address congestion in Denmark. For example, in (*1*) the Danish Congestion Commission called for improved traffic signal control to reduce congestion. Additionally, rerouting strategies to avoid congestion are being actively investigated, as seen in (*2*–*5*).

In this work we propose an online and proactive method for collective reroute recommendations based on real-time data and stochastic optimization. Our models are stochastic to represent the choice of the drivers to accept or reject a recommended route. Our system periodically monitors the status of the network to identify potentially congested roads together with vehicles affected by them. For example these data could be gathered as seen in Work et al. (*6*). Given the current observations on the network, our system will build a David et al. (*7*) model. The tool will perform simulations and apply machine learning techniques to approximate the best rerouting scenarios. The solution avoids the lemming effect experienced in current solutions, as this will result in a worse optimization value. The solution will seek to find the optimal solution—with regard to average travel time—between rerouting to an alternative route and continuing on the same route.

Problem Definition

In this work we consider urban traffic networks with multiple intersections, each with traffic lights. We assume that the vehicles in the network have a current route and destination. For each of these vehicles there is the possibility to recommend a new route, for example, via a GPS. Vehicles will accept or decline the proposed route with some probability. Different route assignments to vehicles will induce different measurements, for example, waiting times, queue lengths, and so forth. In this work

[1]Department of Computer Science, Aalborg University, Denmark
[2]Department of Civil Engineering, Aalborg University, Denmark

Corresponding Authors:
Peter Taankvist, ptaank16@student.aau.dk
Marco Muñiz, muniz@cs.aau.dk

we would like to approximate the best set of route assignments using stochastic models and machine learning.

Related Work

There are multiple proposals for dissolving traffic congestions such as Uppaal Stratego intelligent traffic lights (*8*). However, the traffic lights presented in the paper do not communicate in any way and therefore only look at a single intersection. There are also services such as Google Maps and Waze that suggest new routes based on events such as accidents or congestions. However, both of these are reactive and not preventive. Furthermore, they suggest the same alternative routes to all vehicles using the application resulting in a lemming effect. As such, an increase in the number of users could become problematic (*9*). We instead attempt to distribute traffic so that more users only contribute to the system's performance. In Seongmoon Kim and White (*5*) they route vehicles according to real-time traffic information. The authors make optimal routing policies under time varying traffic flows based on a Markov decision process formulation. In Cao et al. (*3*), the authors introduce a pheromone-based framework for reducing traffic congestion in metropolitan cities. In Pan et al. (*4*), three different rerouting algorithms, based on live data, are presented. The algorithms attempt to distribute traffic across a network to reduce travel times. The algorithms presented all show promising results, giving motivation for researching other approaches as well. As opposed to these approaches we use techniques from model checking and machine learning as well as considering environmental conditions and different traffic light controllers.

Outline

The next section will cover the implementation and look in depth at how the rerouting model is made. This includes the implementation of the two sub-models, preliminaries for the models, and how we apply the traffic lights presented in Eriksen et al. (*8*). We then introduce our case study. This is what the experiments will be conducted on. Thereafter, we present the results of the experiments we conducted and the measurements we use. Here we present the results of the experiments without closed roads, followed by the results of experiments where some roads are closed. Lastly, we conclude and describe concepts for further work.

Vehicle Rerouting with Uppaal Stratego

In this section we explain the high level idea of our solution. First, we briefly describe the tool Uppaal Stratego and the stochastic models we use for finding near-optimal rerouting strategies. We then give a high level idea of how our rerouting system works. This is followed by a description of the preparations required before Uppaal Stratego is invoked on a model generated online. Finally, we describe in detail the Uppaal Stratego model for synthesizing near-optimal rerouting strategies.

Uppaal Stratego

Uppaal Stratego is part of the Uppaal Stratego tool suite, which are integrated tool environments for modeling, validating, and verifying real-time systems modeled as networks of timed automata (*10*). The tool has been successfully applied to multiple cases of cyber-physical systems such as online floor heating (*11*), adaptive cruise control (*12*), and intelligent traffic lights (*8*). The tool combines techniques from both model checking and machine learning to synthesize near-optimal strategies for a given scenario. The idea behind Uppaal Stratego is that we can state certain guarantees, for example, safety or liveness using model checking, while optimizing the outcome of the problem using machine learning. For example, in Larsen et al. (*12*) the authors create a model that guarantees that a vehicle does not collide with the vehicle in front, while minimizing the travel time of the given vehicle.

In our case, we compute a near-optimal rerouting strategy for a given configuration of vehicles in a traffic network. This is done by feeding the Uppaal Stratego models a snapshot of the current traffic situation. Uppaal Stratego will use reinforced learning to synthesize a near-optimal solution to the problem. It does this by either minimizing or maximizing a value. We attempt to minimize the average travel time through the traffic network.

The Uppaal Stratego models can be seen as one and a half player games, in which a controller plays against the environment. In our case the controller only has the option to suggest a rerouting option. The environment is everything that can affect this choice such as the probability that drivers accept the rerouting suggestion, road conditions, speed, and so forth.

