

UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Technology

Aleksandrs Kozjutinskis

Multi-Vehicle Path Planning using Shared Data

Master's Thesis (30 ECTS)
Robotics and Computer Engineering

Supervisor(s): Kallol Roy, PhD

Tartu 2024

Resümee/Abstract

Multi-vehicle path planning using shared data

Vehicle navigation is a problem without a real solution. Every action taken can be disturbed by an accident, making the suggested path not optimal. As a result, the only moment it is possible to state that the selected road was optimal is after the vehicle has completed it and all possible disturbing events have occurred. Due to the high impact of the navigation system on traffic flow, this research suggests sharing information between vehicles to make decisions as a team, giving way for those who benefit most. It also allows them to renavigate vehicles if they face traffic jams or congestion on the road. The newly developed algorithm proved to be useful, overperforming other common path-finding strategies by 10-15% and providing results 5% close to the optimal path if possible disturbing events have occurred. Traffic flows after the developed algorithm also provided a lower traffic jam rate compared to other algorithms. Algorithms were tested using the simulation developed for this research.

CERCS: T120 Systems engineering, computer technology; T125 Automation, robotics, control engineering; T280 Road transport technology

Keywords:

Path planning; Traffic flow; Simultaneous multi-vehicle navigation; TomTom; Shared data; Time-based network

Mitme sõiduki tee planeerimine jagatud andmete abil

Lühikokkuvõte: Sõidukite navigeerimine on probleem, millel puudub tegelik lahendus. Iga ettevõtetud tegevust võib õnnetus häirida, mistõttu soovitatud tee ei ole optimaalne. Sellest tulenevalt saab ainuke hetk väita, et valitud tee oli optimaalne, kui sõiduk on selle läbinud ja kõik võimalikud häirivad sündmused aset leidnud. Kuna navigatsioonisüsteem avaldab liiklusvoogudele suurt mõju, soovib see uurimus jagada teavet sõidukite vahel, et teha otsuseid meeskonnana, andes teed neile, kes saavad sellest kõige rohkem kasu. Samuti võimaldab see neil sõidukeid ümber navigeerida, kui teedel tekivad liiklusummikud või ummikud. Äsja väljatöötatud algoritm osutus kasulikuks, toimides teiste levinumate teeotsingu strateegiatega puhul 10–15% võrra ja andes 5% optimaalsele teele lähedasi tulemusi, kui võimalikud häirivad sündmused on aset leidnud. Liiklusvood pärast väljatöötatud algoritmi andsid ka väiksema liiklusummikumäära võrreldes teiste algoritmidega. Algoritme testiti selle uuringu jaoks välja töötatud simulatsiooni abil.

CERCS: T120 Süsteemitehnoloogia, arvutitehnoloogia; T125 Automatiseerimine, robotika, juhtimistehnika; T280 Maanteetransporditehnoloogia

Võtmesõnad:

Teede planeerimine; Liiklusvool; Samaaegne mitme sõiduki navigeerimine; TomTom; jagatud andmed; Ajapõhine võrk

Contents

1	Introduction	4
1.1	Problem statement	4
1.2	Objectives	5
2	Navigation algorithms nowadays	6
2.1	Path finding problem formulation	6
2.2	Optimal solution	6
2.3	Factors	6
2.3.1	Traffic jams	6
2.3.2	Risk of congestion situation	7
2.3.3	Risk averse vehicle navigation	7
2.4	Modern algorithms	8
2.5	Shared data	9
3	Methods	10
3.1	Developed solution	10
3.2	Limitations	10
3.3	Input data	10
3.4	Road costs	12
3.5	Path choosing algorithm	14
4	Results	16
4.1	Single generation performance	16
4.1.1	Path selection efficiency	16
4.1.2	Global situation on roads	17
4.2	Performance over continuous iterations	18
4.2.1	Path selection efficiency	18
4.2.2	Global situation on roads	19
4.3	Renavigation impact	21
5	Discussion	22
5.1	Single generation performance	22
5.2	Performance over continuous iterations	22
5.3	Threshold function	22
5.4	Simulator	23
6	Conclusion	24
	References	28
	I. Licence	29

1 Introduction

Vehicle navigation using navigation tools like GPS is very common nowadays. Even if a person is an experienced driver, it is customary to check a suggested path from GPS to analyze the road situation. Navigating an unfamiliar city is highly complicated, and choosing a path depends entirely on the navigation device. New technologies even offer autonomous vehicles, where the driver is needed only for road emergencies. This leads to a situation where the path-choosing strategy no longer depends on the driver and is mainly defined by the navigation device's algorithm. As a benefit of such an action, traffic flow control moves from human suggestions to automated algorithms, making it more predictable and easily controlled.

The situation on the road is also well observed nowadays. It is possible to check the road status in real-time to find out how crowded it is compared to the other roads or itself at different times, clarify if some unfortunate event, like a car crash, has occurred, and even find out that the road is closed due to a local event or repair works. Navigation algorithms use this data to plan the paths and even have strategies to predict the car flow based on the current situation on the roads and historical data they have. [ZML23]

However, navigation algorithms tend to use selfish strategies to achieve path destinations. It leads to a situation where the same path nodes are chosen too frequently, leading to traffic jams on this road segment, increased risks of car crashes, and potentially making the best path worse than the alternatives. This work suggests avoiding such a situation, keeping track of paths cars choose, and planning the number of cars on roads in the future. [KB10]

This work also suggests reacting to car crashes, which significantly reduces the traffic flow. It is helpful to avoid using roads where car crashes have happened. The renavigation algorithm, which changes the car path if a significant time-slowng event occurs at the suggested road, may solve this problem.

Continuous renavigation may annoy drivers. Thus, it is also essential to introduce a separation between autonomous vehicles, which can handle any amount of navigation, and human-driven vehicles, renavigated with lesser priority.

1.1 Problem statement

Considering the overwhelming usage of navigation tools and the occurrence of autonomous devices in traffic, it may be wise to start sharing information about the selected destination points and paths towards them. Shared information will allow to predict traffic jams and possible congestion in advance. Conversely, sharing the destination and the selected path is sharing personal information. Does advanced path planning provide sufficient navigation benefits to make this trade valuable?

