Intelligent Traffic Signal Control

Dave McKenney
Carleton University
School of Computer Science
Presentation Layout

- Problems caused by traffic
- Outline of a traffic model
- Previous Work
- Initial algorithm, tests, and results
- Moving to SUMO traffic simulator
- Modeling a real world example
- Improvement ideas
- Future Work
Why is Traffic a Problem?

Problems Caused by Traffic

- Economic, Social, Environmental, Safety issues
- 37 Million hours spent commuting daily in Thailand\(^1\)
- Avg. time spent commuting in Toronto area has increased 16% in the last 10 years\(^2\)
- 21 Million hours spent a day commuting in UK (£226 Million working time lost)\(^3\)
- Wasted fuel has large economic and environmental impact
Traffic Model

- Several things can be controlled/observed:
  - Traffic Signals
  - Vehicle Routes
  - Driver/Vehicle Behaviour
  - Roads/Streets/etc.
Traffic Signals

- Can store known information
- Can receive/calculate traffic information
- Each light implements a signal plan, composed of cycles of the light phases (green, yellow, and red)
- Global (entire traffic network) optimization is difficult
Vehicle Routes

- Real-time vehicle routing is now extremely common (GPS devices)
- Correct strategies could route drivers efficiently:
  - Balance traffic flow
  - Alleviate traffic jams
  - Avoid accidents and other road blockages
  - Take the fastest path, not just the shortest
Driver/Vehicle Behaviour

- Real world drivers act differently
- A traffic model must represent this in some way

Roads/Streets/Etc.

- Each roadway has specific attributes
- These attributes are generally static
Hierarchical/Coordinator Approach

- Almejallli et al. implemented a system which uses a coordinator agent to help find a global optimum among all agents within the system.
Genetic/Evolutionary Approach\textsuperscript{5}

Applying a Traffic Lights Evolutionary Optimization Technique to a Real Case, Sanchez et al.
SuRJE Traffic Simulation

- Uses a swarm model to simulate traffic
- Vehicles leave pheromone as they drive
- Pheromone detection is used to choose actions

Swarm-based Traffic Simulation with Evolutionary Traffic Light Adaptation, Penner and Hoar
Creating a SuRJE Simulation

Swarm-based Traffic Simulation with Evolutionary Traffic Light Adaptation, Penner and Hoar
Creating a SuRJE Simulation
Running a SuRJE Simulation

Swarm-based Traffic Simulation with Evolutionary Traffic Light Adaptation, Penner and Hoar
SuRJE Simulation Results

Swarm-based Traffic Simulation with Evolutionary Traffic Light Adaptation, Penner and Hoar
SuRJE Simulation Results

Swarm-based Traffic Simulation with Evolutionary Traffic Light Adaptation, Penner and Hoar
Bazzan and de Oliveira\textsuperscript{7} - Traffic Signal Group Formation

- Treat each intersection as a social insect
- Each 'insect' attempts to optimize their signal plan
- Pheromones left by vehicles motivate change in the behaviour of the 'insect'
- Goal is to create non-stop flow of traffic through sections of road in a certain direction
Traffic Signal Group Formation

- All vehicles waiting at an intersection drop pheromone at every time step
- The amount of pheromone in a given lane $l$ at time $t$ is given by the following equation:

$$d_{l,t} = \frac{\sum_{i=1}^{w} \frac{w-i}{\beta} (d_{l,t-i})}{\sum_{i=1}^{w} \frac{w-i}{\beta}}$$
Traffic Signal Group Formation

- Below is a representation of the area visible to an intersection.
Traffic Signal Group Formation

- The stimulus $s$ of a signal plan $j$ is then computed based on:
  1) The pheromone levels within the lanes during each phase
  2) The plans being used by visible neighbours
  3) The proportion of the entire time of cycle $k$ that the light is green ($\Delta k$), given by:

$$
\Delta_k = \frac{(\text{time}_{\text{end}} - \text{time}_{\text{begin}})}{\text{time}_{\text{cycle}}}
$$
Traffic Signal Group Formation

- The equation used to calculate stimulus is:
  \[ S_j = \alpha \sum_{k=0}^{n} (d_{in_{k,t}}) \Delta_k + (1 - \alpha) \frac{a_j}{A} \]

- The probability of a signal plan \( j \) being implemented by an intersection \( i \) is then given by:
  \[ T_{\theta ij} (S_j) = \frac{S_j^2}{S_j^2 + \theta_{ij}^2} \]
Traffic Signal Group Formation

- $\theta_{ij}$ is reinforced at the end of every time interval $\delta_t$ (set at 10 minutes for their tests) using the following formula:

$$\theta_{ij} = \theta_{ij} - l \delta_t$$

- Where $l$ is a learning coefficient calculated with:

$$l = 1 - 2\sigma$$
Traffic Signal Group Formation

Swarm Intelligence Applied to Traffic Lights Group Formation, Oliveira and Bazzan
Gershenson - Self-Organizing Traffic Lights

- Defined 3 methods of controlling traffic lights:
  - SotlRequest
  - SotlPhase
  - SotlPlatoon

- Claims no communication between lights is necessary, but fail to explain a method in which the information required is passed
Gershenson - Self-Organizing Traffic Lights: Sotl-Request

• Each traffic light keeps a counter $K_i$ which is reset to 0 every time the light switches

• $K_i$ is increased by number of approaching vehicles at each time step

• When $K_i$ reaches a specified threshold ($\theta$), the lights at the intersection switch (red-$\rightarrow$green, green-$\rightarrow$red)