Example. Figure 1 shows an example Uppaal Stratego model, reflecting the choice of whether or not to reroute a vehicle. In the first location, Chooser, the controller has a choice between going to DoReroute or NoReroute represented by the solid arrows. If it chooses the former, it proceeds to a branch point representing a probability of a driver accepting/rejecting the suggested route, where p is the probability that the driver accepts the suggestion. At this point any transition is uncontrollable and is represented by the dashed arrows. Assuming every vehicle is considered according to this Uppaal Stratego model,

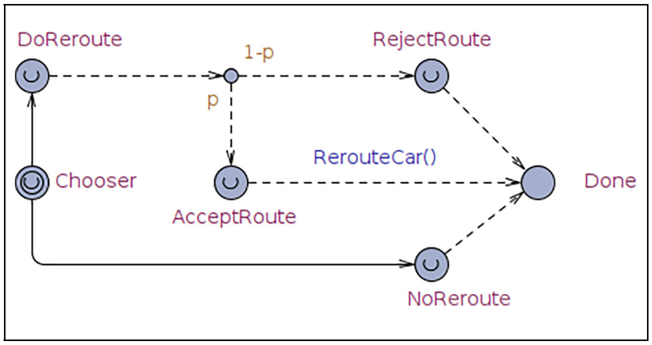

Figure 1. Example Uppaal Stratego model.

Uppaal Stratego estimates a near-optimal strategy of who to recommend rerouting to. The model presented below reflects exactly this.

Vehicle Rerouting Intuition

Formally, we implement a stochastic hybrid game as defined in Larsen et al. (*11*), in which the controllable modes are route suggestions in conjunction with all possible traffic light phases for each traffic light. The uncontrollable modes are the environment actions such as the flow of the vehicles. A high level description of the implementation of our rerouting system is given in Algorithm 1 together with Algorithm 2.

In Algorithm 1 we observe, that every 10 time units (this can be easily modified) the system monitors the status of the network. It then estimates which roads are congested and detects the affected vehicles which are thus flagged for rerouting. This step is explained in more detail below. With this information, a Uppaal Stratego model is built and used to find near-optimal rerouting strategies. Algorithm 2 describes this step. Finally, the system will propose the new routes to vehicles flagged for rerouting. These vehicles can accept or reject the proposed routes with some probability.

Preparations for Model Generation

To generate rerouting strategies. We periodically generate a Uppaal Stratego model with the current state of the network. Computing strategies is computationally expensive, thus we need several preparations to reduce computation done by the model. We have two major considerations. First, since computational complexity increases exponentially with the number of choices, we identify and mark a limited number of vehicles for rerouting. Second, Uppaal Stratego recommends routes dynamically, for this it uses Dijkstra's shortest path algorithm. To speed up this computation an adjacency matrix is precomputed. In what follows we describe how estimation of congested roads and flagging vehicles is computed. We then describe how the Uppaal Stratego model is generated.

Congested Roads and Flagged Vehicles. Which vehicles are flagged for rerouting is essential for the system, and dependent on the congestion of roads. In our system this is done by continually updating the current weight of the roads according to a case specific weight function (defined below). When flagging vehicles we look two road segments ahead and consider their current weight. If the weight of either of the two road segments is above

Algorithm 1. High Level Explanation of Online and Proactive Vehicle Rerouting

```
while True // as long as the traffic network is functioning
  Control trafficlights // e.g., use Algorithm 1 from Paper (8)
  if Time % 10 ==0 then //every 10 time units, observe the status of the system
      Predict congested roads
      Detect affected vehicles // these vehicles are flaged for rerouting
      Compute near-optimal routes // create model and call Algorithm 2
      Suggest new routes in traffic network
  end if
end while
```

Algorithm 2 High level explanation of our model

```
//The status of the traffic network is represented in a rerouter model
  for all vehicles
      if a vehicle has a rerouting flag then
        A new route is calculated using Dijkstra's algorithm
        The vehicle accepts the new route with a fixed probability
      else
        The vehicle keeps its current route
      end if
  end for
Simulate traffic until the horizon is reached
```

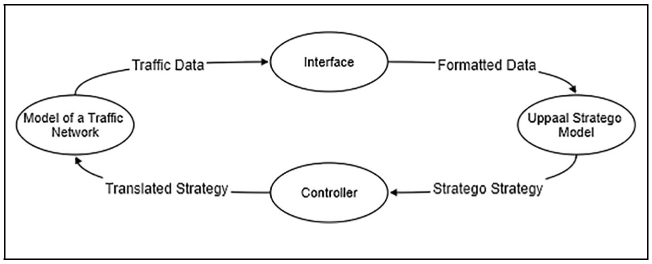

Figure 2. Interaction with a model of a traffic network.

a case specific threshold (also defined below) we flag the vehicle as a potential rerouting possibility.

Model Generation. Figure 2 shows the interaction between the traffic network and Uppaal Stratego. Our method periodically generates a Uppaal Stratego model which is fed with Formatted Data which includes information such as: the flagged vehicles, the road network represented as a directed weighted graph, the number of vehicles per edge, a time horizon for which to simulate, and so forth. This model is used for generating a strategy which is then implemented in the traffic network via a controller.

Models

A Uppaal Stratego model for rerouting is the composition of two internal models. These internal models include the Simulator model responsible for simulating traffic flow, and the Rerouter model responsible for route suggestions.

Simulator Model. The entry point of the model is the Simulator model, seen in Figure 3. All transitions in this model are uncontrollable. Initially, the Simulator model synchronizes with the Rerouter model via the Reroute! channel. The main functionality of the Simulator model revolves around simulating traffic after the Rerouter model has yielded a reroute suggestion strategy. In the location, SimulateTraffic, the model flows the traffic and updates the edge weights every 10 time units until either the horizon is reached or all vehicles have reached their destination. Lastly, the transition to End calculates the sum of the average travel time per edge. Formally shown in Equation 1.

$$\sum_{i=1}^{V} \frac{traveltime_i}{numEdges_i} \tag{1}$$

where

V is the number of vehicles,

$traveltime_i$ is the i -th vehicle's travel time, and

$numEdges_i$ is the number of edges the i -th vehicle has traversed.

