

An aerial, long-exposure photograph of a complex highway interchange. The image shows multiple levels of overpasses and ramps. Numerous cars are visible, their lights creating long, horizontal streaks of white, red, and blue, indicating the flow of traffic. The scene is captured from a high angle, showing the intricate geometry of the road network.

Optimizing Traffic Control using Bayesian Inference

Wenyu Li, Xiangyu Zhao, Yilin Sun

Structure of Presentation

- Problem Description
- Simulation
- Emulation
- Evaluation

A 3D perspective view of a city model with various colored lines and points overlaid. A prominent red line runs diagonally across the center, flanked by yellow lines. Blue lines form a grid-like pattern, and green lines are visible in the lower right. Several red and yellow points are scattered along these lines, with some having vertical stems. The text "Problem Description" is centered over the red line.

Problem Description

Motivations

- Impact on city planning: suboptimal traffic control leads to higher carbon emissions and waste of time for citizens.
- Adaptable for internet traffic: traffic planning can also be adapted in networking and communication for shorter latency, higher throughput and lower packet drop rate.
- Feasibility: computer modeling is widely used in traffic simulation and optimization.
- This project is inspired by the Google Hash Code 2021 Qualification Round

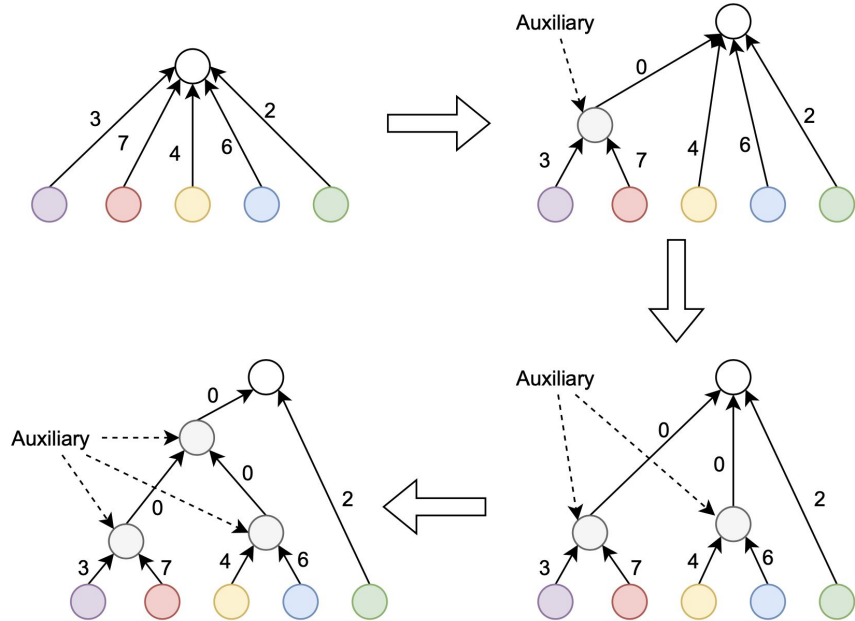
Problem description: network

Network: directed graph

Junctions: vertices

Roads: edges

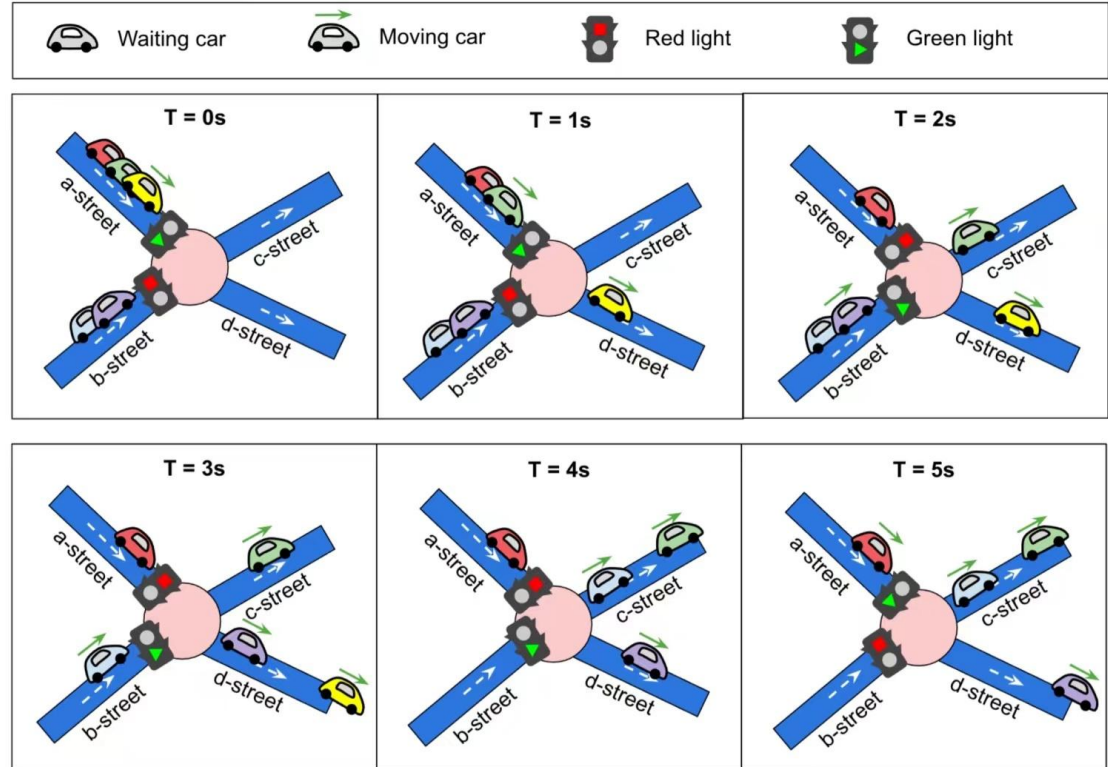
Assumption: Each junction has exactly 2 incoming roads.



Transformation algorithm

Problem description: traffic light and cars

- Traffic lights are placed at the end of each road.
- For each junction, at any point, exactly one of the light is green.
- During each time step, exactly one car passes the junction.
- Cars have uniform speed of 1
- Passing junction generates no delay.



Problem description: reward function

Reward is defined as the total distance travelled by all cars in the system

$$\text{Reward} = \sum_{c_i \in \text{Cars}} \text{Distance travelled by car } c_i$$