1.2 Objectives

Evaluation of possible benefits of advance planning is performed according to the following criteria:

- How can sharing data about selected paths for drivers improve the fastest path suggestions and influence traffic flow;
- Does renavigation of traffic due to congestion or potential traffic jams help determine the fastest path and how it changes traffic flow;
- Will continuous usage of such an algorithm improve traffic flow?

To analyse such algorithm behaviour, a time-based network is required, which can handle the states of the road and store their future statements. As an input for this model, the historical data of the road state can be used. This will allow the creation of a traffic flow model close to real-life traffic.

The benefits of advanced planning must be compared to the optimal or suboptimal algorithms for path choosing. Advance planning must outperform them to make it worthwhile to share personal data.

2 Navigation algorithms nowadays

2.1 Path finding problem formulation

Let us consider that the road plan is described as graph G , where vertices V are crossroads, edges E are roads between crossroads, and weight W is the time taken to move along the edge E . Then the shortest path between vertices V_1 and V_2 will be the combination of connected edges $P=E_1, E_2, \dots, E_n$, with the smallest sum of related weights W_e in P .

2.2 Optimal solution

An optimal solution for the selected problem is the Dykstra algorithm and A^* algorithm. [Vas08] The Dykstra algorithm, known for its efficiency, calculates the cost from the departure vertex A to all other vertices in the graph. Some modifications also store the last edge used to reach the corresponding vertex. This allows for easy backtracking, moving through the last edge array until vertex A is reached.

A^* algorithm does not visit all the edges to provide the result. It introduces an extra coefficient to order the edge it will pick next. It can be set to weigh W_1 connected to edge E_1 . Then, the next edge, which is picking, will always have the shortest path towards it. After vertex B is reached, it is possible to track the path the same way it is done in the Dykstra algorithm. [FFK13] [Vas08]

The main benefit of the A^* algorithm is a configurable coefficient and the possibility to visit only part of the vertices. It allows us to set it to an abstract value like possible distance till point B and navigate through the graph way quicker, reducing the visited nodes count dramatically.[FFK13]

2.3 Factors

Modern navigation systems provide various strategies for picking paths. [PD14] They can calculate the fastest path and the path with the least fuel consumption, emissions, risk of car crashes, prioritize using highways, or any other requested factor. To describe further strategies, let us look at some of these factors considered important in this research.

2.3.1 Traffic jams

Crowded roads lead to car jams, significantly reducing the time it takes to cross a road segment. In recent research, Zhi Cai considered that the time it takes to cross a busy road could be described with the JW (jam weight) equation 1 and figure 1. Where d specifies the distance between the road and the vehicle, and θ represents a threshold within which the traffic status is important for route planning. When d exceeds the threshold θ , it rapidly declines until it is very close to 0. [ZCD23]

$$JW(d) = \frac{1}{1 + e^{d-\theta}} \quad (1)$$

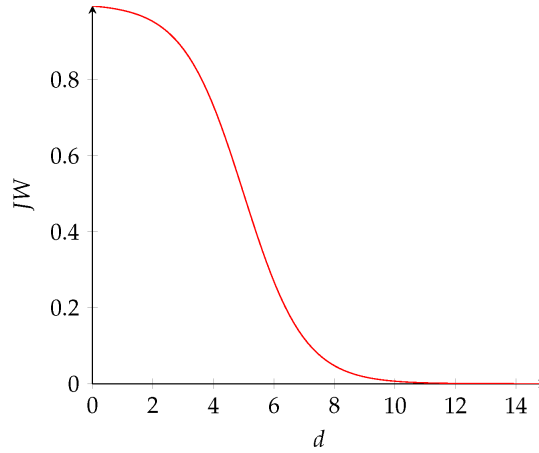


Figure 1. JW function graph

In Zhi Cai research, there are only two road states, crowded or not. In this research, after some adjustments, this function will describe the time taken to move through the road and predict the risk of a car crash (section '3. Methods'). [ZCD23]

2.3.2 Risk of congestion situation

The other way to avoid crowding on the road is to keep track of the risk of facing congestion situations. Jing Guo shows that minimal and maximal travel times can differ significantly for multiple roads with the same average travel time deviation. As a main factor for such distribution is shown, the travel time with congestion and without differs significantly, as well as the possibility of such events. As a result, a 'faster' algorithm was suggested, which shows the possibility of the selected path being the 'fastest'. [JG17] [ZML23]

2.3.3 Risk averse vehicle navigation

It is essential to point out that crossroads are hazardous places where road accidents may occur. Many studies are explicitly focused on solving this problem, either to provide an algorithm for autonomous vehicles or assistance for human-driven vehicles at crossing crossroads or roundabouts. [AJK22] In any case, in human-driven vehicles for the person, it is essential to be careful and pay attention at crossroads. Thus, for human-driven vehicles, it is suggested that a 'risk awareness' factor be introduced, which describes the extra time needed to make safe decisions at these dangerous road points. [Bel09]

2.4 Modern algorithms

It is impossible to provide an optimal algorithm for road navigation. Some unlikely events may occur on the road, and some people may use inefficient routing strategies. As a result, a new path may occur, which is better than expected, and the only result that can be provided is a suboptimal algorithm, which may provide a good choice for a path. Jing Guo suggests the following categories for classifying algorithms for car navigation.

Least expected travel time. (LET) One-factor algorithms. Commonly, the A* or Dykstra algorithm, where the searching criteria is to travel time here. Some variations may provide multiple options. Is not risk-aware and expects the best possible result, which may be deceptive. [ASSV93] [FFK13]

Mean-risk mode. Another algorithm is trying to minimize specific conditions. As extra criteria appear to be a deviation from travel time, the mean from extra is required if some kind of risk will occur. [PD14] [MNM22]

Travel time-based network performance model. Tries create a simulation of a roadmap and create traffic flow in it. Then, find the optimal path after all events occur. It can provide results remarkably close to optimal. However, it requires much computation. The problem is handling such a simulation and recalculating all the traffic flows. Also, supersampling may be needed to provide a more realistic final model. Another problem is to predict vehicle movement in this model. Ant colony [SHOM11] or bee colony [DMS16] optimization models are used as an option to implement traffic assignment.