This summation is performed to provide a value for Uppaal Stratego to minimize.

Rerouter Model. The Rerouter model seen in Figure 4 is in control of suggesting new routes to vehicles that are flagged. The internal logic in this model is similar to the example seen in Figure 1. It is invoked when it can synchronize on the Reroute? channel. The Rerouter model then goes through all vehicles, checking if they are flagged for rerouting. The vehicles can be rerouted in any arbitrary order, but to reduce computation time we suggest reroutes in a sequential manner. This is possible because the vehicles are moved after they have been rerouted. If Uppaal Stratego chooses to reroute the vehicle, choose_route() finds a route for the vehicle, according to a Dijkstra's shortest path algorithm. If the route of the vehicle contains a closed edge then the vehicle has to receive a new route, and the choice is bypassed. Although, the probability remains that the vehicle accepts or declines the route suggestion. If it declines it will keep the route it already has. At the CarDone location, we check if all vehicles have been processed, at which point the transition to done is taken, and the model synchronizes with the Simulator model.

Traffic Light Controllers

In the traffic network there are several intersections. We assume that there is a traffic light at every intersection. We consider two different types of traffic light controllers. We consider static time traffic light controllers, where the traffic light follows a fixed time cycle of red/amber/green phases, and a smart traffic light controller from Eriksen et al. (*8*). The smart traffic light controllers are based on Uppaal Stratego and are traffic controlled traffic lights (contrary to time controlled traffic lights), that is, the traffic light controller adapts to the current traffic flow and changes the phases accordingly to maximize the throughput of the intersection. This is done to measure the performance both with and against a solution for smarter traffic management. To properly use the traffic light controller from Eriksen et al. (*8*) in our scenario, we modify it to consider if the outgoing lanes are already congested. Furthermore, for fast computations, traffic lights are executed concurrently using multiple threads.

Case Study

As a proof of concept we consider a representative traffic network modeled in the simulation tool SUMO (*13*). SUMO is a tool for microscopic traffic simulation.

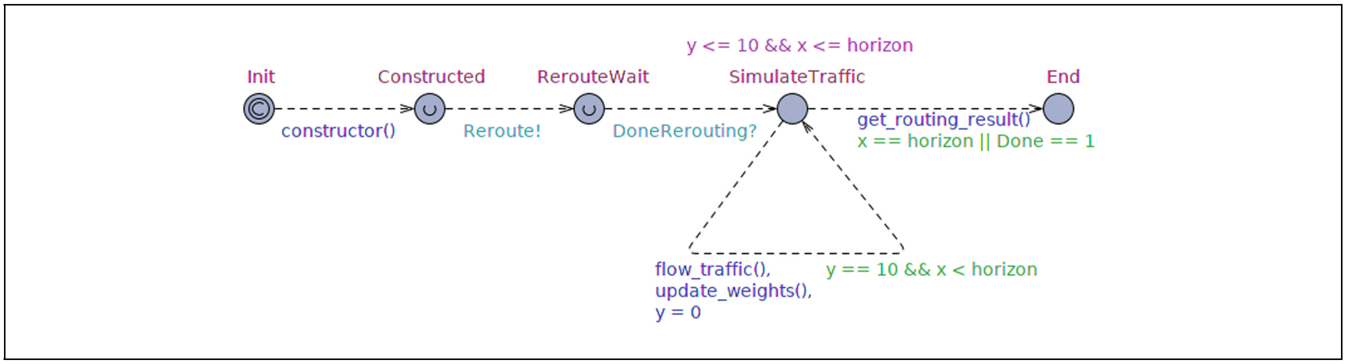

Figure 3. Model used for simulating traffic.

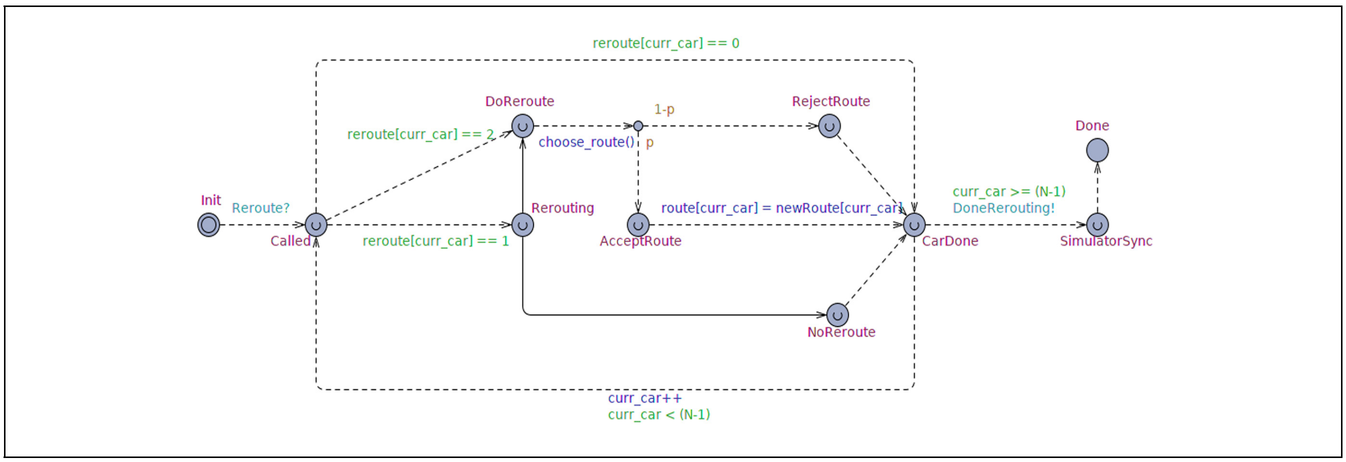

Figure 4. Model that controls the rerouting suggestions of the vehicles within the network.