In Google Hash Code, original reward for each car is

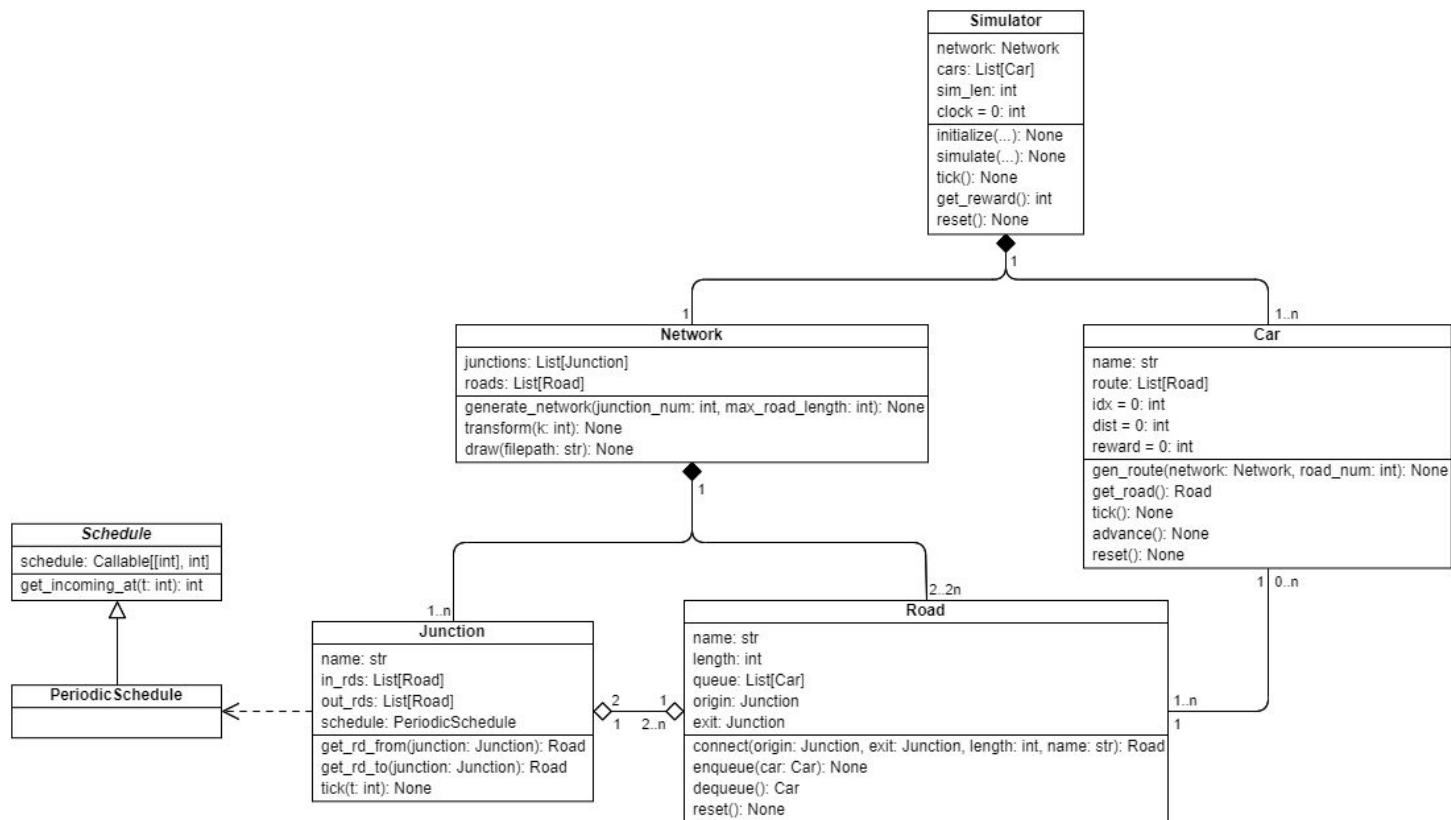
$$\text{Score for car } i = F + (D - T) \text{ if finishes else } 0$$

Goal: To maximize the reward function in a given time period T

Simulation

The image features a digital cityscape simulation. A glowing blue skyline of various skyscrapers and buildings is positioned in the background. The foreground consists of a dark green floor with numerous perspective lines that converge towards the horizon, creating a sense of depth. The word "Simulation" is centered in the middle of the image in a white, sans-serif font. The overall aesthetic is futuristic and technological.

Simulator structure



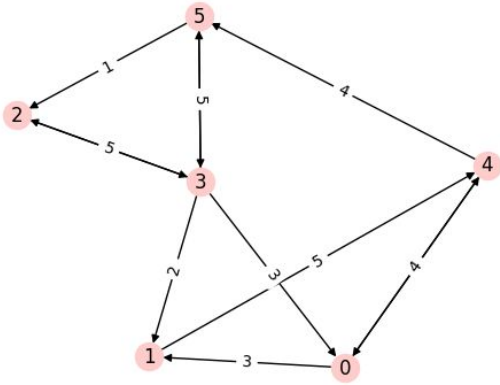
Network initialisation

Problems of most naïve random network generation algorithms:

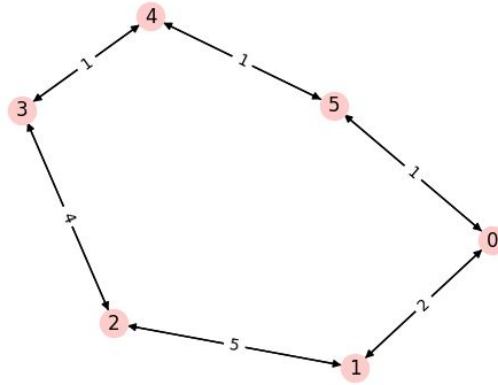
- Deadends: junctions with no outgoing roads
- Disjoint clustering: network becoming multiple disjoint clusters

