**Dissertation Left To Do**

Finish Linear Optimization – read textbook, objective function, yellow phases?

Results:

Compare both versions of my model to sumolight ones – train across 3 diff densities maybe?

Do statistical testing between the two if necessary

Take forward best model and train full version on emv’s and ped (hyper-param tuning?)

Compare this model to results of LDO model

Write report

Cool Model names:

PREDICT (Pedestrian, Roadway, and Emergency Dynamic Intersection Control with Traffic forecasting)

CIPHER (Control of Intersections for Pedestrians, Highway vehicles, and Emergency Responders)

PACE (Pedestrian, Ambulance, and Car Efficiency system)

**Dissertation Notes**

**1**

Linear Optimization queue transmission formula could be cited from here:  
*Traffic Signal Control in an MPC Framework Using Mixed Integer Programming M.A.S. Kamal∗ J. Imura∗∗ T. Hayaka*

Or alternatively from here :

A white background with black text

AI-generated content may be incorrect.

**2**

The world-wide cost of traffic congestion is huge. For instance, in the EU this cost is estimated to be 1%of its GDP [2].

**Dissertation Plan**

**Title:** “Exploring the Effects of Enhanced Environmental Data for Real-World Traffic Signal Control Using Reinforcement Learning”

“CIPHER: A dynamic, real-world traffic signal control solution that makes use of increased environmental data”

**Abstract:**

**Table of Contents:**

**Introduction / Motivation:** can reuse mostly from interim report

**Problem Background: :**introduce RL (and LO? At a very basic level) at a high level

**Related Work:**

Need 20/30 papers, currently have about 16 RL based papers with . Other papers to study:

\* Analysis of fixed-time control

\* Max pressure control of a network of signalized intersections Pravin Varaiya

\* State-of-art review of traffic signal control methods: challenges and opportunities

\* Traffic Lights Control Using Reinforcement Learning: A Comparative Study

\* Adaptive Traffic Light Control Through Reinforcement Learning Based on Sensor Integration

\* Traffic Signal Control in an MPC Framework Using Mixed Integer Programming - M.A.S. Kamal∗ J. Imura∗∗ T. Hayaka

\* Mixed Integer Linear Programming for Traffic Signal Control - Iain Guilliard

\* An optimization modeling of coordinated traffic signal control based on the variational theory and its stochastic extension

\* Optimization of Traffic Signal Settings by Mixed-Integer Linear Programming: Part I: The Network Coordination Problem (and part 2, can cite separately I think😉)

\* Can cite all the algorithms used in my testing section

**Description of Work:**

Developed RL agent focusing on the inclusion of pedestrians and EMVs

Studied the effect of giving wider environment info into the model

Validated results through linear optimization problem formulation and comparison to other algorithms

**Technical problem description –** could this just be the equations from my LDO model?

?

**Methodology:**

Talk about state, action, reward design for my RL model

Talk about experimentation between formulation of my LDO model

**Implementation:**

Can include design / implementation here I think, could reuse UML diagram from interim here?

Simulating arrivals through poisson distribution

SUMO!!

Could talk about workflow between local machine and GPU cluster, as well as

Talk about base 3 representation of traffic lights  
**Evaluation / Results:**

Do we do statistical testing between my two models?

Talk about computational resources used in training my two models

Show boxplots comparing my cars only model to all others in sumolights, show charts for low, medium and high-density scenarios

Then run full model and compare to LDO model

**Conclusions and Future Directions**

**Summary and reflections:**

Methodology is where we talk about the two different LO models

“The report provides an excellent introduction and motivation for the problem, clearly highlighting its importance, linked to the current literature. The literature review provides good depth and demonstrates understanding. It contains a critical analysis of previous work, and is used to identify shortcomings in the current state of the art. Section 3 could provide improved contextualisation and detail on the problem and approach. Some background on the methodologies would aid the reader. Section 4 could be improved by giving them an increased focus on the scientific contributions instead of the current "process" based approach. One good way of achieving this is by considering what may be relevant to the reader from a scientific perspective. Section 5 could be strengthened by providing additional detail on the experimental results (e.g. computation times). There are some very minor typographical errors across the report.

In summary, the report starts in an excellent manner, with good rigour and depth. I would recommend maintaining the same rigour in later sections of the report

“

Linear Optimization – format on/off times as continuous variables representing the time the light was switched on / off

Testing Plan

Model state factors:

Queue\_lengths

Phases\_array

Maybe wider environment context depending on initial results?

Emv\_distances

Pedestrian\_wait\_times

Model A testing:

* Do we use model trained on all state factors or just car specific ones? Just car specific ones
* Can compare to FixedTime, MaxPressure and Ducroqc et al:
  + [DQN-ITSCwPD/env/custom\_env/sumo\_env.py at 9ab0dc5ce2562af47245ac4b0550e19843f15cae · romainducrocq/DQN-ITSCwPD](https://github.com/romainducrocq/DQN-ITSCwPD/blob/9ab0dc5ce2562af47245ac4b0550e19843f15cae/env/custom_env/sumo_env.py#L19)

Model B testing:

* Run full model with all state info and compare to results from LDO programme

Talk about CO2 in results section

**Technical Problem Description – can probably expand on this more? What else should be included,should we just make this the LDO section?**

The traffic signal control (TSC) problem involves finding the optimal sequencing of various traffic light phases in order to maximize or minimize a certain evaluation metric, whilst ensuring satisfaction of various constraints such as driver safety, fairness, yellow-timings or EMV prioritisation.

