IoT Traffic Management System for Autonomous Vehicles

Rohit Singh
Electrical Engineering
University of Waterloo
Waterloo, Ontario
rkochhar@uwaterloo.ca

Vojdan Bojcev
Electrical Engineering
University of Waterloo
Waterloo, Ontario
vbojcev@uwaterloo.ca

***Abstract*—Each new year, cities across the world see increases in the number of vehicles on the road. This rate of increase outpaces improvements in infrastructure and results in heavy congestion. With the advent of autonomous vehicles, an avenue is opening that allows the traffic control infrastructure of a city to interact with the vehicles using it. In this work, we present an model to represent traffic networks using graph theory, from which an optimized path planning algorithm is designed and tested using a discrete event simulator. The proposed system consists of a sensor network to track the number of vehicles across a traffic network, from which a novel variation of Djikstra's algorithm is used to optimized path planning to minimize traffic flow.**

I. Introduction

With the population of city centres rapidly increasing, the number of vehicles on the road is quickly becoming unsustainable for many cities' infrastructure. Densely populated cities such as Los Angeles and New York City have become notorious for their gridlock and traffic, and there is little evidence to suggest that this problem will relent any time soon.

While autonomous vehicles (AVs) propose a solution to the causes of traffic due to human error, a larger error is caused by the non-dynamic nature of many path planning algorithms, which perform routing by minimizing expected travel time. While these algorithms are improving through the collection of GPS data from mobile devices of users in the traffic network, once the path is planned it is often static unless the user chooses to change course. With the increasing adoption of autonomous vehicles, there is a unique opportunity presented to rethink the status quo in traffic routing, and change from static path planning to dynamic path planning which considers a large set of factors within the traffic network to constantly optimize and update the path of a vehicle, minimizing travel time to its final location.

In this work, we aim to propose a novel solution that transforms traditional traffic networks into dynamic sensor networks that communicate with autonomous vehicles to constantly perform optimized path planning and improve routing, allowing for reduced traffic congestion and improved flow, ultimately improving gridlock and heavy traffic build-up.

The scope of this paper is focused on the process by which the autonomous vehicles communicate to the network to determine the optimal path. For the purposes of this work, it can be assumed that the speed limit, length, and number of travelling cars is known for all roads within the network. In reality, this could be achieved by placing sensors in street lights, traffic lights, or other positions that provide a point-of-view to count vehicles on the road. Additionally, AVs can adopt an ad-hoc network in which their own positional sensors contribute to traffic information. The ideal implementation of the vehicle tracking capabilities in a network is an area that warrants further research and is out of this work's scope.

II. Literature Review

Traffic control can be realized in different ways. The most effective solution is to intelligently build roadways that are sufficiently sized and routed to maximize flow. Roundabouts are an example of renovated infrastructure that improves safety and efficiency [1]. However, building roadways is expensive, time-consuming, and practically permanent which makes them difficult to adapt to future conditions and especially temporary surges. Another way to reduce clogging is to efficiently use existing infrastructure by modulating traffic control devices. The most common form of this is variable-time traffic lights. Finally, an emerging type of traffic control is the employment of dynamic vehicle routing. Software such as Google Maps or Waze will redirect drivers if the optimal route has become congested. This not only reduces the driver's commute time but also removes one car from the queue had they continued on their original path. With the advent of autonomous vehicles, this form of traffic control is possible to achieve on a larger scale.

A. Traditional Traffic Control

Currently, congestion on existing infrastructure is typically controlled by deciding which vehicle paths are allowed to progress through intersections. We define three levels of current traffic control intersection infrastructure:

- **Uncontrolled or Minimally Controlled**: This style encompasses stop signs and fixed-phase traffic lights. In this style, there is no traffic optimization done after the initial installation. Each direction of traffic has a fixed amount of priority relative to others. The most basic example is a 4-way stop sign or a traffic light with equal time spent as green and red.

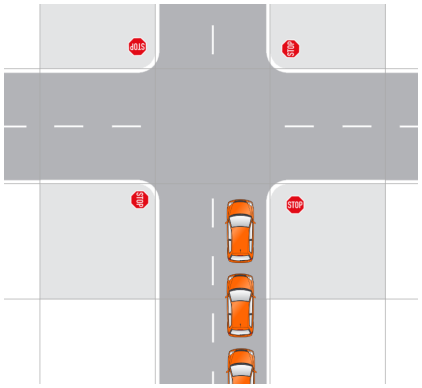

Fig. 1. Traditional 4-way stop intersection. Traffic diagrams are made using [2].