Stochastic routing over time-based networks. The algorithm simulates the stochastic movement in the time-based network according to the selected strategy. The performed path will be the suggested path of the algorithm.[LTL10]

Faster criterion. Along with the path, calculate the possibility of this path occurring. Each risk possibility on the path will split the path into 2 with different probabilities. The final suggestion is to select based on both the possibility of the path and the time it takes to travel it. [JG17]

Only two states are tracked: risk occurred and risk not occurred. The possibility of path P to be completed in time T is a function determined by possible risks. This distribution is replaced by two points uniting the 'no risk' and 'risk' conditions.

2.5 Shared data

Usually, shared data are not used for path navigation. They are used only for specific problem solutions, like delivery, taxi, or car evacuation. Ethically, it is inappropriate to share a person's position on the road. On the other hand, it will be known that the vehicle passed the selected road in a while based on statistics. Thus, providing sufficient anonymization may be enough to avoid this ethical dilemma.

3 Methods

3.1 Developed solution

Develop a real-time traffic flow simulator based on historical data on traffic flow on the roads. The simulator can create an instance of a road model based on historical information or path state from the previous iterations. During each simulation, requests with department and destination points on the traffic flow are created. Then, a developed path-finding algorithm can be called to perform the vehicle navigation. Alternatively, a risk-aware or fastest-path algorithm can be chosen for vehicle navigation. They represent the control group of 'known sub-optimal' algorithms. After simulation, the optimal path can be determined for each request based on each instance of the traffic flow model that occurred after all movements were performed and all congestion situations generated in the simulation.

3.2 Limitations

The simulator does not track traffic light work. It is assumed that traffic lights do not exist.

Data of the path selected for a vehicle is shared to simulated. The simulator information about the vehicle, which has a send path, can be anonymized to show only the information that something will occur at a specific point at a specific time. On the other side, if a vehicle is navigated, it can use the secret token to understand that the renavigation request is for it.

3.3 Input data

Input data is provided by TomTom (figure 2 [Tom]), which provides traffic statistics about the selected area in the world at the required time span. This data is parsed into a graph model with additional information necessary for computations as shown in table 1.

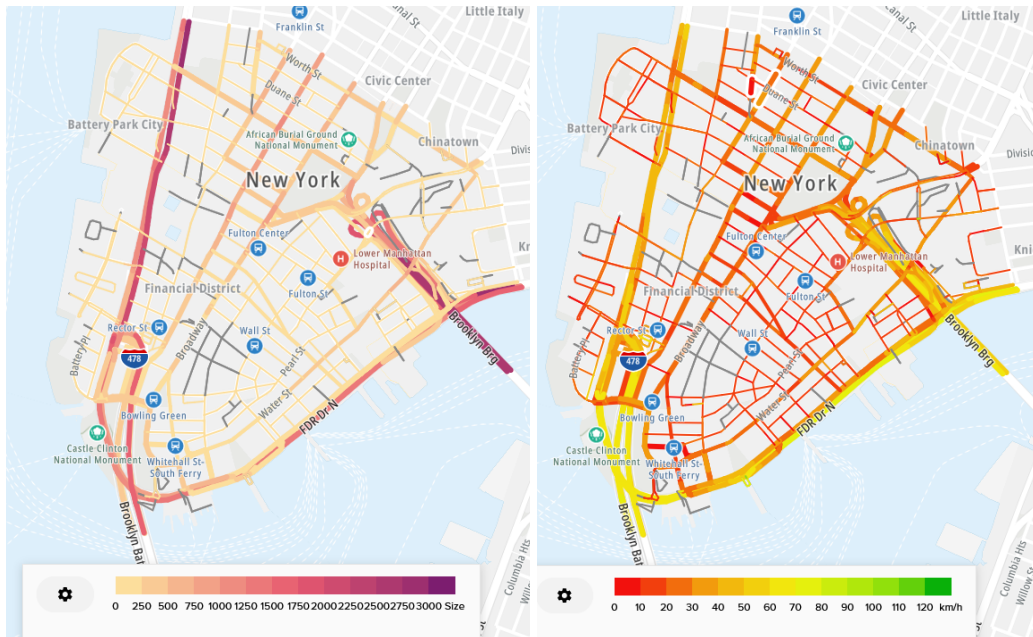


Figure 2. New York traffic visual representation in TomTom. a) amount of vehicles at left; b) median speed at right;

Data	Example
Coordinates of vertices	$x = 0.1$ $y=0.2$
Speed limit s	70 km/h
Road distance s	60 m
Data at time periods	
Time from, time to	9:00 - 9:15
Amount of vehicle at time	7
Average and median speed	50 km/h
Amount of vehicle at time	7

Table 1. Additional data provided

3.4 Road costs

Algorithm 1 describes the possible time it will take to complete the selected path at a given time. This value is determined by multiple factors, which may delay the travel time on the road. This algorithm's output value is required for path determination. It does not indicate the time value for path completion.

Algorithm 1: costCalculation

Input: time, path, driver

Result: cost

```

1  $cost \leftarrow path.CostFunctionFromThreshold(count_v, time);$ 
2 if  $path.riskOccured()$  then
3    $cost+ = riskEffect;$ 
4 if  $driver.Autonomus = false$  then
5    $cost+ = humanDelay[path];$ 
6 if  $driver.AvoidRisk$  then
7    $cost+ = path.riskPosibility * path.riskEffect;$ 
8 return  $cost;$ 

```

Risk awareness All vehicles are either human driven or autonomous. If the vehicle is human-driven, it has a delay at each crossroad varying from 0.5s to 1.5s. Autonomous drivers do not have delays. During simulation, a risk awareness table is generated once for each path request and shared among all simulated algorithms. (2)

	$Road_1$	$Road_2$...	$Road_n$
$PathRequest_1$	1,4679	0,9363	1,4244	1,314
$PathRequest_2$	0,7287	1,0013	0,6799	0,5795
...	0,5566	0,6581	0,8714	0,7568
$PathRequest_k$	1,0082	0,9432	0,967	0,8263

Table 2. Example of generated delays

Traffic jams Input data contains information about how many vehicles were on the road at a specific time period. It allows us to make a graph of the vehicle to median speed relation and to fit with the JWthreshold function (section 2.2). In Figure 1, some adjustments are observed. To make data more responsive compared to actual data, the formula has been updated (2). Now, it considers minimal (k_{outer}) and maximal (k_{inner})

speed on the road and the number of vehicles needed to decrease speed to a minimum Th_{length} . (Cnt_{cur}) describes vehicles which are currently on the roads, while (Th) represents the amount of cars needed to create traffic jams.