In this context a traffic phase can be defined as specifying the set of lights that are green, yellow and red, and the time for which that specific phase will be active for.

The problem has been applied to numerous road networks of varying size and complication, the most typical scenario in which the TSC problem is experimented on is the classic 4 way intersection (shown right), due to the large amount of incoming traffic and competing to cross the centre of the junction.

**Problem Background**

This section introduces the technical preliminaries of reinforcement learning that are required to understand this paper.

Reinforcement Learning

Reinforcement learning [CITE HERE]? is a subset of artificial intelligence that enables a model to learn through trial and error, rather than utilizing a specific training dataset. Reinforcement learning leverages the Markov decision process (MDP). A framework for decision making in an environment with uncertain outcomes. In RL there are 5 main components to consider:

* Agent – the learner or decision maker
* Reward – A numerical variable returned to the agent that indicates how favourable a particular state is.
* Actions – the set of actions the agent can undertake; this action is executed on the current environment to produce a new state
* State – represents the current configuration of the environment in which the agent is deployed
* Policy - The set of learned actions the agent will take when in a particular state

RL agents select the most appropriate action based on the long-term reward they can expect to see from it, this value is called a Q-value and there is one assigned to every state x action pair. The Q-values are stored in a Q-table and learned through a process called Q-learning. Q-values are updated using the Bellman equation [insert bellman eq here].

A red arrow pointing to a number of mathematical equations

AI-generated content may be incorrect.

The table is initialized with all Q-values of 0, the model will pick an action, measure the immediate reward returned by the environment, update the Q table and repeat. After multiple iterations a well fit Q-table is ready to be used as a policy which will instruct the model on the most optimal action to take within a specific state.

Deep Reinforcement Learning

Classical reinforcement learning will struggle to scale for large problems due to the curse of dimensionality – as the state / action space grows so will the respective Q-table. This large Q-table soon becomes computationally infeasible to store and update values. This issue is particularly common in environments with continuous state spaces or a diverse set of potential actions.

To address this limitation, DRL replaces the Q-table with a neural network that is used to approximate Q-values and derive an effective policy. The bellman equation is still used, serving as a comparison value to backpropagate weights and biases in the network. The model receives state information through the input nodes, and outputs Q-values for each possible action,the action with the largest Q-value is enacted upon the environment. Pseudocode for the DRL training process is defined below

A crossroad with a crosswalk

Description automatically generated**Methodology**

The TSC problem has been applied to a variety of different road networks and environments. This paper will focus on the most common one: a classic 4 way intersection with 3 incoming lanes and one outgoing lane in each way (shown right). Across each incoming way is also a pedestrian crossing. With this configuration there are 16 traffic lights that must be configured as one of three potential states to form the phases.

Reinforcement Learning Methodology – Agent Design

State

The state representation of the agent combines both continuous and discrete factors to effectively represent the wide range of road actors present in a simulation at time t. The state of an agent at time t can be defined as:

Queue lengths – The number of cars currently waiting in queue j at time t. A car is only defined as ‘waiting’ if it’s current speed is 0.

Phase status – A binary array representing whether a light is green or not at time t, since the state is captured before a phase transition occurs, information on yellow phases is not relevant here.

EMV distance – The distance of the nearest EMV approaching from each direction.

Pedestrian flag – A binary value indicating whether the number of pedestrians at a crossing is exceeds 0.

Enhanced environmental data – to satisfy one of the experimental goals of this paper, investigations were carried out upon the effect of providing the model with the expected number of vehicles arriving per lane before the next action, decision, results of this experiment are discussed in section [INSERT HERE].

To enhance the numerical stability of the model, all continuous state information was normalized in relation to one another to be between 0 and 1. This is with the exception of EMV distances, which were normalized to be between 0 and 1 according to a known upper and lower bound.

Action

The action space for the model consists of 36 different traffic light phases, phases are not constrained to a cyclical structure, but each phase is forced to be active for 30 simulation time steps.

As observed by [CITE HERE], providing models with a large action space will result be more expensive in both time and computational resources, when training the model to converge on an optimal policy. For the environment defined above, there are 16 traffic lights each with 3 possible states (red, yellow, green). This gives 3^16 = 4096 configurations for the traffic lights. However the majority of these are either unsafe or inefficient, the following assumptions and constraints have been applied to reduce the action space down to 36.

• Every green phase of a traffic light must be preceded and succeeded by a yellow phase, in the interests of driver safety. This is handled externally by the environment; the model does not need to predict yellow values. This reduces the action space size to 256.

• Routes that co-inflict with one another (e.g. vehicle paths from two different routes would cross) should not be active simultaneously as this would result in crashes or gridlock. This removes 45 phases, leaving an action space of size 211.

• The most efficient implementation will have the maximum number of routes (4) active at the same time. This assumption further narrows down the action space to just 36 potential phases the model can select.

**justifying the choices made for the project, where possible with supporting evidence derived from the existing work.**

Reward

The reward for the model is defined as follows: min([INSERT REWARD HERE])

The reward function is defined so as not to penalize the model that entered the simulation during the execution of the traffic phase and thus the model was previously unaware of. It gives the model incentive to minimize pedestrian and vehicle wait times. In addition to this, a collision penalty is applied to reward safe routing of emergency vehicles. Some pre-existing work has implemented a squared reward function, so as to penalize longer waiting times more; however, it was observed that this would cause numerical instability in the CIPHER model and was therefore not included.