SUMO comes with an interface, TraCI, which we use to gather the required information for periodically generating Uppaal Stratego models. In Figure 2, SUMO implements a Model of a traffic network and a Python script using TraCI serves as the Interface and the Controller processing both the traffic data and the strategy. The full source code of this project can be found in https://github.com/Marglib/AAUP7.

Network Preliminaries

Figure 5 shows the artificial SUMO traffic network we consider. The figure also includes additional information explained below. The network consists of two perpendicular large roads and several smaller side roads. We will refer to the large roads as the main roads. The small roads each have a single lane in each direction. The main roads have two outgoing lanes and two incoming lanes that fork into three lanes, with dedicated left turn lanes, when nearing an intersection with a small road. When the two main roads meet they fork into four lanes, with dedicated left and right turn lanes. As a constraint,

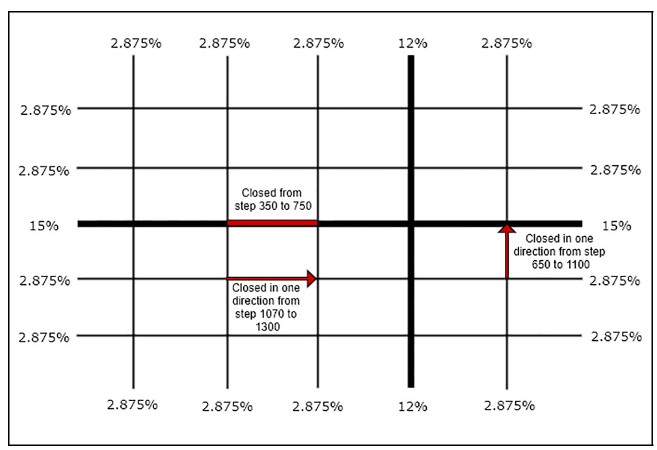

Figure 5. SUMO model for an urban traffic network with 25 intersections. The entry point probabilities for traffic generation are displayed for every road. For additional experiments, the network supports closing roads. These roads are displayed with their corresponding closing times.

U-turns are illegal in any intersection with a main road. Additionally, each road has a speed limit of 16 m/s, as

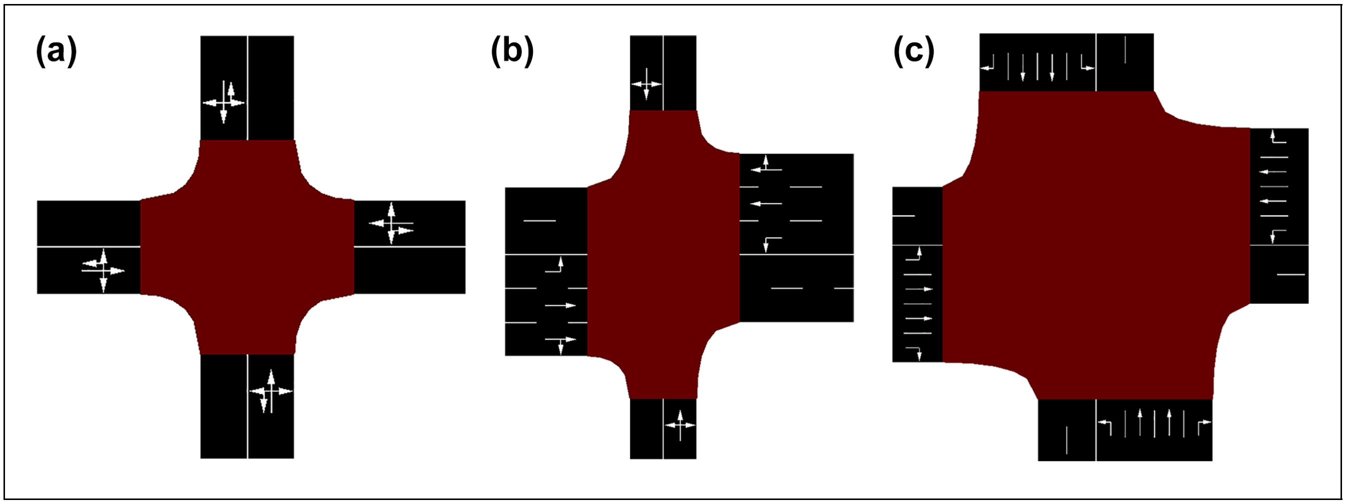

Figure 6. Different types of intersections: (*a*) intersection between two small roads, (*b*) intersection between small and main road, and (*c*) intersection between two main roads.

we want to represent an urban traffic network. The different types of intersections can be seen in Figure 6. For simplicity there is only one type of vehicle, which is a standard passenger car. Additionally, we assume perfect knowledge of the network to acquire the data mentioned above.