Network initialisation

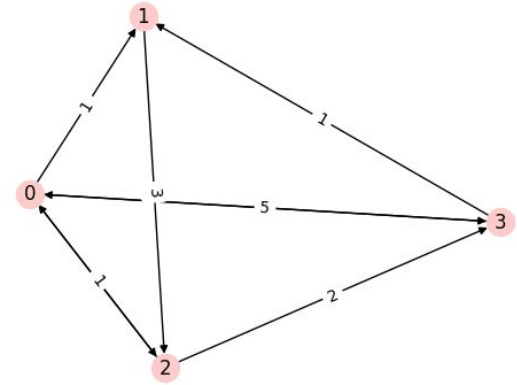
- Random network



- Ring network



- Text input



Simulation mechanism

Ticking: take turns to simulate every junction and every car at every time frame

- Junction: picks a road whose traffic light is green, and dequeue a car from that road (if any)
- Car: travels a distance of 1 along a road, or pushed into the queue at the end of the road



The background of the image is a complex financial visualization. It features a world map composed of blue dots, overlaid with various financial data series. There are multiple candlestick charts in orange and red, showing price movements. A prominent orange line graph with several peaks and valleys is visible. A numerical value '+11,000.00' is displayed in orange text. The overall color palette is dominated by dark blues, oranges, and reds, creating a high-tech, data-driven aesthetic.

Emulation

Methods and Techniques

- For our emulator, we used Gaussian Process and Bayesian Optimization
- Rationale behind the choice:
 - Traffic Simulation is costly => minimize number of evaluations
 - Traffic Simulation analogous to a black box, very limited understanding
- The Bayesian Optimization library of choice is GPyOpt
- Pipeline:
 - Scheduler → Simulator → Bayesian Optimizer

Models and Parameters

In addition to our basic model as mentioned in the simulator, we also added certain simplification of assumptions to facilitate the efficient emulation.

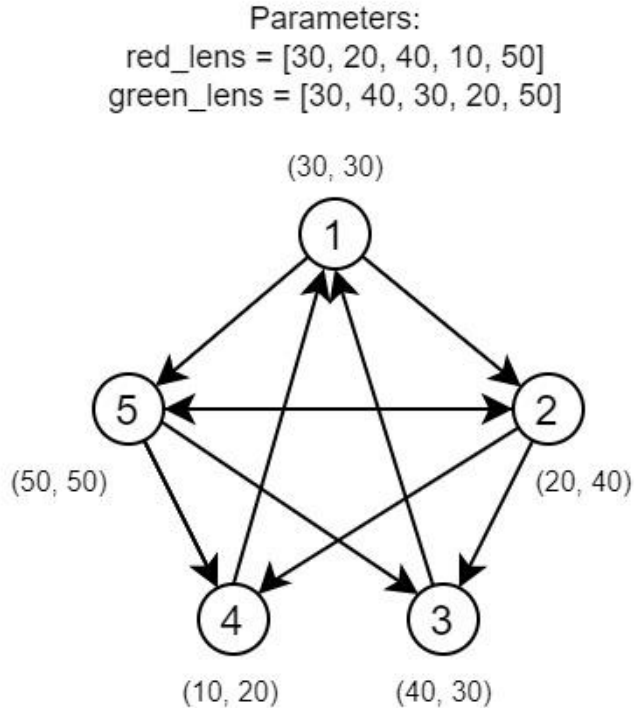
We started off by considering two basic cases:

1. Distinct Scheduling
2. Uniform Scheduling

We then move on to the hybrid of the two:

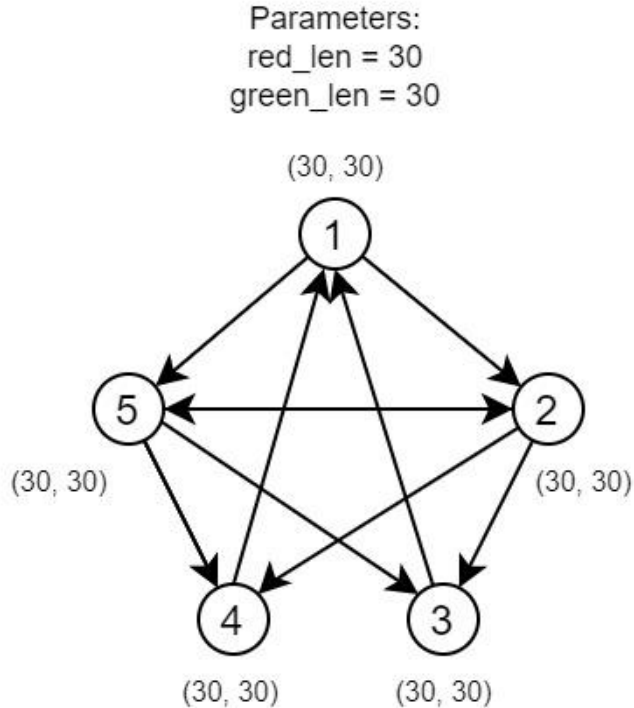
3. Preset scheduling

Distinct Scheduling



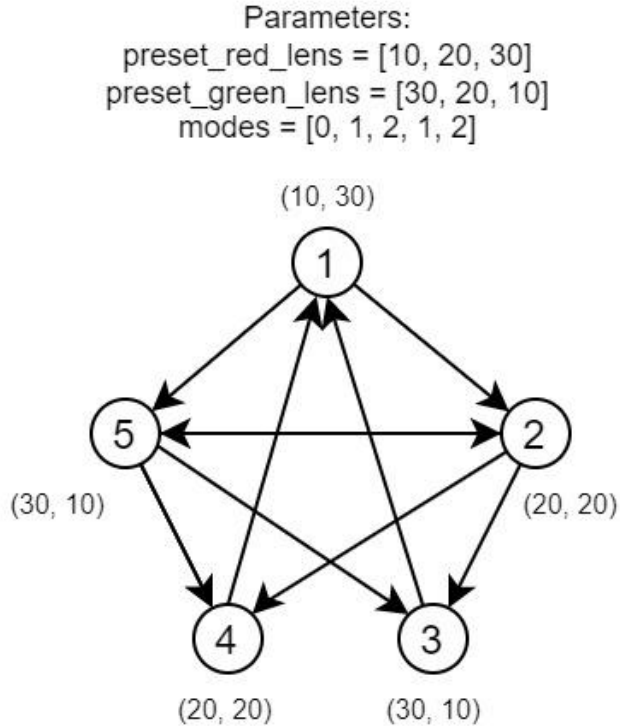
- Treat every junction separately, assign two parameters: red_len_n and green_len_n to describe the scheduling at n-th junction
- (Pro) Very descriptive, covers all possible search space
- (Con) Explosion of search space, take extremely long to converge, impractical to carry out in real time
- (Con) Large search space also leads to higher risk to settling in local minima instead of global minima

Uniform Scheduling



- Another extreme, let every junction share the same schedule
- (Pro) Number of parameters reduce significantly to 2, much faster to emulate and optimize
- (Con) Lack of descriptiveness, unable to adjust for the differences in junctions
- Performs surprisingly well in uniform networks (e.g. ring networks), but poorly in more complex ones. Overall an adequate choice for fast, real-time optimization.

Preset Scheduling



- A hybrid between uniform and distinct scheduling
- Assume there are only a fixed set of schedules (preset schedules), let the junctions choose within this fixed set
- Reduces search space drastically, but still exponential due to the combinatorial nature of the model.
- Could be an inspiration for more efficient optimization techniques.
- Variant: forced preset - we force the preset schedules to span an even range, then let the junctions choose within the set

Can we achieve constant-time optimization?

- Preset scheduling achieves reasonable convergence speed
- But it is subjected to exponential increase in search space.
- As size of network increases, we observe that the time its takes to converge to optimal values becomes much longer.