Linear and Discrete Optimization Methodology

**TODO**

**Implementation**

Simulation

As concluded by numerous literatures [CITE HERE], SUMO is widely accepted as one of the most effective and popular road traffic simulation tools available for use. Due to its flexibility, GUI and strong integration capabilities, it was decided that SUMO would be used to model the environment within which the agent operates.

The lack of extensive real-world data sources meant that data augmentation strategies had to be used for training the agent. Vehicles were generated according to a Poisson distribution [INSERT EQUATION HERE] with a mean inter-arrival rate of 2 simulation time steps.

It was noted across the literature that there was a lack of simulation parameters provided to the reader which limits readability. To improve this, all relevant simulation parameters are included below in table [INSERT HERE].

Code

Communication through SUMO has originally handled via the traci API, a web-socket based library provided by the SUMO developers for retrieving and updating various simulation parameters. However, due to the high communication overhead caused by the socket-based protocol used the project was migrated to libsumo (a more efficient C++ API). This change significantly reduced training times for the model.

The core implementation was written in Python, leveraging libsumo to interface with the simulation environment. The system architecture consists of:

* Environment Wrapper: Provides a standardized interface for the RL agent to interact with SUMO.
* Simulation Runner: Loads a specific traffic signal control (TSC) policy and steps through the simulation, with an optional GUI for real-time visualization.
* Controller Modules: Two controllers were implemented for debugging purposes and experimentation:
  + Fixed-Time Controller: Operates on a predefined cyclical schedule.
  + RL-Based Controller: Selects actions based on a trained reinforcement learning model.

The framework for my DRL model was taken from the PyTorch Docs [CITE HERE] and adapted to function in my chosen environment. Instances of the model were originally trained locally, but soon computational demand grew and training was offloaded to the universities GPU cluster for more efficient training. The model comprises of 3 hidden layers, each with 128 neurons per layer and a ReLu activation function.

[INSERT UML DIAGRAM HERE]

Traffic Signal Control Manner

As described in section [INSERT HERE], numerous steps were taken to provide the model with an action space small enough (36) to allow for rapid convergence on a policy whilst the mandatory 36 other yellow phases would be enforced by the environment wrapper itself. However often two successive phases will have green lights in common, switching these lights to yellow and back to green again would unnecessarily slow down cars and reduce junction throughput. Base-3 arithmetic was leveraged to efficiently represent over 443 unique phases and allow the environment wrapper to select the most efficient one according to the agents action.

**TODO: Expand on technical challenges encountered**

Linear and Discrete Optimization Implementation

**TODO**

**Lit Review;**

Need 20/30 papers, currently have about 16 RL based papers with. Other papers to study:

\* Find a paper on fixed time control

\* Max pressure control of a network of signalized intersections Pravin Varaiya

\* State-of-art review of traffic signal control methods: challenges and opportunities: extract a few papers from this

\* Traffic Lights Control Using Reinforcement Learning: A Comparative Study: extract a few papers from this – requested access, waiting on author response

\* Reinforcement learning based adaptive control method for traffic lights in intelligent transportation

\* Traffic Signal Control in an MPC Framework Using Mixed Integer Programming - M.A.S. Kamal∗ J. Imura∗∗ T. Hayaka

\* Mixed Integer Linear Programming for Traffic Signal Control - Iain Guilliard

\* An optimization modelling of coordinated traffic signal control based on the variational theory and its stochastic extension

\* Optimization of Traffic Signal Settings by Mixed-Integer Linear Programming: Part I: The Network Coordination Problem (and part 2, can cite separately I think😉)

\* Can cite all the algorithms used in my testing section: SOTL, Websters, MaxPressure, DQN, DDPG, Uniform

Paper Notes

Support choice of SUMO with literature citings, can cite:

State-of-art review of tsc methods: challenges and opportunities

Sumolights paper

In literature review can include an overview of statistics maybe: out of all papers mentioned, which used a microsimulator, what % worked on a single intersection, what % used real world data for evaluation, what % used fixed time phases

Daganzo, C. F., 1995. The cell transmission model, part ii: network traffic. Transportation Research Part B: Methodological, 29, 2 (1995), 79–93.

State-of-art review of traffic signal control methods: challenges and opportunities

77% of papers studied utilized a microsimulation tool in some form, showing the industry preference for this over a mathematical formulation

Traffic signals can be broken down into Fixed-Time, Adaptive and Dynamic

Simulators can be microscopic, macroscopic or mesoscopic (e.g. SUMO)

Reinforcement learning based adaptive control method for traffic lights in intelligent transportation

Collected their own traffic data: traffic flow, speed, driving trajectory, evaluated using SUMO

State repsented as traffic flow : a metric which combines traffic density and speed, considers pedestrians as well but no mention of EMVs

Actions: choosing phase and cycle length

Reward defined as cumulative waiting time between two cycles

Deployed with a continuous learning approach to consistently reduce queeing times for up to 6 months post deployment

Dynamic time phases – bounded between 0 and 60 seconds, configurable in 5 second intervals

An Optimization Modelling of coordinated traffic signal control based on the variational theory and its stochastic extension

Also modelled on just a simple 1 lane per approach formulation

Mixed integer linear programming – modelling networked intersections together

Employes a kinetic wave model to simulate traffic flow, cell transmission model has also been used – note this is for modelling across several intersections, I think we can say we simplified the CTM model

Acknowledges the high number of binary variables involved in solving such a problem

Constrained by having competing signals on at the same time and aims for minimal signal switching, no mention of yellow time constraint

Presents a formulation to minimize number of binary variables

Data simulated using a poisson distribution

Experimented with 3 different cycle time constraints:

Model can optimize all 🡨 this was most effective and also leaned towards shorter cycle times

Model is given a cycle length

Cycle length is calculated using websters formula

Discretized time intervals to a certain value

Traffic Signal Control in an MPC Framework Using Mixed Integer Programming

Mixed integer linear programming

This is basically the model I implemented – with a simplfieid CTM

Uses networked intersections

Not fully dynamic – uses constants to assume vehicle turning directions

Uses a simplified one lane per approach formulation

A cell-based distributed co-ordinated approach for network level signal timing optimization

Mixed integer lienar programming, cell transmission model

Networked intersections

Is it live?