Case Specific Functions

As described above, we have two case specific functions, the threshold and the weight function. We conducted a series of smaller experiments to test the capacity of the various road types. Using linear regression we found the following weight functions:

Number of lanes	Weight function
1	$1.54 \cdot numberOfCars + 13.73$
2	$0.17 \cdot numberOfCars + 7.37$
3	$0.44 \cdot numberOfCars + 6.46$
4	$0.84 \cdot numberOfCars + 5.69$

The threshold is calculated as $threshold = 12 \cdot \frac{edgeLength}{100}$, where 12 is a heuristic value and we multiply with $\frac{edgeLength}{100}$ to account for longer edges being able to hold more vehicles before congesting. We divide by 100 to normalize the threshold according to the weight function.

Scenarios for Experiments

We experiment with four different configurations using different traffic light controllers, with or without our rerouting model. The four configurations are as follows: no rerouting model and Default SUMO traffic light (*14*) (ND); Rerouting model and Default SUMO traffic light (RD); no rerouting model and smart traffic light (NS); and rerouting model and smart traffic light (RS). When no rerouting is applied, the vehicles receive a route using a shortest path algorithm on entering the traffic network (*15*). To conduct the experiments we generate traffic loads through repeated Bernoulli Trials over a Poisson distribution for every incoming road:

$$P(k \text{ vehicles per hour}) = \frac{\lambda^k \cdot e^{-\lambda}}{k!}$$

where λ is given by the entry point probability for the given road, seen in Figure 5, multiplied by differing vehicles per hour (vph): 5,400 vph, 6,300 vph, 7,200 vph, 8,100 vph, and 9,000 vph.

We refer to a scenario as a configuration with a given load. We sample 50 simulations for each scenario to gain confidence in the results. This means that we conduct 50 times the number of loads times the number of configurations, giving a total of 1,000 experiments. In case the generated traffic results in an incoming road being congested, SUMO will maintain a queue of vehicles waiting to enter the network on the given road. Such vehicles will not count toward the queue length. If a vehicle is trying to cross an intersection, where the outgoing road is congested, SUMO makes the vehicle wait until there is space on the outgoing road.

Additional Considerations. In addition to the network, scenarios, and probability distributions, we need to fix a horizon for the Uppaal Stratego models (see above). We set the horizon to 40 s. We fix the probability of a vehicle accepting a route suggestion to 90% ($p = 0.9$ in

Table 1. Results for the Experiments without Closed Roads. The Average Measurements are the Mean Over the 50 Simulations, and the Max Values are the Maximum Across the 50 Simulations

	ATT	AD	AWT	AQL	MTT	MD	MWT	MQL	95%	CO_2
5,400 vph										
ND	196.2	95.2	57.9	14.1	499.6	356.2	252.7	282.2	42.0	532.9
RD	190.3	89.5	54.3	12.9	450.5	307.7	221.5	251.2	36.0	520.4
NS	156.0	55.1	25.6	9.5	380.6	243.1	189.3	254.2	27.9	439.7
RS	**155.5**	**54.6**	**25.3**	**9.4**	**371.5**	**237.4**	**184.4**	**233.7**	**26.8**	**438.5**
6,300 vph										
ND	211.2	109.9	67.0	17.0	579.3	438.7	304.6	278.2	53.3	562.5
RD	197.3	96.4	58.4	14.3	512.6	369.9	267.3	**267.4**	42.6	534.4
NS	167.6	66.6	33.6	12.0	450.3	317.5	255.8	288.7	35.9	466.0
RS	**166.1**	**65.1**	**32.6**	**11.5**	**445.2**	**309.6**	**251.9**	278.9	**33.9**	**462.7**
7,200 vph										
ND	240.7	138.6	85.8	21.8	766.7	631.8	443.9	295.0	76.7	621.3
RD	209.0	107.7	65.4	16.3	626.8	485.8	344.5	**286.6**	**47.7**	558.1
NS	194.0	92.4	53.1	17.5	625.0	494.1	416.5	290.9	61.6	526.8
RS	**185.0**	**83.4**	**46.1**	**15.2**	**542.4**	**415.0**	**339.4**	292.4	49.3	**505.9**
8,100 vph										
ND	306.7	203.2	131.2	31.1	1230.8	1098.2	801.1	315.2	115.3	752.9
RD	233.3	131.8	80.8	**19.4**	911.7	774.0	548.1	311.4	**62.1**	605.4
NS	261.0	158.1	106.3	29.4	1068.7	933.6	817.9	309.4	113.0	681.4
RS	**221.6**	**119.5**	**74.6**	21.8	**790.4**	**661.0**	**567.8**	**300.6**	78.0	**590.9**
9,000 vph										
ND	395.5	290.5	202.1	41.1	1843.4	1704.8	1368.9	322.3	158.8	944.5
RD	**272.7**	**171.0**	**107.3**	**24.0**	1193.8	1060.6	**775.0**	319.6	**80.9**	**682.8**
NS	456.1	350.1	281.0	49.1	2293.2	2158.0	1982.4	356.1	171.4	1147.7
RS	288.7	186.0	129.8	31.7	**1187.5**	**1056.3**	937.5	**315.5**	106.3	747.2

Note: Bold indicates the minimum value for the given measurement and load. vph = vehicles per hour; ATT = Average Travel Time; AD = average delay; AWT = Average Waiting Time; MTT = Max Travel Time; MD = Max Delay; MWT = Max Waiting Time; AQL = Average Queue Length; MQL = Max Queue Length; 95% = the 95th percentile queue length; CO_2 = the average CO_2 emission.

Figure 4). These probabilities could be obtained from historical data in a real scenario.

Scheduled Road Closing

Using the network, as described above, we assume there are some scheduled road closings, for example, scheduled road work or events. We fix a set of closed roads in the network. These are the red lines seen in Figure 5.