$$[hbt!]f(x) = k_{inner} * \frac{1}{1 + e^{\frac{Cnt_{cur}-Th}{Th_{length}}}} + k_{outer} \quad (2)$$

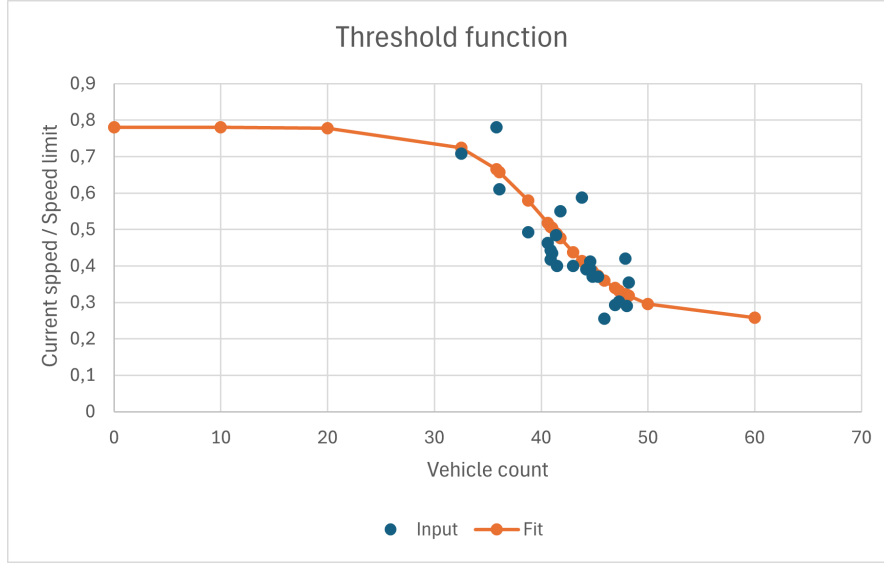


Figure 3. Threshold function for a road. Threshold = 40, threshold length 3.8, innerCoeff = 0.525, minValue = 0.255

After creating the graph representation of the road, each edge's fitting function is called to determine the threshold function. Figure (3) shows an example of this action. From input data, it is evident that vehicle speed is slowly decreasing between 30 and 50 vehicles. The best speed achieved on the road was 80% of the speed limit. This value will determine (k_{inner}). The lowest possible speed achieved is 25% of the speed limit, which is set to (k_{outer}). Fitting showed that the most rapid decrease was at 40 vehicles (Th) and set Th_{length} to match input data as well as possible.

Risk of congestion situation Two factors determine the possibility of risk on the road: randomness and relative risk. Relative risk is determined by the existence of jams on the road and the average speed excess over 50 km/h. (3)

$$[hbt!]P_{risk} = P_{base} + P_{speed}(v_{avg}, v_{limit}) + P_{threshold}(Cnt_{cur}, Threshold) \quad (3)$$

3.5 Path choosing algorithm

Each iteration in the simulator follows the order shown in figure 4. During path section step, the path-choosing algorithm 2 is executed. It suggests a list of paths for navigation between requested points, which is considered to be close to optimal. The first element of the list is always the fastest path determined by the cost evaluation function 1. After the path has been selected, renavigation of other vehicles on the road may occur if the selected path is close to the threshold limit, which will lead to a traffic jam. Afterwards, congestion situations on the road occur according to the risk of occurrence defined by the formula 3. If a risk occurs on the paths of any controlled vehicle, another renavigation may occur to select the new best pathway.

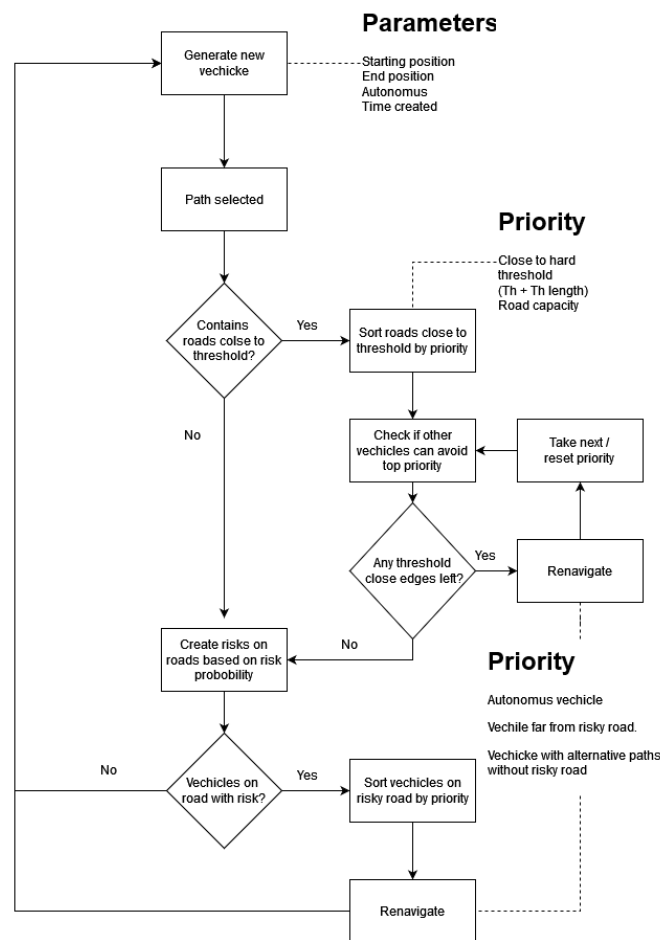


Figure 4. Amount of crowded roads after simulation and 400 path searches

Algorithm 2: pathFinding

Input: Graph model, start time, search method

Result: List of paths

```
1 affordableCost ← MaxInt;
2 while good path may exist do
3   cost[graphEdges] ← MaxInt;
4   previousEdge[graphEdges] ← ∅;
5   foreach weight ∈ startPosition do remainingWeights.add(weight);
6   ;
7   while remainingWeights.Any() do
8     firstWeight ← remainingWeights.First() ;
9     if firstWeight.Cost > affordableCost then
10      | return output;
11     if firstWeight.Cost > cost[destination] then
12      | outputPaths ← backtrackNewPath();
13      | if affordableCost = MaxInt then
14      | | affordableCost ← cost[destination] * okCoeff;
15      | break;
16     foreach weight ∈ graphWeights.BeginsAt(firstWeight) do
17      | newCost ← calculateCostToNewWeight();
18      | if weight ∈ outputPaths then
19      | | newCost.Increase();
20      | if cost[connectedEdge] < newCost then
21      | | cost[connectedEdge] ← newCost;
22      | | previousEdge[connectedEdge] ← weight;
23      | | remainingWeights.Remove(weight.FinalPoint, oldCost);
24      | | remainingWeights.Add(weight.FinalPoint, newCost);
25      | remainingWeights.Remove(firstWeight);
26 if remainingWeights ∈ ∅ then
27 | return output;
```