Also formulated as a binary decision variable for green time

Maximizes intersection throughput

Uses constants to simulate vehicle turning directions

Also 1 lane per intersection

Green time bounded between constraints

Considers just a window of the problem at a time, decomposed into running several models (1 per intersection) simultaneously , reducing complexity. Acknowledges this leads to a locally optimal solution

Could have some kind of flow chart or walk through of the LO process?

Choose green signals

Update queue lengths – arrivals / remove vehicles from simulation

Until time runs out

Simulated on synthetic data : both symmetrical and asymmetrical high / low demand

Mixed Integer linear programming for traffic signal control

Considers the implementation of light-rail systems with vehcie traffic

Uses a microsimulator to compare it to fixed time control

Focused on scalability for large traffic networks – done by simplifying the CTM using a dsicretized time step

QTM:

Variable time steps

Any roadway without diverges can be modelled as a single queue <- THIS IS WHAT WE DID

Assumes linear vehicle arrival – huge flaw!

QTM enabled the model to solve both more efficiently

Uses a switching penalty

Objective function of minimizing delay

**Lit Review:**

**Intro**

Introduce various algorithms Fixed Time, Adaptive – Websters, Maxpresure etc..

This literature aims to provide a comprehensive account and comparison of the various approaches to formulating and solving the Traffic Signal Control problem.

TSC methods can be categorized into one of three main types:

Fixed Time Systems: These operate based on a cyclical structure with a fixed time period for each traffic phase.

Actuated / Dynamic Systems: These follow rule based conditions to dictate signal activation, such as extending green phases for a specific road during rush-hour.

Adaptive Systems: These will dynamically adjust the phase length and cycle pattern based on real-time traffic data, to optimize flow in response to current traffic conditions. Notable algorithms include Websters (Webster, 1958), MaxPressure (Varaiya, 2013) and SOTL (Gershenson, 2004).

More recently, research has been focused on leveraging computational methods in the problem context to develop adaptive systems by leveraging: heuristics, metaheuristics, artificial intelligence and mathematical optimization. This literature review will focus on the latter two.

**RL Section**

Compare framework used?

Reinforcement learning was first applied to the problem through Q-learning with the addition of genetic algorithms to efficiently search and update hyper-parameters across training. Despite being a valid first step in applying RL to the problem, the paper (Kakazu, 1994) rapidly encounters the curse of dimensionality due to the wide and complex state space of the problem. The model is only able to operate within a minimal simulation environment. (Prashanth L. A., 2011) addressed this with function approximation techniques employed to reduce state-space and negate the issues encountered by Kakazu et al. More details on this are discussed in section [INSERT HERE]

Modern research has focused on leveraging neural networks through DRL, in order to cope with the high-dmensional state space. Studies such as (Huang, 2024) (Tham, 2007) (Romain Ducrocq, 2021) (Mobin Yazdani, 2023) (Haoran Su, 2022) have demonstrated the superior results that can achieved through DRL models. The absence of a large Q-table noted in other studies makes the system more precise and scalable.

Compare Environment Used for Simulating and Intersections modelled

Urban simulators are an effective way to model vehicle flow and capture key complexities such as driver behaviour and vehicle acceleration. Using a sophisticated simulator will ensure more accurate adaptation to the models real-world deployment environment.

The simulation tools available fall into one of three categories:

* Microscopic simulators consider individual driver behaviour and their interactions with other road users.
* Macroscopic simulators abstract individual details and focus on vehicular flow as a whole, without modelling individual vehicle dynamics.
* Mesoscopic simulators are a hybrid of both approaches, and will model overall vehicle flow with some consideration to individual behaviour.

With the exception of (Kakazu, 1994) and (Tham, 2007), who developed their own macroscopic environment, the other papers considered here used third-party, microscopic simulators. The most popular simulator noted was SUMO (Lopez, 2018) used by 3 papers (Huang, 2024) (Romain Ducrocq, 2021) (Haoran Su, 2022), other simulation environments used were VISSIM (Martin Fellendorf, 2011) and GLD (M. Wiering, 2004). Another literature review (Syed Shah Sultan Mohiuddin Qadri, 2020) collated 57 papers and found that 33% of papers studied used VISSIM whilst 25% of papers used SUMO, this highlights the two software’s dominant superiority across the field of urban simulation.

One notable shortcoming across the literature is the lack of simulation parameters disclosed, meaning it is hard to replicate and verify results. Factors such as: max car speed, acceleration, road length, vehicle length and driver imperfection rating are often omitted, despite the sensitivity of final results in regard to these parameters.

Actors Involved

Simulation tools allow for the inclusion of various road actors. SUMO allows for the inclusion of Cars, Cyclists, Pedestrians, Emergency Vehicles and Public transport. Despite this, the majority of papers studied focus exclusively on cars. This shortcoming limits the real-world validity of the majority of studies and is a major barrier to deploying systems into the real-world.