These stops are simple but flawed. In figure 1, congestion occurs because the congested path has equal priority to the other paths even though those paths are empty. When a certain path is known to be more vehicle-dense relative to others, that path may have its stop signs removed or, in the case of lights, the *green light phase time* (GLPT, the proportion of the light's cycle that is green) will be increased.

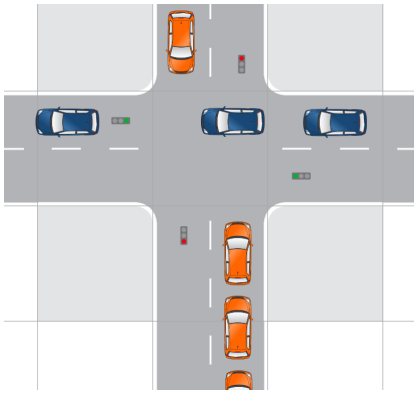

Fig. 2. Modified 4-way traffic light.

This solution fails when there are temporary surges along the path that has lower priority. Congestion occurs and the *traffic wave* effect [3] sets in and causes cascading congestion upstream of the intersection.

- **Locally Controlled**: These intersections have a dynamic GLPT that changes depending on current traffic conditions. Typically, intersections in modernized cities will have sensors underneath the pavement before the stop line [4]. If the system notices that one direction is seeing a higher number of vehicles than another, it will increase that route's GLPT. However, this form of traffic control is only local and does not consider downstream or upstream traffic effects. An intersection may relieve traffic for itself on a given direction, however if the next downstream traffic light is red for that direction, then overall that route experiences the same congestion and the critical point has simply shifted.

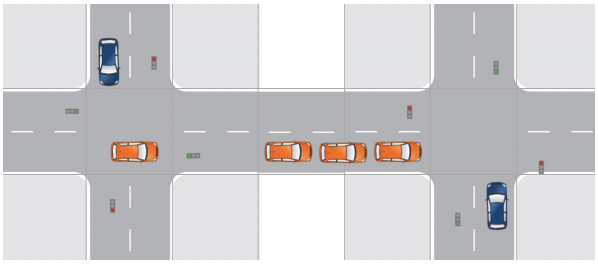

Fig. 3. Example of local control causing downstream congestion.

- **Networked**: Networked traffic management systems are able to alleviate congestion over longer routes. Individual intersections communicate with each other and contribute to a more global algorithm to reduce overall congestion. Large cities have central traffic control offices that constantly monitor and manage traffic lights to achieve this.

Tian et al [5] explores the characterization of a road system in terms of a network of nodes (intersections) and weighted edges (roads between intersections). This graph-style characterization allows for a flexible method to perform analyses and optimizations by using different functions to determine the weight of edges. In this paper, the authors investigate the assignment of edge weight in three different ways:

- **Length**: Edge weight is proportional to the length of the edge, $w_{edge} = l_{edge}$. This would be used for algorithms that minimize the total length that vehicles travel, for example.
- **Traffic Capacity**: This weight assignment $w_{edge} = c_{edge}$ can be used for routing vehicles for the purpose of maximizing road usage for cases such as rush hour or alleviating congestion.
- **Traffic Efficiency**: Defined as the ratio of capacity to length, $w_{edge} = e_{edge} = \frac{c_{edge}}{l_{edge}}$. This provides a hybrid between the first two weight assignments. Maximizing efficiency along a path provides a balance between short distance (as $e_{edge} \propto \frac{1}{l_{edge}}$) and low congestion (as $e_{edge} \propto c_{edge}$.

After network characterization, we explore traffic management methods. *Saad et al* [6] outlines the use of Ant Colony Optimization (ACO) in finding optimal routes. Essentially, for any two points in a city the algorithm marks the paths of all the vehicles that travel between them (the path could be the entire trip of the vehicle or merely a subset). Over time, a dominant path emerges that most vehicles have taken as it is faster. Subsequent travellers will then know of the fastest path. This is an actuator-less system; it only monitors vehicles to later provide information that may reduce congestion. However, if all vehicles in a city are autonomous, they may be considered actuators as their behaviour would be decided by this algorithm.

Göttlich et al [7] applies the graph characterization of road networks and proposes a traffic light flow control algorithm. Traffic junctions are modeled as nodes of a directed graph. Additionally, each lane is its own edge, meaning that multi-lane roads are represented with parallel edges and lane splits are sub-junctions.

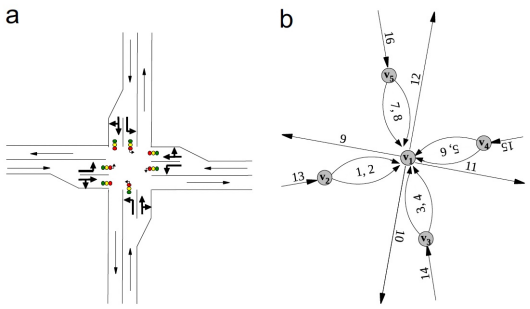

Fig. 4. Example from *Göttlich et al* showing the characterization of a multi-lane junction.

