Precise Design of Research

Date: 11/05/2025
Week: 10
Author: Alvin B. Lin
Student ID: 112652040

Problem 1

Overview and Motivation

Urban traffic congestion remains a persistent challenge in modern cities, directly impacting economic productivity, energy efficiency, and air quality. Traditional traffic light systems are often static or heuristically adaptive, adjusting signal phases based on fixed rules or simple sensor thresholds. However, these approaches fail to react intelligently to dynamic and stochastic events such as sudden congestion, accidents, or weather disruptions.

In this work, we propose a **mathematical and learning-based model** for adaptive traffic light control. The system aims to minimise total congestion and delay by optimising traffic signal states in real time, guided by observed car flow and incident probabilities.

The model combines:

- Continuous traffic flow modelling (via partial differential equations),
- Probabilistic jam detection, and
- Reinforcement learning for decision-making under uncertainty.

Mathematical Formulation

Road Network Representation

Let the urban network be modelled as a directed graph

$$G = (V, E),$$

where $V = \{1, 2, \ldots, N\}$ is the set of intersections, and $E \subseteq V \times V$ is the set of road segments (directed edges).

Each intersection $i \in V$ controls a traffic signal that regulates the flow between its incoming and outgoing roads. For concreteness, consider the **Tian**-shaped road layout (a 3×3 grid), where $N = 9$ and each intersection connects up to four directions.

Traffic Signal Control Variables

For each intersection $i \in V$, define the signal state at time t as

$$s_i(t) = \begin{cases} 1, & \text{if east–west traffic is allowed (green),} \\ 0, & \text{if north–south traffic is allowed (green).} \end{cases}$$

Since conflicting flows cannot proceed simultaneously, the perpendicular state satisfies

$$s_i(t) + s_i^{\perp}(t) = 1.$$

Each signal operates with a cycle time $T_i(t)$ and a phase offset $\phi_i(t)$, producing a square-wave type function

$$s_i(t) = \text{square}(t; T_i, \phi_i),$$

which can adapt dynamically over time.

Traffic Flow Dynamics

Each road segment $e = (i, j) \in E$ is parameterised by position $x \in [0, L_{ij}]$. Let

$$\begin{aligned} p_e(x,t) &: \text{car density (vehicles per unit length)},\\ v_e(x,t) &: \text{average velocity},\\ f_e(x,t) = p_e(x,t)v_e(x,t) &: \text{traffic flux},\\ q_e(x,t,p) &: \text{probability density of congestion or accident}. \end{aligned}$$

The fundamental relationship between these quantities follows the Lighthill–Whitham–Richards (LWR) conservation law:

$$\frac{\partial p_e}{\partial t} + \frac{\partial f_e}{\partial x} = 0,$$

with a nonlinear flux function

$$f_e = v_{\max} p_e \left(1 - \frac{p_e}{p_{\max}}\right),$$

to capture the decline in velocity as roads become congested.

Intersection Boundary Conditions

At intersections, inflows and outflows are determined by the current signal states. Let $C_{i\to j}(t)$ be the effective capacity coefficient (vehicles per unit time) for the connection from i to j. Then

$$f_{i\to j}(t) = s_i(t)\, C_{i\to j}(t)\, p_i(t),$$

so when the signal is red ($s_i = 0$), flow ceases.

The mass balance at intersection i is

$$\sum_{j:(j,i)\in E} f_{j\to i}(t) = \sum_{k:(i,k)\in E} f_{i\to k}(t) + \frac{dp_i^{\text{wait}}(t)}{dt},$$

where $p_i^{\text{wait}}(t)$ represents vehicles waiting at red lights.

Accident and Jam Influence

The presence of an incident locally reduces effective capacity. We model this via

$$C_{i\to j}(t) = C^0_{i\to j}\big(1 - \alpha\, q_{i\to j}(t,p)\big),$$

where C^0 is the nominal capacity and $\alpha \in [0, 1]$ quantifies the sensitivity to congestion.

The incident probability density $q(x, t, p)$ may be estimated from sources such as Google Maps jam data, historical accident rates, or sensor inputs. Note that q is meaningful primarily near the roads, i.e. for positions x within the road network.

Optimisation Objective

The overall goal is to minimise congestion, waiting time, and disruption from incidents. Define the performance functional

$$J = \int_0^T \int_\Omega \Big(p(x,t) + \lambda_1 p_i^{\text{wait}}(t) + \lambda_2 q(x,t,p) \Big)\, dx\, dt,$$

where the λ's are penalty weights.

We then solve

$$\min_{\{s_i(t)\}} J \quad \text{subject to traffic flow PDEs, capacity constraints, and signal switching rules.}$$

This represents a PDE-constrained optimisation problem, balancing flow smoothness with practical signal constraints.

Reinforcement Learning Formulation

Because the above optimisation is computationally demanding and the environment is stochastic, a reinforcement learning (RL) formulation is natural.

State Space At time t, the observed state is

$$o_t = \big[p(x,t), v(x,t), q(x,t,p), w(x,t), t_d, w_t, e_t \big],$$

where:

- $w(x,t)$: average waiting time,
- t_d: time of day,
- w_t: weather condition,
- e_t: special event indicator.