One exception to this statement is EMVLight (Haoran Su, 2022). The paper proposes a framework for coupling efficient EMV routing with a RL-based traffic signal control agent. The model aims to reduce travel times for both civilian vehicles and emergency ones. Similarly, IVPL (Mobin Yazdani, 2023) extends the simulation environment to involve pedestrians. The work is based off a pre-existing model [cite here] but extended to penalize pedestrian delays. Real-world validity of this work is extended by simulating jay-walking and studying the effects this realistic factor has on model performance.

Despite the efforts of these two studies, to the authors knowledge, there are no available studies that focus on developing a RL agent that can effectively handle all actors it would encounter in a real world deployment. Further investigation into this area would not only reduce overall congestion but also potentially result in:

* Optimized emergency response times
* Increased walkability and pedestrian safety by ensuring fairness across signal timings

Network Used

The TSC problem extends to a wide variety of road configurations, but is most commonly studied in the context of one or more networked four-way intersection. This could be attributed to the complexity that makes solving such a problem highly desirable: incoming vehicles from all 4 directions, competing for control of one singular intersection.

A graph of a number of traffic models

AI-generated content may be incorrect.Whilst some studies ( (Huang, 2024), (Romain Ducrocq, 2021), (Mobin Yazdani, 2023)) focus on creating an agent that can effectively control just one of these intersections, others ( (Kakazu, 1994), (Prashanth L. A., 2011), (Tham, 2007), (Haoran Su, 2022)) proved the scalability of their models across multiple intersections networked together. Such a setup allows for experimentation on the results of increased environment context for the models. Shown right are the results from (Tham, 2007) who networked two RL-based agents together to allow for sharing of information. This model (case III) was compared to two solitary models (Case II) and a fixed time model. As shown by the graph, allowing the networked agents to communicate environmental data between one another resulted in significantly reduced average delay within the simulation.

Compare State

As outlined in section [INSERT HERE], solving problems such as TSC requires them to be formulated as an MDP. Given the problems infinite, continuous state space; efficient representation of the current environment state will enable a model to effectively converge on an optimal policy.

Common state factors observed across the literature are:

* Current phase – a vector representation of lights that are currently green
* Vehicles in lane – the number of vehicles currently on a lane
* Queue lengths – the number of vehicles queueing on a lane, a vehicle can be defined as queueing if its current speed is 0m/s (stationary).

As noted earlier, Q-learning based algorithms are hard to apply to large problems with complex state spaces, due to the curse of dimensionality. However, (Prashanth L. A., 2011) applied function approximation techniques to categorize the continuous state space within discrete categories. Reducing the size of the state / action pair space from 10101 to just 200. However, this approach proved to be unscalable to larger intersections due to the state space still growing exponentially.

Networked, multi-intersection agents naturally extend their state space with information from other intersections. (Tham, 2007) state information would also include the action and Q-value obtained from any downstream agents, so that upstream agents could anticipate incoming traffic.

State space size also grows with actors present in the simulation, as there is more information to represent. (Mobin Yazdani, 2023) algorithm was focused on the inclusion of pedestrians, and state information would include pedestrian numbers on each crossing. Meanwhile, (Haoran Su, 2022) included comprehensive information on emergency vehicles: the distance of the EMV to the intersection, the ETA of the EMV at the intersection, and the EMVs proposed next step.

By extending state representation to allow for multi-agent co-ordination and varied road users, research is closing the gap between real-world and simulation. However, the computational resources required to develop a comprehensive policy grows exponentially with the state space, and this remains an open challenge in DRL-based TSC.

Compare Action

All RL-based algorithms studied have their action focused on picking the next, most-optimal traffic phase. Careful design is needed in order to design an action space large enough that it can comprehensively respond to all possible states, whilst remaining small enough that training the model is computationally efficient.

The majority of papers noted in this study will choose phases in an acyclic manner. (Hua Wei, 2018) instead opted for a cyclical structure and a binary action space {0,1} where 0 opts to stay in the current phase and 1 will transition to the next. Whilst this policy may be efficient to train to, it gives a much more limited flexibility in comparison to some of the other studies.

Some papers (Haoran Su, 2022) (Romain Ducrocq, 2021) opt to provide the model with a pre-configured set of phases, where the action space size reflects the different phases available to the model, which will output a score for each, indicating the suitability.

Other papers (Tham, 2007) (Huang, 2024) have taken a more flexible approach, allowing the model to select both the next active phase, and its duration, Tham et al’s action space was a continuous one. 0..n , where n is the number of phases available to the model, which will output a continuous (bounded) value for each indicating the green time it should be allocated. As noted by the author of this paper, It is important to consider the negative effects of such a large action space, even on a simple 2-light intersection, indicating this approach may not viably scale to larger scenarios.

(Mobin Yazdani, 2023) uses a hybrid approach to iteratively clear pedestrians, and allocate a constant amount of green time to each car lane. If there are still vehicles remaining in the simulation the RL-agent will allocate additional green time to each lane according to the policy, if there are no vehicles present the cycle repeats. Whilst this approach can ensure fairness across both groups, it does not allow for pedestrian lights and vehicle lights to be green at the same time, even if the two routes don’t co-inflict.

Compare Reward

Effective reward functions allow for dictation of the models priorities and can differ from problem to problem.

Some reward functions in the literature focus on minimizing cumulative user delay, this correlates well with the real world objective of minimzing user travel times, and can also be exponentially weighted to penalize longer wait times. However, it also requires constant vehicle tracking, something much easier done in a simulation environment than real world.