Vehicle density and flow rate is calculated and stored along each edge, continuously. At junctions, the flux is computed. This continuous computation along an edge provides much more accuracy and granularity compared to computing a single value as the edge's weight (as in traditional graph algorithms) and intuitively also allows for traffic waves (that propagate continuously along a road) to be expressed. The paper explores several approaches to traffic lights switching, including cell transmission, fluid dynamics, and heuristics-based models. The authors were able to devise an algorithm for traffic light switching decisions that theoretically significantly reduces congestion. However, for a grid road network with only 9 junctions, computation times exceeded reasonable limits with a large number of variables. In the worst case, computations took over an hour to complete for a simulated time of 400 seconds. this shows a possible issue for such algorithms: all gathered data (vehicle locations, road density, etc.) is funneled into one centralized algorithm that is responsible for making traffic flow decisions for an entire city. Decentralizing such an algorithm (for example, by running one instance of it for each smaller segment of a city) does not alleviate the issue because there is no algorithm to manage the border of city segments. Instead, we now examine congestion alleviation through individual vehicle routing.

B. Dynamic Vehicle Routing

With autonomous vehicles on the temporal horizon, there is another option for traffic management: dynamic travel routes.

Reducing congestion through infrastructure as discussed above is the only option for human-operated vehicles. This is because each driver selects their own route and typically does not deviate to avoid traffic unless they are using route planning applications such as Google Maps, which is typically not the case during times like *rush hour* [8] when drivers are typically following known, repetitive routes. However, autonomous vehicles provide another option. As they will be fully computer-controlled and networked, it is possible to control congestion exclusively by dynamically routing vehicles. That is, when a vehicle becomes aware of congestion ahead on its route (which it knows from the other vehicles reporting their location), it can find other routes to its destination that are possibly faster. The reason that this method is significant specifically for autonomous vehicles is because route changing happens automatically (unlike human drivers) and on a much larger scale. When several vehicles deviate from a route onto a faster one, that original route also has its congestion alleviated.

Liang et al [9] examines the problem of routing autonomous taxi-cabs. This is technically a separate problem to centralized traffic control in that each vehicle is fending for itself and trying to reduce its travel time which does not necessarily reduce congestion. However, if every vehicle is reducing its travel time and/or distance travelled, then the entire road network will see an overall improvement. The simulations mainly compare what happens when a travel time for a route is strictly maintained compared to allowing for a dynamic travel time. An important finding is that allowing for some increase in total travel time by staggering vehicles both reduces congestion and time spent idling.

III. Experiment Methodology

To validate the findings of this paper, our experiment consists of three main sections.

1) **Traffic Network Model**: This section discusses how our proposal represents traffic networks as sets of nodes and edges combined to form directional graphs.
2) **Optimized Path Planning**: This section outlines how our proposal performs optimized traffic routing on the graphical representation of the traffic network.
3) **Discrete Event Simulation**: This section describes the discrete event simulator created to validate the efficiency of our optimized path planning model compared to traditional routing methods.

A. Traffic Network Model

For our experiment, we need to create a model that can represent traffic flow between intersections. To accomplish this, we implement a directed graph where intersections are represented by nodes and roads by edges. Nodes within the graph are communication points between the network and the vehicles within it. As a vehicle passes by a node, there is a quick exchange of information between the vehicle and the network to see if the vehicle's current path is still optimal and whether or not it requires updating.

Each edge contains information on the speed limit, length and current amount of occupying vehicles for the road which it represents. Nodes of the graph constantly update the number of vehicles that occupy an edge. Based on the number

of cars currently occupying the edge, various parameters are calculated to determine the traffic flow on that edge, which is used as a routing metric for the optimized path planning algorithm. In the following subsections, the implementation of nodes and edges are discussed in further detail.

1) Road (Edge) Modelling: The traffic flow of a whole network is directly dependent on the traffic flow of all edges within it. Therefore, in the process of optimizing the traffic flow of a network, we must first create a model that quantifies congestion along an edge. As mentioned earlier, the implementation of our model assumes that three key pieces of information are known about each edge within the network:

1) **Length**, L: The distance a vehicle travels while on the edge, in kilometres.
2) **Speed Limit**, v: The assumed maximum average speed a vehicle moves at while on the edge, in kilometres per hour.
3) **Number of Vehicles**, N: The current number of vehicles driving on the edge at a given point in time.