4 Results

The effectiveness of the developed algorithm is compared with two paradigms frequently chosen for the shortest path finding. The first algorithm to compare will be the fastest roat algorithm. It is affected only by historical data and suggests one of the quickest paths it finds. The second algorithm also suggests the most rapid path based on historical data, but now, for calculation, it also considers the probability of a congestion situation occurrence.

To ensure some distribution for suggested algorithms, they may suggest not only the first path decided to be effective but also any other option which shows similar results. A similar result, a difference of 20%, has been chosen. This limitation can reduce the average time path on the road but will provide some variation in path choosing, which will avoid situations when one road is desired all over the time.

The developed algorithm is iterated in different compositions. As extra options for algorithm control, it has the possibility to turn off the following features: avoiding risks (A), renavigation if a busy road faced (T), and renavigation if a congestion situation occurs on the road (R). Naming the corresponding graphs and tables will have a corresponding abbreviation showing that the algorithm was executed with one of these features.

A final compartment is made to the optimal path. The optimal path is the fastest path after all events on the road happened. This situation is called after the simulation with all suggested paths is finished.

4.1 Single generation performance

4.1.1 Path selection efficiency

The simulation for algorithm effectiveness comparison included parallel execution of four variations of the developed algorithm (A/ATR with all renavigation options and risk avoidance enabled, A/T with only crowded road renavigation, A/R with navigation due to the congestion situation on the road, and A without any extra options) and two algorithms selected for each compartment. All algorithms receive the same requests for path creation, including start position, end position and starting time, and the same input data about the roads. Road information is parsed from the statistics provided by TomTom, where 10% of all vehicles are removed. Algorithms add back removed vehicles and create their own vehicle distribution on the road.

After simulation, the optimal paths for each navigation request are calculated in the road network developed during simulation. Then, the best possible path among all simulations is chosen to represent the best possible solution algorithms could achieve. To make the data more consistent, all unfinished paths at the end of the simulation are ignored. Statistics for remaining data can be observed in table 3. One representation also removes the first 50 of 400 path suggestions, defined mainly by input data rather than

simulation.

	Result diff.		Good guess		Diff. from the best		Prob. of the best	
	All	50+	All	50+	All	50+	All	50+
A_ATR	1,82%	1,76%	50,98%	51,95%	8,78%	8,77%	27,17%	26,95%
A_T	3,87%	3,85%	52,38%	52,92%	13,92%	14,62%	19,33%	17,86%
A	6,13%	6,40%	50,42%	50,65%	12,62%	12,95%	20,73%	19,81%
A_R	1,79%	1,79%	52,94%	54,55%	10,28%	10,64%	24,37%	24,03%
Risk-aware	1,23%	1,22%	57,70%	56,82%	23,67%	22,97%	0,56%	0,65%
Fastest	3,49%	3,91%	36,69%	36,04%	23,73%	23,08%	1,12%	1,30%

Table 3. Algorithm compartment after one generation and 400 path searches.

All the algorithms provided results that were close to the optimal path suggested after the simulation. Ignorance of the congestion situation on the road for the developed algorithm slightly increased the gap between the suggested and the optimal path. Still, it provided results that were close to optimal.

While the possibility of providing the most optimal result for all developed algorithms is about 50%, there is room for improvement. The risk-aware algorithm, for instance, demonstrated a slightly better picking rate of 57%, indicating its potential for further enhancement. On the other hand, the fastest-path algorithm, at 36%, showed a poor possibility of picking the best way, highlighting areas for future development.

To compare road situations across all algorithms, the fastest optimal path was picked and compared to the suggested paths of other algorithms. It can be observed that even though both risk-aware and fastest-path algorithms provide the best picks in their road situation, globally, the paths they suggest are worse by 23 % compared to the best path. This underscores the need for further development, as the road situations created after the proposed algorithms are worse compared to the developed algorithm.

The developed algorithm showed results that were much closer to the most optimal path. Even without any renavigation technologies, the difference between the suggested path and the most optimal path drops to 14%. Enabling renavigation options helps decrease this value even more, to an 8

The contribution of the best possible path across simulations is split evenly between developed algorithm variations. This means that the road situation after the iteration of the developed algorithm is approximately the same and independent of the renavigation strategy. The risk-aware and fastest-path algorithms showed complete incompetence, guessing only 2 and 4 routes of the fastest across all simulations.

4.1.2 Global situation on roads

Figure 5 shows the traffic flow status after simulation. Using the threshold function to determine the cost of the road produced a better traffic flow in the simulation, leading to

fewer traffic jams on the road. It correlates with the possibility of the road suggestion of the developed algorithm being the best option across all simulations with different path-searching strategies. renavigation due to the congestion situation also proved helpful, reducing traffic jams even more. It allows us to suggest that one congestion situation leads to traffic jams near it.

The fastest-path and risk-aware algorithms have more significant traffic jams. Compared with the input data, the number of significant traffic jams increased dramatically while the overall number of traffic jams dropped. This behaviour is mainly impacted by path departure and destination point selection. Requested paths in this work are primarily determined by the most crowded roads rather than actual requests humans create. Thus, movement in the simulation is unnatural and leads to disproportional traffic congestion.