[INSERT EQUATION HERE]

Another common function is to maximize junction throughput, this reward style will encourage smooth traffic flow but may result in unfairness to certain road users, as there is no incentive for the model to regulate wait times.

[INSERT EQUATION HERE]

One intuitive study (Hua Wei, 2019) acknowledges that the primary objective: reducing travel time, is a culmination of micro factors. This model instead implements a reward focused on minimizing intersection pressure: a metric defined by the adaptive algorithm MaxPressure. The introduction of RL-based methods to the MaxPressure algorithm proved more effective than the standalone MaxPressure algorithm.

[INSERT EQUATION HERE]

Compare Testing / Training Data used

Due to there being no known optimal solution for the TSC problem, evaluation is usually done through comparison of different models. A significant challenge in regards to this is the low-availability of open source, consistent datasets, meaning most studies use synthetic data for the training and evaluation. Arrivals are often simulated with a Poisson or Gaussian distribution to mimic arrival patterns.

Certain papers (Mobin Yazdani, 2023) (Huang, 2024) have used real-world datasets to evaluate their models. However, it is observed in an alternative literature review (Syed Shah Sultan Mohiuddin Qadri, 2020) that 42.2% of studies would use generated data for evaluation. This once again, raises questions about the real-world validity of the models.

From the literature review it is clear that the lack of real-world data is a serious barrier to standardization and real-world deployment of models. The lack of datasets could possibly be attributed to privacy concerns, or practical difficulties in collecting such a comprehensive dataset. Nevertheless, work focused on capturing and processing detailed traffic data would be a valuable contribution to the field.

Compare Results – Evaluation metrics used, what proved to be effective?

From the literature, it is appropriate to say that RL-based TSC algorithms are capable of outperforming non-RL adaptive ones and fixed time algorithms in a variety of metrics including: junction throughput, vehicle delay and overall travel time. Studies that included pedestrians (Mobin Yazdani, 2023) and EMVs (Haoran Su, 2022) were also able to extend fairness to these actors.

Interestingly, there are conflicting results across the literature about including information on the wider environment for the model. (Tham, 2007) observed significantly improved results from networking two intersections together and (Haoran Su, 2022) successfully routed EMVs by predicting their arrival times. In contrast, other papers [cite here from interim report] were able to outperform various benchmark models through minimalist state design, leading to reduced model training times and costs.

Conclusion on RL based methods

[insert interim report table, here? Or elsewhere]

The literature shows extreme potential for the application of RL paradigms in the context of TSC and DRL specifically as a promising approach for handling such a complex problem. However, it also highlights the significant gap between real-world and simulation. The majority of papers cited only consider cars, meaning it is impossible to evaluate how the models will perform when deployed and encounter EMVs or pedestrians. Although certain papers have set out to include one or the other, to the author's knowledge, there is no model capable of effectively handling all 3.

Furthermore, there is a lack of standardization across the field, models are often designed and trained across varying simulation environments and road networks. Detailed simulation parameters are often not included, making it hard to reproduce results.

From this review, it is clear that future work should be focused on:

* Producing real-world datasets for standardized evaluation across models,
* Developing systems capable of optimizing: vehicle, pedestrian and EMV travel times
* Conducting research into the effects of supplying increased environment data to the model.

TODO:

Go into more information on intro / adaptive methods in intro. Expand on their limitations with proof

**LO Section**

Recent work has also focused on using linear optimization methods to solve the problem. These models leverage mathematical programming techniques to determine (near) optimal traffic settings. A systematic review of the remaining literature highlights two flaws also commonly seen within RL based studies: a lack of real-world validity and a high computational cost associated with running these models.

Despite these flaws, LO-based methods are able to provide a near globally optimal solution, meaning they are able to provide a strong baseline for comparison against RL-based approaches and serve as a highly valuable area of future research.

Intro – talk about usage of binary variables and problem size

Out of the 4 papers (Kentaro Wada, 2017) (M.A.S. Kamal, 2013) (S.M.A. Bin Al Islam, 2017) (Guilliard, 2020) noted in this study, all formulate the key decision variable as an array arr[S][T] ex {0,1}. Where:

* S represents a specific traffic signal present in the network
* T represents a discrete time step
* Arr[S][T] ex {0,1} indicates whether signal S is green at time T.

This binary formulation offers a flexible and effective representation of the problem as a Mixed\_integer Linear Programme (MILP). However, the large number of binary variables does not optimize well, particularly when using the branch-and-bound method used by LO solvers today.

Compare intersections and actors modelled – network or singular, mention how all only include cars

Unlike the microscopic approach noted amongst the RL based studies, all LO models took a macroscopic approach to modelling the intersections, this can be attributed to the challenges in representing microscopic factors such as individual driver behaviour mathematically and the inability to leverage simulators like SUMO when solving the problem in this manner.

Similar to the RL-based papers studied, the literature in this field covers both singular and networked intersections. However, all intersections modelled represented a simplistic layout, with one lane per incoming direction. This reduces the problem size and simplifies constraints in that only one light can be active at a time. There remains an important direction for future research into models of more complex intersections, which would theoretically allow for multiple lights to be safely green at the same time.