With these three known parameters, key metrics describing traffic flow within an edge can then be defined:

- **Minimum Travel Time**, t_{min}: The time it takes for a vehicle to complete a trip across an edge, in seconds, assuming no delays. This parameter is calculated by the simple equation:

$$t_{min} = \frac{L}{v} \times 3600 \tag{1}$$

- **Distance Factor**, d: The distance between vehicles in an edge, in meters.
  - To model the distance factor, we define the following equation which gives a measure of the amount of free space on an edge per vehicle.

$$d = \frac{1000 \times L - N \times \text{Car Length}}{N} \tag{2}$$

  - This gives the average distance between vehicles along the edge, which serves as the basis for congestion estimations. The less space between two vehicles on an edge, the more congestion occurs, causing a reduction in traffic flow.
  - The distance factor equation provided here is an initial estimate to test the validity of our model. The optimal distance factor equation presents the opportunity for further research, but is out of the scope of this work.
- **Real Travel Time**, t_{real}: The actual time it takes for a vehicle to complete a trip across an edge, in seconds.
  - Unlike the minimum travel time, t_{min}, this travel time is dependent on the distance factor, d, and thus the amount of vehicles currently on the edge.
  - To model how the real travel time changes depending on the number of vehicles on the edge, we must consider how in a queue of vehicles, a change in speed at the front of the queue takes time to propagate to the back of the queue. This phenomenon is known as the *concertina effect* or *accordion effect*, and is a predominant cause of traffic jams [3]. This effect is caused by the nonzero time it takes for vehicles to react to cars ahead, and it becomes more pronounced with longer queues. Additionally, while the concertina effect has been studied for human drivers, it would remain a non-negligible phenomenon in autonomous vehicles as they also have nonzero processing times. To model this effect, we define the actual speed due to traffic delays caused by a large volume of traffic, v_{real}. v_{real} is inversely proportional to d, since with less distance between cars, there is more compounding delay experienced across the vehicles on the edge. We define v_{real} as follows:

$$v_{real} = \begin{cases} v & d > 10 \\ v(1 - \frac{1}{d}) & 10 \geq d > 1 \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

    This equation is an accurate model for slow-down among vehicles due to large volumes of traffic, as we see a sharp decrease in v_{real} as d approaches values smaller than 2 meters, which can be interpreted as the significant decrease in speed during bumper-to-bumper traffic.
  - From equation 3, we define t_{real} as:

$$t_{real} = \frac{L}{v_{real}} \times 3600 \tag{4}$$

From these calculated metrics, our network is able to model the expected travel patterns of vehicles within the network in terms of minimum travel times assuming no congestion and maximum speed, as well as real travel patterns of vehicles within the network in terms of real travel times that assume congestion and non-ideal speed.

2) Traffic Intersection (Node) Modelling: The nodes in our network are the communication points between the network and the vehicles within it. As a vehicle passes by a node, there is a brief exchange of information in which the vehicle communicates its target location and current path, from which the node checks whether the path is still optimal based on the routing algorithm presented in section III-B. If the node determines a faster path exists, it communicates the new path to the vehicle. This process is outlined by the pseudo-code function shown in algorithm 1.

Due to the volume of vehicles in the network, it is important that nodes have enough computational power to quickly compute the optimal path to any node within the network. This can be ensured by designing the path planning algorithm to have minimal complexity, or by providing enough resources to each node. This work focuses on the former optimization, and leaves the ideal implementation of the nodes as an area for further research.

Algorithm 1 Vehicle-to-Node Communication

if communication established **then**:
    AV sends node target location and current path
    Node computes optimized path
    if Optimized Path is different than current path **then**
        Node sends optimized path
        Vehicle updates current path
    else
        Vehicle keeps current path
    end if
end if

By ensuring that each node can quickly communicate optimized paths to the vehicles, the network is highly adaptive to current conditions, creating a highly dynamic network that, in theory, routes traffic very efficiently and minimizes traffic flow.

B. Optimized Path Planning

To find the minimum travel time for a vehicle in the network, we propose a variation of Djikstra's algorithm [10] to compute the shortest path in terms of the real travel time of the graph's edges, t_{real}.

Algorithm 2 Path of Minimum Travel Time Using Djikstra's

for N in nodes **do**
    N.isVisited ← false
    N.tentativeDistance ← 0
    while all N.Neighbours.isVisited = False **do**
        for E in N.outgoingEdges **do**
            edgeToNodeTime ← E.realTime + N.tentativeDistance
            if edgeToNodeTime ≤ E.sinkNode.tentativeDistance **then**
                E.sinkNode.tentativeDistance ← edgeToNodeTime
            end if
            E.sinkNode.isVisited ← true
        end for
    end while
end for

Since we have to iterate across all nodes and their connecting edges within the network, the worst case complexity occurs when all nodes are connected to each other. In this case, our algorithm has a squared complexity of $N^2 \times E$, where N represents the number of nodes in the network and E represents the number of edges.