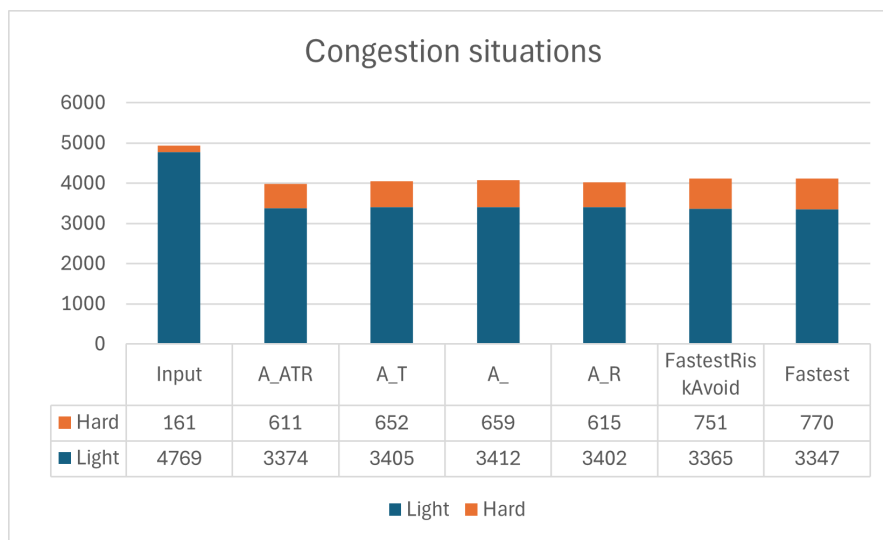


Figure 5. Amount of crowded roads after simulation and 400 path searches

4.2 Performance over continuous iterations

4.2.1 Path selection efficiency

Due to unnatural requests for departure and destination points selection, which are corrupted even more during continuous simulations, the final results in table ?? do not show the overwhelming benefit of using a developed algorithm. All the algorithms provide results that are close to the optimal solution. The fastest path algorithm reaches the point when all suggestions are optimal for its simulation. Comparing the best path across all simulations, the developed algorithm provides only slightly better results compared to risk-aware and fastest-path algorithm suggestions. It still has a slight

advantage of suggesting the best possible path across all simulations, but overwhelming the correctness of the pick faded out.

	Result diff.		Good guess		Diff. from the best		Prob. of the best	
	All	50+	All	50+	All	50+	All	50+
A_ATR	2,19%	1,76%	46,34%	43,97%	9,75%	10,66%	27,17%	26,95%
Risk-aware	1,91%	3,85%	61,59%	61,21%	14,48%	13,19%	19,33%	17,86%
Fastest	0,00%	6,40%	97,56%	96,55%	13,48%	12,06%	20,73%	19,81%

Table 4. Algorithm compartment after 10 generation and 200 path searches

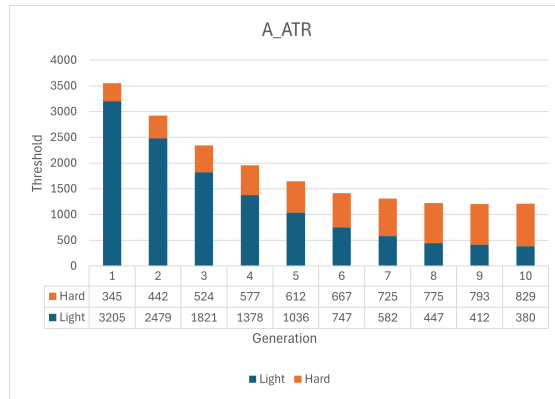
4.2.2 Global situation on roads

The situation on the roads during the simulation is dynamic. The choice of pathing algorithm significantly impacts the occurrence of traffic jams and risky situations. The developed algorithm offers a distinct advantage in this regard, primarily due to its ability to control car jams using a threshold function. The cost of the road is determined by the likelihood of risk occurrence and its congestion level. This approach results in the avoidance of busy roads, leading to a more even distribution of vehicles across all roads.

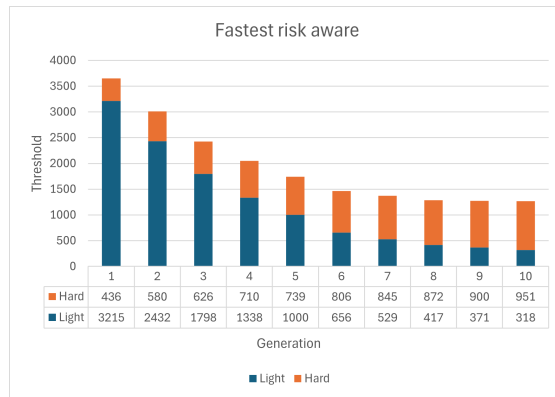
However, it's important to note that both the risk-aware and fastest-path algorithms have their limitations. They do not incorporate a mechanism to control crowdedness, which means they cannot accurately predict the occurrence of road jams.

Results for all three algorithms are shown in figure 6. Continuous usage of any tested algorithm significantly reduces the overall number of road jams. The difference between the usage of the fastest-path and risk-aware algorithms proved insignificant. The usage of the developed algorithm provides a slightly better chance of avoiding a significant car jam, which will dramatically increase path cost. The overall amount of busy roads is about the same, which suggests that the developed algorithm optimizes the distribution of crowded roads to lower average travel time.

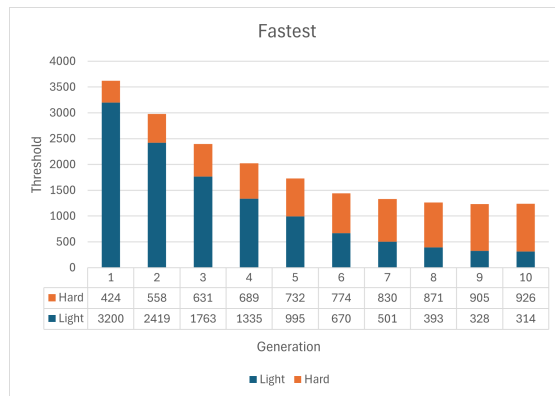
The congestion situation during the simulation shows a small increment over a generation. The randomization algorithm that creates them is mostly affected by the fact that the roads are the most crowded. As results show, the number of significantly crowded roads increases over generations. Thus, the congestion situation has increased over the generations as well. The developed algorithm provides fewer chances of risky situations occurring on the road due to what is affected by a smaller amount of bustling roads for this algorithm.