Furthermore, the vast majority of LO studies focus exclusively on vehicle traffic, mirroring a similar limitation seen in RL studies. Amongst the four papers reviewed, three exclusively considered cars, while only one paper (Guilliard, 2020) had the additional actor of light-rail systems. This study was able to successfully model constraints surrounding this and optimize the phases in regards to both traffic flow and rail systems simultaneously. The bias towards cars in the literature is evident. The lack of extraneous road actors considered, and the omittance of microscopic factors make the results from these models mere approximations,rather than fully developed solutions that would be expected to work in a real world environment

Compare representation of queue transmission mode – QTM / CTM / KW

A fundamental of modelling the TSC is the equation used to model vehicle flow across a road network and the buildup of traffic queues. This area of the problem will be subsequently referred to as the queue evolution strategy. One of the first such strategies was initially proposed by (Miller, 1963). This is called the Queue Transmission Model (QTM). Miller states that the queue for one approach at a single intersection, at a given time step t can be modelled as [INSERT EQUATION HERE]

Across the modern literature there were 3 primary approaches observed that aim to utilize and build off of Miller’s work:

* CTM (Cell transmission model) – Initially proposed by (Daganzo, 1995) and building directly upon the work of a QTM proposed by (Miller, 1963) . This was incorporated into models by (M.A.S. Kamal, 2013) (S.M.A. Bin Al Islam, 2017). Daganzo broke each road segment into discretized distances of a fixed length, a section is denoted by its origin (o) and destination (d) as (od). The continuous variable vod denotes the number of vehicles on the section (od) at time t. Therefore the queue update equation can be modelled as:



Where pout and pin are the inflow and outflow traffic of a section respectively. Whilst the extension of the CTM upon the QTM gives the model a finer representation of vehicle positions within the network, and partially satisfies the lack of microscopic representations noted previously, it also extends the problem space by representing each road as a series of segments rather one as one large one. With that said, employing a CTM is a necessity for modelling multiple intersections networked together, as these two papers did.

* Extended CTM – Building upon the work of (Daganzo, 1995) was (Guilliard, 2020). Guilliard correctly stated that a CTM is only advantageous (and necessary) over a QTM when a roadway does not diverge or merge into another. Guilliard then employed a hybrid approach for the complex road networks modelled, only using a CTM when strictly necessary. Further optimisations employed by Guilliard are discussed in section [INSERT HERE]. Although as a result of this extended CTM he was able to scale the model to effectively handle a 3x3 grid of intersections and still achieve optimal results, more than any other paper considered in this study.
* Kinetic Wave Model– proposed by (Daganzo, 2005) and utilized by (Kentaro Wada, 2017) states that traffic flow can be characterized by: ur (forwardwavespeed),wr (backwardwavespeed), qr max (saturationflowrate),andκr j (jamdensity). Unlike the CTM, this model captures the stochasticity of traffic flow, a valuable microscopic factor that most models do not. However, implementation of this equation with the model also results in more binary models than the previous two queue evolution strategies.

Compare objective function

A variety of objective functions were noted across the literature, perhaps the most sophisticated being presented by (Guilliard, 2020) who leveraged the extended CTM to calculate exact travel times for vehicles in the simulation. Other objective functions would focus on minimizing total traffic in the network at any given time (M.A.S. Kamal, 2013) or similarly, maximizing junction throughput (S.M.A. Bin Al Islam, 2017).

Compare constraints and constraints omitted

There were also multiple similarities noted across the literature in terms of the problem constraints noted. Multiple papers (S.M.A. Bin Al Islam, 2017) (Guilliard, 2020) (Kentaro Wada, 2017) imposed constraints on both the minimum and maximum green time. The respective authors cited various reasons such as: mimicking pedestrian clearance, safety and realism.

Furthermore, all papers modelled cars travelling through the network by using a constant saturation rate parameter. Formulated as: [insert equation here: dcell,time < lambda. Where lambda is the maximum number of vehicles that can travel across a cell per time step.

One notable shortcoming of all papers cited above is the lack of explicit mention of yellow signals within the problem. Yellow signals will cause lower throughput than a green signal due to the necessary acceleration/deceleration of cars from the preceding or succeeding red phase. This omission could be attributed to the fact that the inclusion of yellow signals would introduce another set of binary decision variables. Instead, researchers modelled this lost start-up time in various ways: (S.M.A. Bin Al Islam, 2017) introduced a constraint to reduce the previously mentioned saturation rate parameter at the initiation of a green signal, to account for the lost start-up time. Meanwhile, (Guilliard, 2020) (Kentaro Wada, 2017) both introduced a penalty for phase switching in the objective function.

Compare optimization attempts

A consistent theme throughout the literature is the high number of binary variables involved in representing the problem, despite optimization attempts such as the extended CTM. Other successful optimization attempts include splitting time up into discrete intervals. This was an optimization proposed by (Kentaro Wada, 2017) where instead of a decision variable being represented as an array of time \* lane size it is instead represented by an array of size (time / lambda) \* lane, where lambda is the discretized time step value. Although being an efficient reduction of problem size it is important to note the loss of fine-grained control with such an approach. This is due to all arrival departure information from time steps n to n + lambda being collated within one array cell.

Alongside the extended QTM model, (Guilliard, 2020) laid out further optimizations. By considering just a small problem window at the same time to drastically reduce the problem space. Furthermore, when running the model on networked intersections, the model was run on subsets of the network in parallel. Although successful in reducing the computational cost of solving the issue, it was acknowledged by the researcher that this approach would obviously lead to producing only an optimal solution to the problem.

Compare evaluation – datasets used, results etc.