Due to the worst-case squared complexity, the networks should be configured in a way that maximizes coverage while minimizing nodes. This optimization problem presents room for further research that is out of the scope of this work, however the implementation of the model presented in this work within a distributed traffic network system could present promising results. In such a configuration, the nodes could be assigned as local and global nodes, with the local nodes communicating information about a subset of the graphs edges to each other and a single global node, with the global nodes then communicating information about the entire network to each other. For the sake of this work, we define test networks which have a significantly reduced complexity than that of the worst-case, allowing us to verify the functionality of our optimal path planning network to allow for future adoption in more complex traffic systems.

C. Discrete Event Simulator Implementation

Now that our graph definitions are complete, we implement a way to model the behavior of the AVs when they are travelling along an edge. To do this, a *discrete event simulator* (DES) is implemented. The DES serves as the clock of the entire simulation environment, maintaining a global reference time across all nodes, edges and vehicles. Since our edges are defined by speed, length, and travel time, we can mathematically define the average speed of any vehicle along a given edge at a given moment. From this calculation, we are able to determine the amount of time left before the vehicle completes a trip along the given edge and begins a trip across another.

The simulator is given a topological graph G from the definition of various nodes and their connecting edges described using an adjancency matrix. The speed limit and distance of each edge is declared at edge creation, from which the remaining parameters outlined in section III-A1 are derived. Once the network is defined, AVs must be added to the edges to model congestion and traffic flow. This is done by creating a number of vehicles, N, with declared or randomly assigned start and end nodes, and adding them to a collection of *waiting* vehicles, W. Once the simulation starts, these vehicles are gradually added into the system by random selection, moving them from the *waiting* collection to the *travelling* collection T, and their *baseline path* is computed using Djikstra's algorithm with the minimizing edge weight being *minimum travel time*. This baseline path simulates routing uninformed by traffic flow, and serves as the benchmark for which the efficiency of our model is compared against. Once each AV is deployed into the graph, they travel along every edge in their path, with their trip progress tracked by the global reference time t maintained by the simulator. Once the vehicles have traversed all the edges in their route, we determine that they have reached their target node and they are added to the *completed* collection, C. Once all vehicles are added to the completed collection the baseline results are collected. This process is outlined in the pseudocode function shown in algorithm 3.

IV. Experimental Results

A. Experimental Network Design

To accurately simulate the behavior of our model in various situations, we have designed multiple graphs to simulate different types of traffic networks found in the real world.

Algorithm 3 Baseline Simulation

$G \leftarrow$ Nodes, Edges
$W \leftarrow$ addVehicles(N)
$T \leftarrow$ [ ]
$C \leftarrow$ [ ]
while len(W) $\neq 0$ and len(C) $\neq$ N **do**
  if len(W) $\neq 0$ **then**
    if $t \mod 10 = 0$ **then**
      v = random(W)
      getBaselinePathInfo(v)
      W.remove(v)
      $T \leftarrow v$
    end if
  end if
  for $v \in T$ **do**
    if v is done on current edge **then**
      if v has traversed path **then**
        $C \leftarrow v$
        T.remove(v)
      else
        setNextEdge(v)
      end if
    end if
  end for
  $t = t + 1$
end while

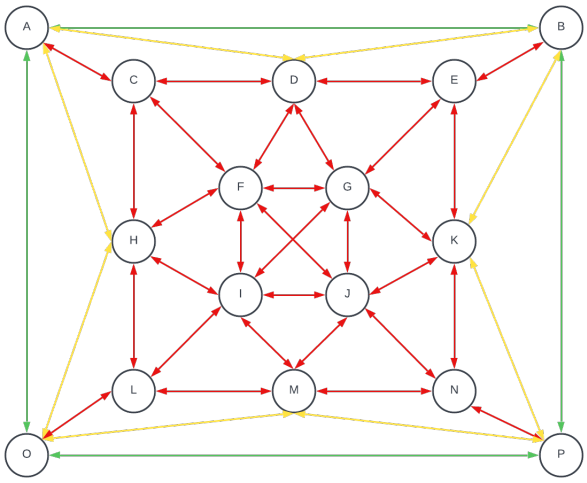

Fig. 5. Test Network 1

The motivation for each graph's layout and the experimental results are discussed in the next section.