Because of time constraints we fix a set of closed roads, and experiment only with this one set. For this set, we instead conduct only three simulations for each scenario, totaling in 48 experiments with closed roads. This is again because of time constraints, as we deem the aforementioned experiments more important for testing the system in a real scenario. For the experiments we parse the information that the road is going to close 20 s before the road closes, such that the Uppaal Stratego model has time to react.

Experimental Results

In this section we present the results of our experiments for each of the four configurations. The results with no closed roads can be seen in Table 1 while the results with closed roads can be seen in Table 2.

Measurements. We measure the performance of our model on 10 different parameters. All of the measurements are retrieved from the SUMO (*13*) statistics files. The measurements we consider are: average travel time per vehicle in seconds (ATT); average delay per vehicle in seconds (AD); average waiting time per vehicle in seconds (AWT); max travel time in seconds (MTT); max delay in seconds (MD); max waiting time in seconds (MWT); average queue length per road in meters (AQL); max queue length in meters (MQL); 95th percentile queue length in meters (95%); and average CO_2 emission per vehicle in grams (CO_2). Waiting time is the time the vehicle spends driving slower than 0.1 m/s and delay is the time in which the vehicle was delayed, for example, time it would not have lost if it had been driving at the maximum speed limit at every point in time.

Results Without Closed Roads

The first thing to notice, in Table 1, is that in most cases, ND is the worst. Furthermore, it should be noted that

Table 2. Results for the Experiments with Closed Roads. The Average Measurements are the Mean Over the three Simulations, and the Max Values are the Maximum Across the three Simulations

	ATT	AD	AWT	AQL	MTT	MD	MWT	MQL	95%	CO_2
5,400 vph										
ND	201.9	100.3	61.0	14.8	496.3	386.2	265.7	295.9	43.5	545.8
RD	193.6	92.1	55.8	13.3	508.0	380.0	246.5	**225.9**	36.0	528.3
NS	**158.9**	**57.5**	**27.1**	**10.0**	**373.7**	**247.2**	**182.0**	295.8	**28.5**	**447.6**
RS	161.3	59.9	29.4	10.6	449.3	301.7	251.0	295.9	33.5	454.3
6,300 vph										
ND	221.0	118.6	72.5	18.4	647.7	502.3	361.7	**218.0**	61.0	585.1
RD	203.8	101.8	61.8	15.1	503.0	362.7	**266.0**	328.1	43.5	550.3
NS	179.1	76.8	40.8	14.1	609.0	475.1	393.7	295.9	46.0	494.7
RS	**173.2**	**71.2**	**36.9**	**12.6**	**483.7**	**336.0**	283.7	254.3	**38.7**	**480.7**
7,200 vph										
ND	283.8	179.6	115.9	28.2	1263.7	1117.9	865.7	**295.9**	98.5	713.3
RD	219.8	117.7	71.9	**18.0**	655.3	519.7	**390.0**	**295.9**	**56.0**	582.0
NS	230.4	127.4	79.9	23.7	878.3	752.5	635.7	**295.9**	88.8	610.1
RS	**204.4**	**102.0**	**60.2**	18.4	**644.3**	**518.0**	431.0	328.1	68.2	**553.0**
9,000 vph										
ND	1065.9	941.9	813.7	65.7	4848.3	4703.3	4248.7	368.5	211.0	2533.4
RD	**315.1**	**212.4**	**137.6**	**28.9**	**1338.7**	**1220.6**	**896.7**	**295.9**	**88.5**	**772.9**
NS	891.2	777.2	672.3	74.4	3979.7	3825.8	3522.7	392.7	193.8	2182.2
RS	346.2	242.3	177.2	38.1	1614.0	1482.3	1337.7	**295.9**	136.0	882.0

Note: Bold indicates the minimum value for the given measurement and load. vph = vehicles per hour; ATT = Average Travel Time; AD = Average Delay; AWT = Average Waiting Time; MTT = Max Travel Time; MD = Max Delay; MWT = Max Waiting Time; AQL = Average Queue Length; MQL = Max Queue Length; 95% = the 95^{th} percentile queue length; CO_2 = the average CO_2 emission.

each configuration performs worse on the 9,000 vph load than on any of the other loads. This is likely a result of some lanes in the network being congested.

Additionally, it can also be deduced that for our rerouting model to have any effect we need a relatively large load. This can be seen when comparing NS and RS, as seen in Figure 7*a*, where the only notable difference is on the three largest loads. Here, there is a relatively large difference between NS and RS showing that it is worthwhile rerouting the vehicles in some cases. The same can be seen when comparing ND and RD. As such, our rerouting model can improve travel time with either traffic light controller on the three largest loads

Another thing we can derive from Table 1 is that using our rerouting model is less punishing for the individual vehicle in most cases as can be seen in the max measurements (MTT, MD, MWT, and MQL) when comparing ND with RD and NS with RS.

Results with Closed Roads

The results in Table 2 show many of the same tendencies as seen in the results without closed roads. However, every configuration performs worse than their counterparts in Table 1, which can be seen when looking at the ATT for closed roads in Figure 7*b*. It should be noted that, when comparing *NS* and RS, it is still better to use our rerouting model, with the exception of the 5,400 vph load. We see that RS and RD, in general, are less negatively affected than NS and ND by the closed roads. This can be seen, comparing NS and RS on ATT for 7,200 vph, where NS is increased by 36.4 s while RS is only increased by 19.4 s.