Across the literature there are notable flaws in the data used to evaluate model results. No papers studied made use of real-world data and instead opted to modelling traffic arrivals assuming either a linear (Guilliard, 2020) (S.M.A. Bin Al Islam, 2017) (M.A.S. Kamal, 2013) or Poisson (Kentaro Wada, 2017) distribution. Theses distribution methods do not effectively capture the stochastic and variable demand that modern RTNs encounter. Future studies should look to include real world data when evaluating their models.

All studies reported levels of success in developing a model that can find (near) optimal solutions to the TSC problem; however, all studies also cited the computational times as being the key barrier to deploying these solutions to the real world. One interesting set of results comes from (Kentaro Wada, 2017), who experimented with the level of control the model had over signal green times. Three different scenarios were tested: model has full control over a phases green time, green time is constrained to an exact integer value, green time is calculated using Webster’s formula (Webster, 1958). It was recorded that giving the model full control over green time gave the most optimal solution.

LO Conclusion

The literature review has shown that studies often take a similar approach to problem representation. Common themes include modelling the environment macroscopically and utilizing binary decision variables to represent signal status. Whilst having been proved through successful results as a correct representation of the problem, the large problem space and high occurrence of binary values leads to a high computational cost being associated with the problem. Although certain optimization attempts have been made and were successful in reducing the computational complexity, reductions were not enough to allow for modelling of additional microscopic constraints.

The omittance of modelling microscopic factors and the lack of real-world data used in previous studies means that models are not an accurate reflection of real world studies seen today.

FULL Lit review conclusion

# Bibliography

Daganzo, C., 1995. The cell transmission model, part ii: Network Traffic. *Transportation Research Part B: Methodological,* Volume 29, pp. 79-93.

Daganzo, C. F., 2005. A variational formulation of kinematic waves: basic theory and complex boundary conditions. *Transportation Research Part B,* 39(2), pp. 187-196.

Gershenson, C., 2004. *Self-Organizing Traffic Lights.* s.l.:arxiv.

Guilliard, I., 2020. *Mixed Integer Linear Programming for Traffic Signal Control.* s.l.:s.n.

Haoran Su, Y. D. Z. J. Y. C. B. D. L. J., 2022. EMVLight: a Multi-agent Reinforcement Learning. *Transportation Research,* Volume 3.

Hua Wei, C. C. G. Z. K. W. V. G. K. X. Z. L., 2019. *PressLight: Learning Max Pressure Control to Coordinate Traffic.* s.l.:Proceedings of the 25th ACM SIGKDD Internatinoal Conference on Knowledge Discovery & Data Mining.

Hua Wei, G. Z. H. Y. Z. L., 2018. IntelliLight: A Reinforcement Learning Approach for Intelligent Traffic Light Control. *ACM SIGKDD International Conference,* Volume 24.

Huang, Z., 2024. Reinforcement learning based adaptive control method for traffic lights in intelligent transportation. *Alexandria Engineering Journal,* Volume 106, pp. 381-391.

Kakazu, S. M. a. Y., 1994. Genetic reinforcement learning for cooperative traffic signal control. *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence,* 1(10), pp. 223-228.

Kentaro Wada, K. U. T. T. M. K., 2017. An optimization modeling of coordinated traffic signal control based on the variational theory and its stochastic extension. *Transportation Research Procedia,* 23(2352-1465), pp. 624-644.

Lopez, P. A., 2018. Microscopic Traffic Simulation using SUMO. *International Conference on Intelligent Transportation Systems,* pp. 2572-2582.

M. Wiering, J. V. J. v. V. a. A. K., 2004. Simulation and Optimization of Traffic in a City. *IEEE Intelligent Vehicles Symposium,* pp. 453-458.

M.A.S. Kamal, J. I. T. H. A. O. K. A., 2013. Traffic Signal Control in an MPC Framework Using Mixed Integer Programming. *IFAC Proceedings Volumes,* 46(21), pp. 645-659.

Martin Fellendorf, P. V., 2011. Microscopic traffic flow simulator VISSIM. *Fundamentals of Traffic Simulation.*

Miller, A. J., 1963. Settings for Fixed-Cycle Traffic Signals. *Journal of the Operational Research Society,* Volume 14, pp. 373-386.

Mobin Yazdani, M. S. S. A. B. N. N. J. P. H. P., 2023. Intelligent vehicle pedestrian light (IVPL): A deep reinforcement learning approach for traffic signal control. *Transportation Research Part C: Emerging Technologies,* Volume 149.

Prashanth L. A., S. B., 2011. Reinforcement Learning with Function Approximation for Traffic Signal Control. *IEEE Transactions on Intelligent Transporation Systems,* 12(2), pp. 412-421.

Romain Ducrocq, N. F., 2021. *Deep Reinforcement Q-Learning for Intelligent Traffic Signal Control with Partial Detection.* s.l.:arxiv.

S.M.A. Bin Al Islam, A. H., 2017. Distributed coordinated signal timing optimization in connected transportation networks. *Transportation Research Part C: Emerging Technologies,* Volume 80, pp. 272-285.

Syed Shah Sultan Mohiuddin Qadri, M. A. G. &. E. Ö., 2020. State-of-art review of traffic signal control methods: challenges and opportunities. *European Transport Research Review,* Volume 12.

Tham, S. S. a. C. -K., 2007. SensorGrid for Real-Time Traffic Management. *International Conference on Intelligent Sensors, Sensor Networks and Information,* Volume 3, pp. 443-448.

Varaiya, P., 2013. Max pressure control of a network of signalized intersections. *Transportation Research Part C: Emerging Technologies,* Volume 36, pp. 177-195.

Webster, 1958. Traffic Signal Settings.