B. General Performance Network

First, the graph designed in figure 5 was implemented to test our model's performance in a general network comprised of a variety of roads. This network was designed to validate the general performance of our models across three main types of roads making up typical traffic networks:

1) **Highway Roads**: These roads are represented by the green edges in our graph. They are characterized by their high speeds and long distances, with each edge of this type having $L = 4$km and $v = 110$ kph.
2) **Rural Roads**: These roads are represented by the yellow edges in our graph. They are characterized by their medium-high speeds and long distances, with each edge of this type having $L = 2$km and $v = 80$kph.
3) **City Roads**: These roads are represented by the red edges in our graph. They are characterized by their low speeds and short distances with many connecting nodes. Each edge of this type has $L = 1$km and $v = 30$kph.

These three types of edges have varying degrees of susceptibility to congestion. Highway roads are designed to prevent congestion, as they have no intersections across their length, instead opting for on-ramps and off-ramps as traffic exchanges. This, in theory, creates a road in which traffic flow remains high and traffic is not an issue. However, this results in a large amount of traffic occupying these roads, resulting in heavy volumes of vehicles condensed together. This results in the *concertina effect* discussed in section III-A1, causing braking delays to propagate through chains of vehicles close to one another resulting in a slow-down along the edge, as modelled by equation 3. Rural roads often consist of longer stretches of medium speed roads with minimal intersections, and depending on the volume of traffic, can be viable alternatives to highway roads. Finally, city roads are highly prone to congestion, and typically contain a large amount of intersections controlled by traffic lights that result in significant slow-downs.

For the baseline, unoptimized result, we assume the cars are first routed by the minimum path determined by ideal travel time, which is the travel time assuming that there is no congestion and optimum travel flow. Initially, this will result in ideal traffic flow, but as a large volume of vehicles plan their paths this way, there will be a large build up of congestion along the paths with the ideal minimum time, causing congestion and slow-downs.

In simulating the network presented in figure 5, 400 vehicles were added to the network before it became completely saturated. Of these 400 vehicles, a total of 28 took an optimized path. Resulting in an optimization percentage of 7%. The comparison between baseline travel times and optimized travel times can be seen in figure 6. From the results of this simulation, it was observed that the travel time of optimized vehicles was decreased by an average of 11.34%, as shown in figure 7. These results are largely in line with the expected behavior of the model, with the notable exception of the optimized vehicle with ID=11. This vehicle seems to have been routed across a less efficient path. This is likely due to other cars also being routed to edges along this path, that were not considered in the initial computation of the ideal path, causing the vehicle to reach