It might be possible to achieve constant-time optimization using following method:

1. Pretrain the model to assign modes , either using GP or assign using graph properties such as degrees/centrality of nodes.
2. In the real-time system, we only finetune preset schedules, which have constant number of parameters and constant upper-bound on converge time.

Evaluation of the models

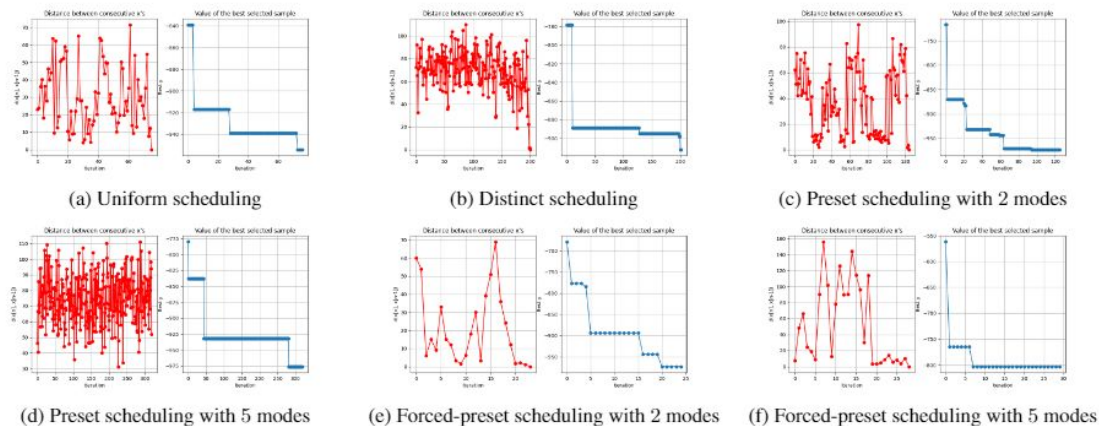


Figure 4. Convergence graph for running Bayesian optimisation on a 5-junction network

Schedule	Opt. schedule	Opt. reward	Convergence iter.
Uniform	[17, 12]	931	78
Distinct	[(21, 38), (30, 41), (26, 13), (32, 29), (26, 36)]	964	200
Preset (2 modes)	schedules: [(16, 29), (54, 9)] modes: [0, 0, 0, 1, 0]	1014	125
Preset (5 modes)	schedules: [(15, 13), (40, 47), (31, 17), (14, 7), (29, 31)] modes: [0, 2, 2, 2, 0]	984	322
Forced-preset (2 modes)	cycle length: 6 modes: [0, 0, 0, 1, 1]	992	24
Forced-preset (5 modes)	cycle length: 48 modes: [1, 2, 2, 3, 3]	939	29

Table 1. Experimental results for running Bayesian optimisation on a 5-junction network