(a) developed algorithm with renavigation and congested path avoidance



(b) risk-aware algorithm



(c) fastest-path algorithm

Figure 6. Threshold limit passing

4.3 Renavigation impact

Table 5 shows an example of the amount of navigation needed during simulation. It is observable that the congestion situation on the road is mostly a reason for changing the selected path. Usually, the congestion situation leads to a significant increase in the required time to cross the road. Thus, renavigation has a good chance of providing a better solution.

Renavigation due to threshold reaching occurs incredibly rarely. The algorithm already considers all the possible traffic jams using the threshold function for cost evaluation and tries to avoid them if necessary. As a result, there is no need to perform renavigation at all.

	A_ATR		A_R A_T	
	Risk	Threshold	Risk	Threshold
Sitations forcing renavigations	1168	773	1167	812
Renavigation calls	229	703	198	655
Usefull renavigations	158	21	146	3

Table 5. Example of renavigation calls for one generation and 400 path searched

5 Discussion

5.1 Single generation performance

Pathfinding using shared data to control and renavigate vehicles proved to be helpful. Results show that all observed algorithms provide results close to the optimal solution in their simulation. The difference between the fastest path in the simulation and the path suggested by the algorithm does not exceed 10%, which makes them close to optimal independently from the chosen algorithm. However, the development of the road condition differs depending on the algorithm. After the simulation, paths in the traffic situation created by developed algorithms are 10-15% faster than those suggested by the least expected time and mean risk models. Also, amount of busy roads after the simulation is a bit smaller for developed algorithms, showing a better global situation on the road.

The renavigation algorithm did not meet expectations. The benefit of renavigation cars to avoid traffic jams must be more significant than constantly forcing vehicles to change their path. Results show that it is useful to avoid only huge time-loss events on the roads, like car crashes. Algorithms ignoring this renavigation condition suggested paths with a bigger difference to the optimal solution.

5.2 Performance over continuous iterations

Pathfinding using shared data to control and renavigate vehicles over multiple generations did not show significant improvement over risk-aware and path-finding algorithms. All the observed algorithms provide results close to the optimal solution in their simulation. The fastest-route algorithm provided the best possible solution in its simulation with a surprising accuracy of 97%. Comparisons beyond all algorithms for the best route selection and the traffic flow after the final generation show that the algorithm progressed to approximately the same state and that the gap between all algorithm's best paths has shortened. Now, routes suggested by developed algorithms are only 5% faster than the alternatives. It also has slightly better traffic flow conditions.

Overall, the amount of crowded roads decreased significantly, but more extremely crowded roads occurred. This situation occurred due to a specific path request algorithm in the simulator algorithm, which does not have any connection with the algorithm selected for path determination.

5.3 Threshold function

The threshold function is an excellent way to represent the time it will take for a vehicle to ride a road. It intuitively correctly guesses the behaviour of the car on the road. It is a limit when adding extra vehicles will not cause time loss on the road, which is what the

threshold describes. Passing this limit leads to a rapid speed decrease on the road. This process is not immediate, and a threshold length precisely describes this period.

It is complicated to determine the minimal speed on the road from the historical data. Even though they provide sufficient data for functioning fitting, it is not expected when the road is filled to its limits in real life. Historic data fitting shows only 10% of roads in a situation where the threshold limit has been broken. It may make the guess of the minimal possible time on the road not precise.

Results show that algorithm iteration using threshold function for road cost determination overperformed algorithms ignoring it. That shows that the threshold function expresses behaviour on the roads well enough to be a tool for road cost representation.

5.4 Simulator

The simulator plays a crucial role in this research, aiding in the determination of the fastest path and providing a dynamic environment for algorithm testing. The developed algorithm represents a variation of the stochastic algorithm over a time-based network performance network. Thus, it would be impossible not only to prove the efficiency but also to present the result of the algorithm without it.

Even though the developed simulator generally provides all the necessary tools for a fair algorithm comparison, the generation departure and destination points can still be improved. The current solution suggests removing 10% of vehicles from all the roads before iterations and asking for random departure and destination points until it returns removed 10%. The starting and destination are chosen based on the possibility of the point to be visited. This leads to the situation where paths that are usually used are used even more. As a result, the busiest roads become even more crowded, while paths rarely used become even more emptier. It would be wise to use a more advanced randomisation algorithm with a more intuitive distribution of the points. In this role, ant [SHOM11] or bee colony [DMS16] movement may show better distribution, providing a 'natural' traffic flow on the roads.

Considering this imperfection, all the developed simulators still provide fair options for algorithm comparison. Even though the option to compare new versions of vehicle distribution to the starting distribution is no longer reliable, all the algorithms remain in the same conditions during progression. This allows us to draw conclusions about the progression of algorithms in the same generations and their progression between multiple simulator iterations to compare their effectiveness with one another. The only downside is the impossibility of seeing whether the traffic road system can be optimised to make navigation faster in general.

6 Conclusion

Sharing data between all drivers proved useful. It allows for control of the traffic flow on the roads and fewer traffic jams. The renavigation feature presented in this work, which also relies on sharing data, positively affected the time required to complete the selected route. However, the continuous iteration of the developed algorithm also performed better than the continuous usage of algorithms without careful path planning for the future. The final result was less impressive than expected.

Overall, the developed solution has the potential to be implemented for industrial needs. Considering the importance of navigation technologies and the introduction of autonomous vehicles, sharing data and using global control for vehicle navigation may be an optimal navigation tool. It is debatable whether sharing vehicle locations with other road members is ethically correct. However, implementing a similar algorithm for public transport, taxi, or parcel delivery, where sharing vehicle locations is acceptable, may provide better navigation options for them.

Further investigation Even though the provided algorithm shows excellent results after a single generation, its progression over generations does not show any reasonable benefits. One possible reason for such behaviour is the unnatural randomisation of requested paths. It is useful to try out the suggested algorithm with another path-requesting algorithm, either using historical data of travel paths or a reliable travel time-based network performance model as a path-picking source.