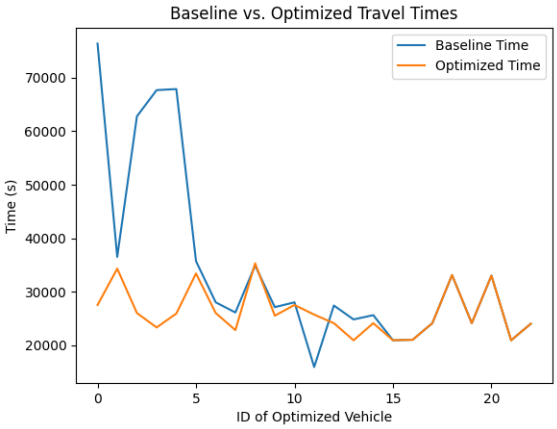

Fig. 6. Test Network 1: Baseline Travel Times vs. Optimized Travel Times

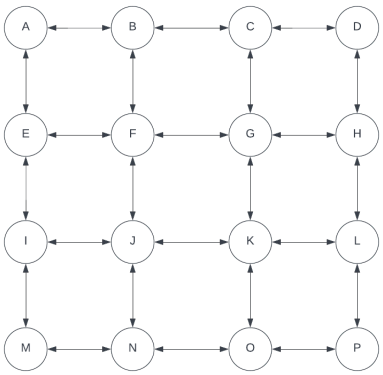

Fig. 8. Test Network 2

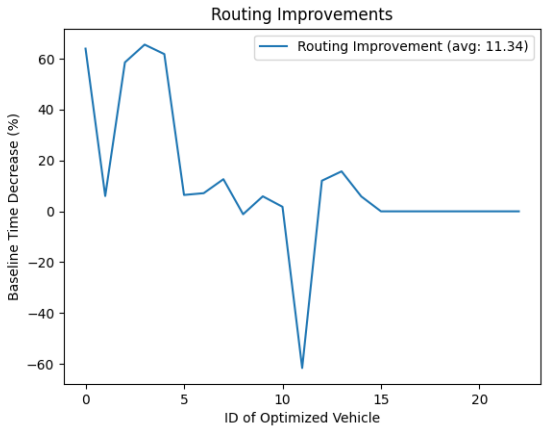

Fig. 7. Test Network 1: Routing Improvements

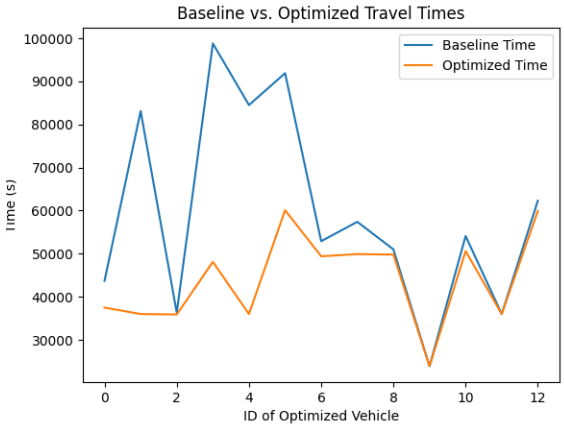

Fig. 9. Test Network 2: Baseline Travel Times vs. Optimized Travel Times

the path and find it is more congested than calculated. This presents an opportunity for further research where the model can take into account vehicles en route to a specific edge, and incorporate them into the congestion calculation.

Further, it is worth noting that although sometimes the optimized path yields the same time as the baseline path, this nonetheless improves congestion, as it balances the load across multiple paths with the same travel time. By referring to figure 5, we can observe that for a vehicle travelling between node A and node P, the minimum path is either node A $\rightarrow$ node O $\rightarrow$ node P, or node A $\rightarrow$ node B $\rightarrow$ node P. Although the baseline path may originally be calculated as the former, our model takes into consideration how the addition of a vehicle to that path will affect the congestion and route it accordingly.

C. City Street Network

As discussed earlier, the goal of this model is to improve gridlock and traffic congestion in highly populated city centres. To test our model's efficiency in improving the problem, we designed the traffic grid shown in figure 8.

This network was designed to test our model's performance exclusively on city streets, so each edge is loaded with the same speed limit and length used for city streets in section IV-B of $v = 30$kph and $L = 1$km, respectively. This network is highly prone to congestion, due to the slow speeds and short distances of the edges combined with the high number of vehicles that occupy them. Because of this, we expect our model to provide better optimization than the network examined in section IV-B.

In simulating the network presented in figure 8, 150 vehicles were added to the network before it became completely saturated. Of these 150 vehicles, a total of 13 took an optimized path. Resulting in an optimization percentage of 9%. The comparison between baseline travel times and optimized travel times can be seen in figure 9. From the results of this simulation, it was observed that the travel time of optimized vehicles was decreased by an average of 19.05%, as shown in figure 10.

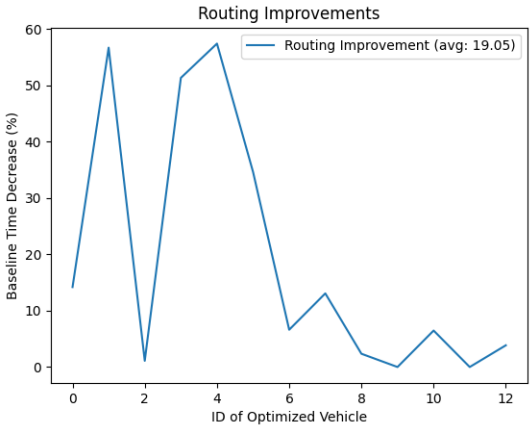

Fig. 10. Test Network 2: Routing Improvements

These results confirm our hypothesis, as the model has significantly higher routing improvements in this city network compared to the network examined in section IV-B. This network had a significantly reduced capacity due to its reduced size. In contrast to the general performance network which had a total length of 62 km, this network had a total length of 24km. This is the reason that only 150 vehicles could be added to the network before it overflowed, still the relatively high optimization percentage and model efficiency on a small volume of traffic are encouraging.

Although it was expected that the optimization percentage would be higher, it is worth noting that optimizing a single vehicle within the network improves congestion for all vehicles throughout the network. When the path of a single vehicle is optimized, it frees up space on a congested edge that may benefit another vehicle that enters the network.

V. Conclusion

In this work, we have presented a model which represents traffic networks using graph theory, a variation of Djikstra's algorithm to optimize path planning with the network, and a discrete event simulator to validate the model's performance across various test networks. This work allows for a significant amount of further research to be conducted in the area of traffic optimization. The model presented is highly customizable, and allows for future researcher to contribute improvements to the proposed solution. We believe that the following areas present promising areas for future research in the field of traffic optimization.