We also see that NS performs better than RD in every load but the two largest loads. This indicates that, at some point, given closed roads, the reroute recommendations are more effective than only having a smart traffic light controller. Though these results are promising the experiments are not as exhaustive as the experiments without closed roads, and thus not completely conclusive.

Whether roads are closed or not, the results suggest that rerouting can improve the traffic flow in general, especially with large traffic loads.

Discussion

Throughout the conduct of the experiments we identified some sources of error. The largest source of error being that the vehicles are removed by SUMO from the network after a certain amount of time standing completely still (300 s). This is a necessity to ensure that the simulation progresses and the network cannot be gridlocked forever. We measured how many vehicles were removed and in most cases there was no need to remove vehicles. However, some experiment scenarios congested to the

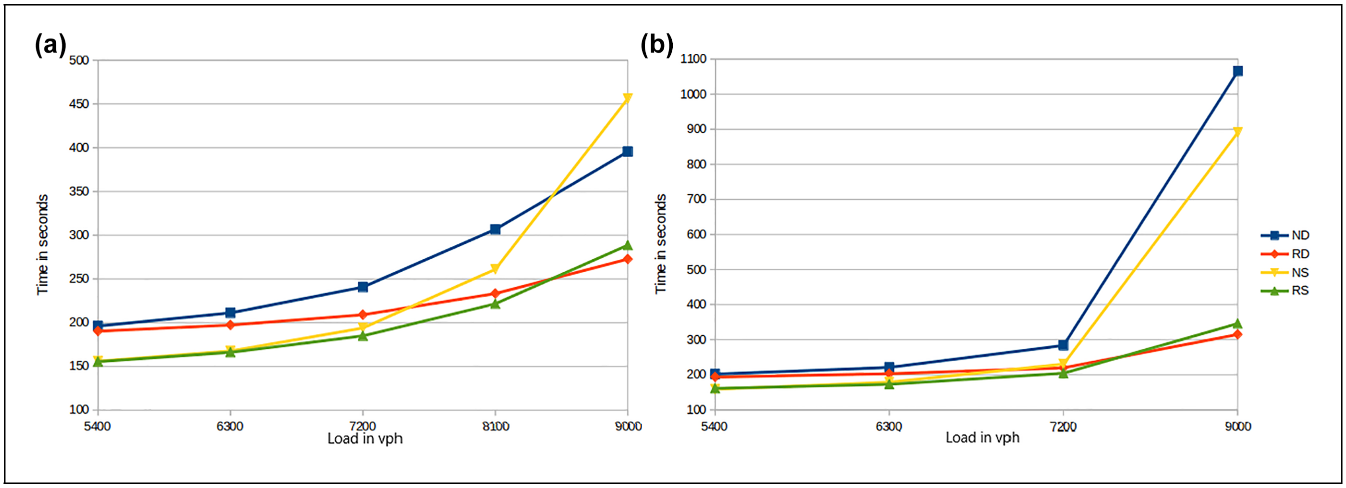

Figure 7. Average travel time per configuration with and without closed roads: (*a*) without closed roads, and (*b*) with closed roads.
Note: vph = vehicles per hour; ND = No rerouting model and Default SUMO traffic light (14); RD = Rerouting model and Default SUMO traffic light; NS = no rerouting model and smart traffic light; RS = rerouting model and smart traffic light.

point where removals were needed. The worst case was a load with 9,000 vph. This load was used with *NS* and resulted in 121 removals throughout the simulation. The worst case of removals for other configurations was: 17 for ND, 6 for RD, and 13 for RS. We assume that, if not for the removed vehicles, the net would have been gridlocked. We see in the results that the smart traffic lights yielded worse results than expected. We believe this is because of some unexpected behavior, which probably also caused the high amount of removals. When the smart traffic lights are run in conjunction with our rerouter (RS) it seems that the bug is negated and the results are better when using this configuration compared with ND, NS, and RD. However, we have not identified why this is the case.

We performed the experiments in parallel on the AAU MCC cluster (*16*). We allocated four cores, AMD Opteron 6376 Processor, 2.3 GHz, and up to 1 TB memory for each experiment. With these specifications the longest experiment took around ~ 25 days of computation time. This was an instance of *RS* with the largest load.

Conclusion

It is possible to optimize traffic flow on an urban traffic network using our Uppaal Stratego model to synthesize reroute recommendation strategies. We show by simulation in SUMO that, rerouting vehicles is a valid method for reducing travel times, queue lengths, and CO_2 emissions. The results show that on small loads the smart traffic lights are the most important, but that, as the loads get larger, rerouting becomes increasingly effective to the point where it outperforms the smart traffic lights.

We observe improvements of up to a 31% decrease in average travel time for the representative traffic network without closed roads and up to 70% with closed roads, while also observing promising results in relation to fairness. As such, we conclude that using rerouting recommendation strategies, made with Uppaal Stratego, is a valid approach to distributing traffic to prevent and dissolve congestions, with and without closed roads. Additionally, our study suggests that incorporating the smart traffic light controller further improves this approach.

Future Work

This paper considers applying a Uppaal Stratego model to a traffic network, which showed promising results. However, further testing of the model is necessary. In this paper the model was tested on a single traffic network. To explore whether these results hold for traffic in general, experiments on more heterogeneous scenarios should also be considered. This includes testing different speed limits, road sizes, vehicle route distributions, and intersection controllers as well as testing our rerouting model on real road networks with actual data. Some of the experiments took several days. To combat this, partitioning could be a possibility as done in Larsen et al. (*11*). However, we expect this would yield slightly worse results.