Another useful investigation direction is using this algorithm for large-scale road systems. The current solution processes a graph representing all the road networks as one object. However, it can be optimised to work on large-scale road systems, like countries, by splitting the graph representing the road network into subgraphs, performing path-finding in subgraphs, and combining them backwards into one complete path. In this case, the algorithm can be parallelised, and its performance will be increased. [PGLC23]

Extending the algorithm's renavigation part is also a crucial area for development. The current solution has demonstrated its reliability in managing significant traffic jams. However, its handling of minor road delays could have been more effective, which does not align with the expected behaviour. It may be beneficial to investigate further how to prevent traffic jams from occurring to enhance the algorithm's ability to handle traffic situations and prevent congestion.

Appendix

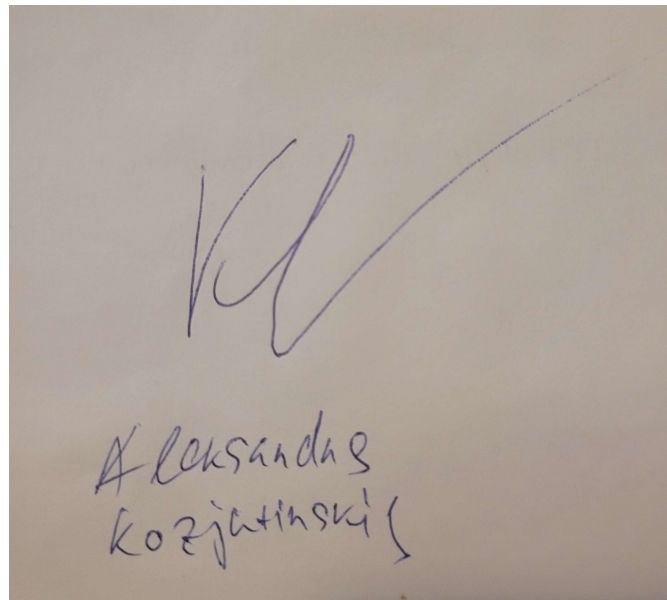
Code Repository

code: <https://github.com/Silaris/MasterThesis>

Acknowledgements

I am incredibly grateful for all the support my supervisor, Kallol Roy, provided me for this work development. It is only possible to complete this thesis with his guidance and advice. His ideas and suggestions immensely helped to avoid long debugging sessions during simulator development and made the research process inspiring.

I also want to thank my family for the motivation they provided and the careful listening to my ideas for thesis development when brainstorming was required.



A handwritten signature in blue ink, consisting of stylized initials 'AK', is positioned above the name 'Aleksandra Kozjatinski' written in the same ink. The signature is fluid and cursive, while the name is written in a more legible, slightly cursive script.

References

- [AJK22] Kunal Menda Arec Jamgochian and Mykel J. Kochenderfer. Multi-Vehicle Control in Roundabouts using Decentralized Game-Theoretic Planning, Jan 2022.
- [ASSV93] José Augusto Azevedo, Maria Emília O. Santos Costa, Joaquim João E.R. Silvestre Madeira, and Ernesto Q. Vieira Martins. An algorithm for the ranking of shortest paths. *European Journal of Operational Research*, 69(1):97–106, 1993.
- [Bel09] Michael G.H. Bell. Hyperstar: A multi-path Astar algorithm for risk averse vehicle navigation, January 2009.
- [DMS16] Mauro Dell’Orco, Mario Marinelli, and Mehmet Ali Silgu. Bee colony optimization for innovative travel time estimation, based on a mesoscopic traffic assignment model. *Transportation Research Part C: Emerging Technologies*, 66:48–60, 2016. Advanced Network Traffic Management: From dynamic state estimation to traffic control.
- [FFK13] Mogens Fosgerau, Emma Frejinger, and Anders Karlstrom. A link based network route choice model with unrestricted choice set. *Transportation Research Part B: Methodological*, 56:70–80, 2013.
- [JG17] Xuexi Zhang Le Zhang Wei Chen Zhiguang Cao Lu Zhang Hongliang Guo Jing Guo, Yaoxin Wu. Finding the ‘faster’ path in vehicle routing, October 2017.
- [KB10] Ioannis Kaparias and Michael G. H. Bell. A reliability-based dynamic re-routing algorithm for in-vehicle navigation. In *13th International IEEE Conference on Intelligent Transportation Systems (ITSC 2010)*, pages 974–979. IEEE, 2010.
- [LTL10] Xiangyong Li, Peng Tian, and Stephen C.H. Leung. Vehicle routing problems with time windows and stochastic travel and service times: Models and algorithm. *International Journal of Production Economics*, 125(1):137–145, 2010.
- [MNM22] Majid Mazouchi, Subramanya Nagesh Rao, and Hamidreza Modares. A risk-averse preview-based q -learning algorithm: Application to highway driving of autonomous vehicles, 2022.
- [PD14] Parham Pahlavani and Mahmoud R. Delavar. Multi-criteria route planning based on a driver’s preferences in multi-criteria route selection. *Transportation Research Part C: Emerging Technologies*, 40:14–35, 2014.

- [PGLC23] Alvaro Paricio-Garcia and Miguel A. Lopez-Carmona. Application of Traffic Weighted Multi-Maps Based on Disjoint Routing Areas for Static Traffic Assignment, September 2023.
- [SHOM11] Jin-Ho Ahn Sungho Kang Seung-Ho Ok, Woo-Jin Seo and Byungin Moon. An ant colony optimization approach for the preference-based shortest path search. *Journal of the Chinese Institute of Engineers*, 34(2):181–196, 2011.
- [Tom] Tomtom traffic statistics. <https://ts.tomtom.com>. Accessed: 2024-05-20.
- [Vas08] Virginia Vassilevska. *Efficient algorithms for path problems in weighted graphs*. PhD thesis, USA, 2008. AAI3323788.
- [ZCD23] Qing Mi Xing Su Limin Guo Zhi Cai, Tao Wang and Zhiming Ding. Dynamic Weighted Road Network Based Multi-Vehicles Navigation and Evacuation, March 2023.
- [ZML23] Yiming Zhao, Lei Mo, and Ji Liu. Path planning based on traffic flow prediction for vehicle scheduling. pages 1–5, 08 2023.

I. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Aleksandrs Kozjutinskis**,
(author's name)

1. grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright, my thesis

Multi-vehicle path planning using shared data,

(title of thesis)

supervised by Kallol Roy.

(supervisor's name)

2. I grant the University of Tartu a permit to make the thesis specified in point 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.
3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.
4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Aleksandrs Kozjutinskis
20/05/2024