A. Implementation of Tracking Nodes

Our model relies on the implementation of tracking nodes throughout the traffic network, but our discussion omitted specific discussion into the ideal implementation. Our model is created to be highly flexible with the placement of nodes, allowing them to be theoretically placed anywhere, with two promising examples being traffic lights and street lights. The more nodes in the network, the more distributed the load and more accurate the tracking. This would allow for the path planning algorithm to be informed by more accurate data, and the amount of routing each node would have to do would be reduced. This could allow for the implementation of low cost sensors in a highly connected network, which we believe presents the most ideal implementation of the model presented here. Future research in this area could determine the amount of computational resources required per node, and the optimized tracking methods and location for optimal node placement.

B. Optimization of Path Planning Algorithm

As mentioned in section III-B, our current implementation has a squared worst-case complexity, and although in our simulations we did not experience excessively slow runtimes, for this network to be deployed in highly connected node networks running computations for a high volume of vehicles, it has to be as fast as possible. As mentioned, reducing the number of nodes in the entire network would help alleviate this problem, so the implementation of the model presented here in a distributed network shows promise to address the complexity problem. Future research in this area could focus on the best way to create traffic networks modelled as distributed networks, or improvement to the algorithm to bring the complexity to either linear or ideally logarithmic time.

C. Improved Usage of Computational Resources

In this work, we assumed that the nodes perform the computation, since they are connected to one another and have a plethora of data available for congestion calculations. Alternative approaches could depend on the communication of larger data packets containing more information about the network to the autonomous vehicle's computing system, which could then perform the optimized path planning. There are numerous opportunities for research in this area which were omitted from the scope of this work so that the implementation of the optimized path planning algorithm could be the main focus. Future research in this area could determine the optimal way to balance the computational load between the vehicles and the nodes. Such research could determine methods by which the computational power of a node is minimized, resulting in a cheaper implementation of the network.

References

[1] R. L. Reid, "Modern roundabouts boost traffic safety and efficiency," 2021. [Online]. Available: https://www.asce.org/publications-and-news/civil-engineering-source/civil-engineering-magazine/issues/magazine-issue/article/2021/03/modern-roundabouts-boost-traffic-safety-and-efficiency.

[2] *Accidentsketch*. [Online]. Available: https://www.accidentsketch.com/.

[3] C. F. Daganzo, "The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory," *Transportation Research Part B: Methodological*, vol. 28, no. 4, pp. 269–287, 1994, ISSN: 0191-2615. DOI: https://doi.org/10.1016/0191-2615(94)90002-7. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0191261594900027.

[4] B. C. M. of Transportation and Infrastructure, "5 things that make traffic signals change," [Online]. Available: https://www.tranbc.ca/2014/01/30/5-things-that-make-traffic-signals-change/.

[5] Z. Tian, L. Jia, H. Dong, F. Su, and Z. Zhang, "Analysis of urban road traffic network based on complex network," *Procedia Engineering*, vol. 137, pp. 537–546, 2016, Green Intelligent Transportation System and Safety, ISSN: 1877-7058. DOI: https://doi.org/10.1016/j.proeng.2016.01.290. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705816003179.

[6] A. A. Saad, H. A. E. Zouka, and S. A. Al-Soufi, "Secure and intelligent road traffic management system based on rfid technology," pp. 41–46, 2016. DOI: 10.1109/WSCAR.2016.9.

[7] S. Göttlich, M. Herty, and U. Ziegler, "Modeling and optimizing traffic light settings in road networks," *Computers & Operations Research*, vol. 55, pp. 36–51, 2015. DOI: 10.1016/j.cor.2014.10.001.

[8] E. M. Laflamme and P. J. Ossenbruggen, "Effect of time-of-day and day-of-the-week on congestion duration and breakdown: A case study at a bottleneck in salem, nh," *Journal of Traffic and Transportation Engineering (English Edition)*, vol. 4, no. 1, pp. 31–40, 2017, Special Issue: Driver Behavior, Highway Capacity and Transportation Resilience, ISSN: 2095-7564. DOI: https://doi.org/10.1016/j.jtte.2016.08.004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2095756416300290.

[9] X. Liang, G. H. de Almeida Correia, and B. van Arem, "An optimization model for vehicle routing of automated taxi trips with dynamic travel times," *Transportation Research Procedia*, vol. 27, pp. 736–743, 2017, 20th EURO Working Group on Transportation Meeting, EWGT 2017, 4-6 September 2017, Budapest, Hungary, ISSN: 2352-1465. DOI: https://doi.org/10.1016/j.trpro.2017.12.045. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146517309420.

[10] *Introduction to Algorithms*. MIT Press, 2022.