Action Space The control action is the adjustment of light timing parameters:

$$a_t = \{T_i(t), \phi_i(t)\}_{i \in V},$$

or equivalently, the binary signal states $s_i(t)$.

Transition Dynamics The environment evolves according to the traffic PDEs and external disturbances:

$$s_{t+1} = f(s_t, a_t, q_t, \xi_t),$$

where ξ_t represents random factors such as driver behaviour or emergency incidents.

Reward Function Define instantaneous reward:

$$r_t = -\sum_i \Big(p_i^{\text{wait}}(t) + \lambda_1 q_i(t) \Big),$$

so that minimising waiting time and congestion maximises reward.

The learning objective is

$$\max_\theta \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r_t \right],$$

where $\pi_\theta(a_t \mid o_t)$ is the controller's policy.

Graph-Based Coordination

To coordinate neighbouring intersections efficiently, we employ a Graph Neural Network (GNN) architecture. Each node i maintains a hidden representation $h_i^{(t)}$, updated as

$$h_i^{(t+1)} = \mathrm{Update}\left(h_i^{(t)}, \sum_{j \in \mathcal{N}(i)} \mathrm{Msg}(h_j^{(t)}, s_j(t))\right),$$

where $\mathcal{N}(i)$ is the set of neighbouring intersections.

This structure naturally captures spatial dependencies: a change in one signal affects flows on adjacent roads, and the GNN learns those relationships directly.

Implementation and Simulation Framework

For simulation, the `SUMO` (Simulation of Urban Mobility) platform can generate realistic traffic patterns. The controller receives real-time data on:

- vehicle counts and speeds from simulated sensors,
- congestion or incident estimates $q(x, t, p)$,
- environmental context (weather, time, events).

A reinforcement learning loop proceeds as:

Algorithm 1

```
Require: Number of episodes $N$, episode duration $T$, policy $\pi_\theta$, environment simulator
  for episode = 1 to $N$ do
      Initialise traffic environment (reset densities $p(x, 0)$, lights $s_i(0)$, etc.)
      for $t = 0$ to $T$ do
          Observe current state $o_t$ (e.g., densities, waiting times, incident probabilities)
          Select action $a_t = \pi_\theta(o_t)$                ▷ Determine signal durations or phases
          Apply $a_t$ to update traffic signals
          Simulate traffic flow for $\Delta t$ time steps using PDE or microscopic model
          Observe next state $o_{t+1}$ and compute reward $r_t$
          Update policy parameters $\theta$ via gradient-based RL (policy gradient or Q-learning)
      end for
  end for
  return Trained policy $\pi_\theta^*$
```

Extensions and Refinements

1. **Hierarchical Control:** Divide the city into regions, each with a local RL agent, coordinated by a global supervisor.
2. **Stochastic Events:** Introduce random accident events based on $q(x, t, p)$ to train robustness.

3. **Green Wave Optimisation:** Include phase coordination (ϕ_i) in the optimisation to minimise stop–start motion.

4. **Adaptive Reward Shaping:** Dynamically adjust the reward weights (λ_1, λ_2) depending on time-of-day priorities.

5. **Fairness:** Add a term penalising large variance in waiting times:

$$J_{\text{fair}} = \text{Var}_i\big(p_i^{\text{wait}}(t)\big).$$

Discussion and Broader Implications

This framework generalises beyond road intersections. Similar ideas apply to:

- **Railway scheduling**, where trains must share limited track segments and rescheduling is needed after delays or emergencies.
- **Air traffic routing**, managing dynamic capacity under weather uncertainty.
- **Autonomous vehicle coordination**, where each vehicle's control acts as a node in a decentralised network.

The same underlying principles—probabilistic flow estimation, graph-structured coordination, and reinforcement optimisation—can adapt to all these domains.

Moreover, integrating real-world data (e.g. Google Maps traffic intensity, sensor feeds, and historical jam logs) makes the system responsive to live conditions. As the system learns, it can progressively infer which intersections are bottlenecks and dynamically balance flow throughout the network.

Summary of Key Components

Category	**Symbol / Concept**	**Description**
Flow	$p(x,t), v(x,t), f(x,t)$	Density, velocity, flux
Signal Control	$s_i(t), T_i(t), \phi_i(t)$	Binary signal, period, phase
Incidents	$q(x,t,p)$	Probability of congestion or accident
Waiting	$p_i^{\text{wait}}(t)$	Queue density at intersection
Network	$G(V,E)$	Graph of intersections and roads
Objective	J	Integrated congestion and delay cost
Learning	$\pi_\theta(a_t \mid o_t), r_t$	Policy and reward for RL
Architecture	GNN	Spatial coordination between intersections

Concluding Remarks

This formalisation represents a unified approach to adaptive, intelligent traffic light control. It merges physically interpretable traffic flow models with data-driven learning, capturing both macroscopic dynamics and local uncertainties.

By combining probabilistic incident modelling, PDE-constrained flow, and graph-based reinforcement learning, the system can react to complex, evolving traffic environments. Such an approach moves beyond fixed-schedule traffic lights—towards **self-learning, decentralised control systems** that continuously improve as they interact with real-world data.