Evaluation of the models

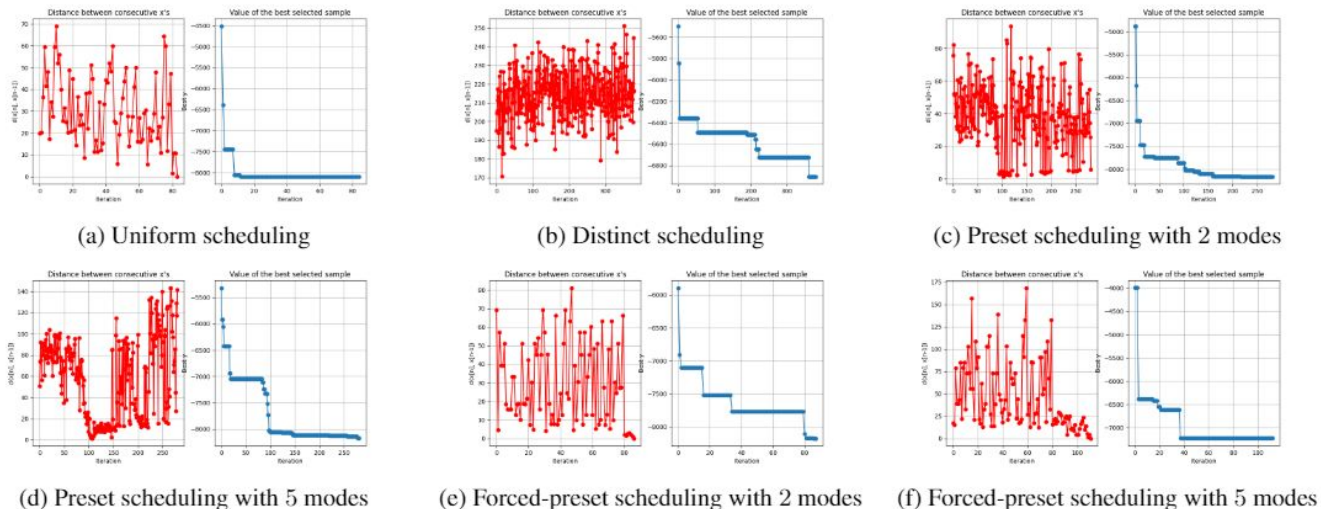


Figure 5. Convergence graph for running Bayesian optimisation on a 40-junction network

Schedule	Uniform	Distinct	Preset (2)	Preset (5)	Forced-preset (2)	Forced-preset (5)
Opt. reward	8096	6904	8164	8176	8175	7320
Convergence iter.	83	391	285	117	88	112

Table 2. Experimental Results for Bayesian optimisation on a 40-junction network

Evaluation of the models

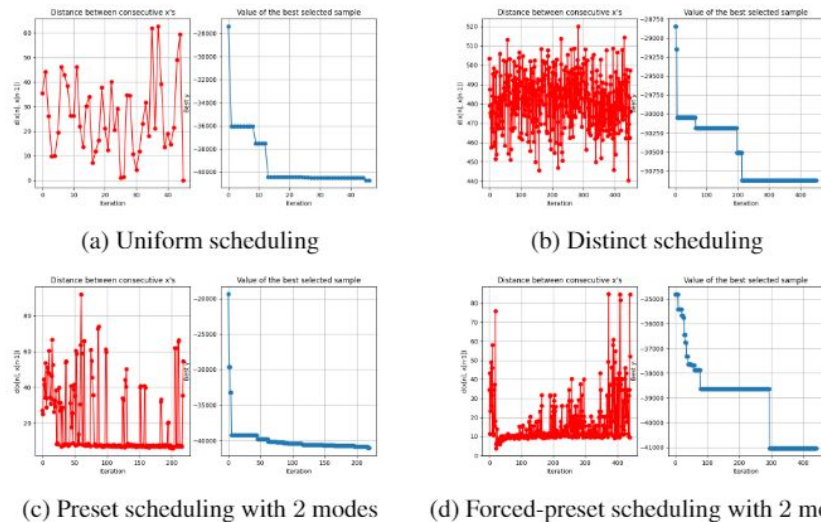


Figure 6. Convergence graph for running Bayesian optimisation on a 200-junction network

Schedule	Uniform	Distinct	Preset (2)	Forced-preset (2)
Opt. reward	40705	30873	41062	41030
Convergence iter.	45	475	218	432

Table 3. Experimental Results for Bayesian optimisation on a 200-junction network

A network diagram illustrating a complex web of connections. Numerous small, dark-colored pins are arranged on a light gray surface, connected by thin, colored lines. The lines are color-coded: yellow, orange, red, purple, and green. The network is dense in the center and left, with a more isolated cluster of green lines on the right. The word "Conclusion" is centered in white text.

Conclusion

Conclusion

- We explored the possibility of traffic optimization using GP and BO
- We have obtained results that confirmed our conjectures
- However, there are certain outliers that are worth looking into
- Ways forward:
 - Attempt other optimization techniques (e.g. Reinforcement Learning)
 - Look deeper into the anomalies
 - Explore the likelihood of our models (evidence)
 - Visualization

Thank you!