Modification of the model itself should also be investigated. Using a different weight function, threshold, or allowing Uppaal Stratego to choose from different pre-computed routes may yield improved results. However, pre-computing routes will only be possible for the cases where no roads are closed or when the specific road closing cases are known. Although, it might be computationally heavy, allowing Uppaal Stratego to have a choice for all vehicles might also improve results. Testing with different road closing sets is required to further discover the solution's ability to handle these as we only experimented with a fixed set of closed roads. In much the same way, testing different probabilities (possibly based on real data) would be relevant. This also means that many experiments are needed (theoretically infinite) to properly represent the effects of the probabilities. Additionally, data gathering methods and the percentage of drivers using the system should also be considered.

Author Contributions

The authors confirm contribution to the paper as follows: study conception and design: A. Bilgram, E. Ernstsen, P. Greve, H. Lahrmann, K. G. Larsen, M. Muñiz, P. Taankvist and T. Pedersen; data collection: A. Bilgram, E. Ernstsen, P. Greve, M. Muñiz, P. Taankvist and T. Pedersen; analysis and interpretation of results: A. Bilgram, E. Ernstsen, P. Greve, M. Muñiz, P. Taankvist and T. Pedersen; draft manuscript preparation: A. Bilgram, E. Ernstsen, P. Greve, M. Muñiz, P. Taankvist and T. Pedersen. All authors reviewed the results and approved the final version of the manuscript.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the Danish center for Data-Intensive Cyber-Physical Systems (DiCyPS) http://www.dicyps.dk.

References

1. Jespersen, P. H., and Trængselskommissionen. *Mobilitet og fremkommelighed i hovedstaden Hovedrapport Betænkning 1539*. Hovedrapport. Mobility and Accessibility in the Capital Area: Committee on Congestion, Denmark, 2013.
2. Krylatov, A., V. Zakharov, and T. Tuovinen. *Optimization Models and Methods for Equilibrium Traffic Assignment*, Vol. 15. Springer, Switzerland, 2020.
3. Cao, Z., S. Jiang, J. Zhang, and H. Guo. A Unified Framework for Vehicle Rerouting and Traffic Light Control to Reduce Traffic Congestion. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 18, No. 7, 2017, pp. 1958–1973. https://doi.org/10.1109/TITS.2016.2613997.
4. Pan, J., M. A. Khan, I. S. Popa, K. Zeitouni, and C. Borcea. Proactive Vehicle Re-routing Strategies for Congestion Avoidance. *Proc., 2012 IEEE 8th International Conference on Distributed Computing in Sensor Systems*, Hangzhou, 2012, pp. 265–272.
5. Kim, S., M. E. Lewis, and C. C. White. Optimal Vehicle Routing with Real-Time Traffic Information. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 6, No. 2, 2005, pp. 178–188. https://doi.org/10.1109/TITS.2005.848362.
6. Work, D. B., O. Tossavainen, S. Blandin, A. M. Bayen, T. Iwuchukwu, and K. Tracton. An Ensemble Kalman Filtering Approach to Highway Traffic Estimation Using GPS Enabled Mobile Devices. *Proc., 47th IEEE Conference on Decision and Control*, Cancun, Mexico, 2008, pp. 5062–5068.
7. David, A., P. G. Jensen, K. G. Larsen, M. Mikučionis, and J. H. Taankvist. *Uppaal Stratego*. Springer, Berlin, Heidelberg, 2015, pp. 206–211.
8. Eriksen, A., C. Huang, J. Kildebogaard, H. Lahrmann, K. Larsen, M. Muniz, and J. Taankvist. Uppaal Stratego for Intelligent Traffic Lights. *Proc., 12th ITS European Congress*, Strasbourg, France, 2017.
9. Madrigal, A. C. The Perfect Selfishness of Mapping Apps. 2018. https://www.theatlantic.com/technology/archive/2018/03/mapping-apps-and-the-price-of-anarchy/555551/.
10. David, A., P. G. Jensen, K. G. Larsen, M. Mikučionis, and J. H. Taankvist. Uppaal Stratego. In *Tools and Algorithms for the Construction and Analysis of Systems* (C. Baier and C. Tinelli, eds.), Springer, Berlin, Heidelberg, 2015, pp. 206–211.
11. Larsen, K. G., M. Mikučionis, M. Muñiz, J. Srba, and J. H. Taankvist. Online and Compositional Learning of Controllers with Application to Floor Heating. In *Tools and Algorithms for the Construction and Analysis of Systems* (M. Chechik and J.-F. Raskin, eds.), Springer, Berlin, Heidelberg, 2016, pp. 244–259.
12. Larsen, K. G., M. Mikučionis, and J. H. Taankvist. Safe and Optimal Adaptive Cruise Control. In *Correct System Design* (R. Meyer, A. Platzer, and H. Wehrheim, eds.), Springer, Berlin, Heidelberg, 2015, pp. 244–259.
13. Lopez, P. A., M. Behrisch, L. Bieker-Walz, J. Erdmann, Y. P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner. Microscopic Traffic Simulation Using SUMO. *Proc., 21st IEEE International Conference on Intelligent Transportation Systems*, Maui, HI, IEEE, New York, 2018.
14. SUMO. Simulation/Traffic Lights. 2019. https://sumo.dlr.de/docs/Simulation/TrafficLights. html#default 4-arm intersection layout opposites.
15. SUMO. Simulation/Routing. 2021. https://sumo.dlr.de/docs/Simulation/Routing.html.
16. MCCAAU. Hardware Organisation. 2019. https://sites.google.com/site/mccaau/home/hardware.