• Problem: Fast switching
Gershenson - Self-Organizing Traffic Lights: Sotl-Phase

- Uses the same strategy as Sotl-Request, only with a predefined minimum phase length ($\phi_{\text{min}}$)
- Intersections now keep another counter ($\phi_i$) which represents the time steps since the last traffic light change
- Even when $K_i > \theta$, lights will not switch until $\phi_i > \phi_{\text{min}}$
Gershenson - Self-Organizing Traffic Lights: Sotl-Platoon

• Adds two more restrictions to Sotl-Phase
• Before switching lights, Sotl-Platoon checks for platoons crossing the intersection
• If a car is within $\omega$ patches of the intersection, the lights will not switch
• This restriction is not taken into account if there is more than $\mu$ vehicles approaching the intersection
Gershenson - Self-Organizing Traffic Lights: Results

- △ sotl-request  → sotl-phase  → sotl-platoon

![Graph showing results of self-organizing traffic lights]
Decentralized Traffic Signal Control With Intersection Communication

- Traffic state can be determined through communication between intersections
- Sensors at each intersection detect traffic state
- Pieces of information can be sent to neighboring intersections
- Intersections use local observation and information from neighbors to generate signal plans
Decentralized Traffic Signal Control
With Intersection Communication

• The algorithm begins by calculating a green length for a single direction as follows:

\[ P_w = \frac{AC_w}{(AC_w + AC_N)} \]

\[ TG_1 = P_w \times CL \]

• Where AC_w and AC_N are the average cars in the west/north approaching roads over the time window and CL is the cycle length.
Decentralized Traffic Signal Control
With Intersection Communication

• To facilitate coordination and prevent extremely volatile phase lengths, the neighbor weight (NW) and current time weight (CTW) parameters are introduced:

\[
TG_2 = (1 - NW)TG_1 + NW\left(\frac{\sum_{i \in X} TG_i}{|X|}\right)
\]

\[
TG_3 = (1 - CTW)TG_2 + (CTW \times CGT)
\]

• Where X is the set of all valid neighbors, and CGT is the amount of green time currently allotted in this direction.
Decentralized Traffic Signal Control With Intersection Communication

• Finally, the amount of green time is bound so a minimum amount of time (MT) is allowed in each direction during each cycle using the following equations:

\[ TG_4 = \max(TG_3, MT) \]
\[ TG_F = \min(TG_4, (CL - MT)) \]
Decentralized Traffic Signal Control With Intersection Communication

- The performance of this algorithm was tested and compared to results found using 3 fixed signal plans (30/30, 40/20, 20/40), as well as the task allocation approach outlined above.

- Each strategy was tested on 17 different distributions, averaging the results found over 25 runs on each distribution.
### Decentralized Traffic Signal Control With Intersection Communication

<table>
<thead>
<tr>
<th>Distribution</th>
<th>30/30 Avg</th>
<th>30/30 SD</th>
<th>40/20 Avg</th>
<th>40/20 SD</th>
<th>20/40 Avg</th>
<th>20/40 SD</th>
<th>TA Avg</th>
<th>TA SD</th>
<th>Prop Avg</th>
<th>Prop SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed Even</td>
<td>224.80</td>
<td>0.69</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>231.67</td>
<td>1.56</td>
</tr>
<tr>
<td>Fixed 2:1</td>
<td>281.06</td>
<td>34.79</td>
<td><strong>206.15</strong></td>
<td>1.64</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>221.36</td>
<td>1.21</td>
</tr>
<tr>
<td>Fixed 4:1</td>
<td>Failed</td>
<td>Failed</td>
<td>192.10</td>
<td>0.93</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>199.23</td>
<td>1.81</td>
</tr>
<tr>
<td>TA</td>
<td>261.15</td>
<td>12.63</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>224.13</td>
<td>1.88</td>
</tr>
<tr>
<td>Sin/Cos</td>
<td>238.80</td>
<td>11.70</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>179.07</td>
<td>1.05</td>
</tr>
<tr>
<td>F1</td>
<td>302.95</td>
<td>42.52</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>244.76</td>
<td>11.35</td>
</tr>
<tr>
<td>F2</td>
<td>341.09</td>
<td>57.99</td>
<td>514.40</td>
<td>29.67</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>171.24</td>
<td>0.95</td>
</tr>
<tr>
<td>FF1</td>
<td>367.64</td>
<td>59.83</td>
<td><strong>217.03</strong></td>
<td>60.48</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>215.88</td>
<td>1.88</td>
</tr>
<tr>
<td>FF2</td>
<td>213.60</td>
<td>4.34</td>
<td>Failed</td>
<td>Failed</td>
<td>312.59</td>
<td>23.31</td>
<td>Failed</td>
<td>Failed</td>
<td>213.77</td>
<td>1.02</td>
</tr>
<tr>
<td>FF3</td>
<td>230.94</td>
<td>22.42</td>
<td>685.37</td>
<td>59.73</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>217.63</td>
<td>1.86</td>
</tr>
<tr>
<td>FF4</td>
<td>292.61</td>
<td>86.30</td>
<td>356.38</td>
<td>23.84</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>213.34</td>
<td>1.26</td>
</tr>
<tr>
<td>FF5</td>
<td>312.12</td>
<td>57.31</td>
<td><strong>205.67</strong></td>
<td>1.29</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>213.47</td>
<td>1.07</td>
</tr>
<tr>
<td>FF6</td>
<td>200.17</td>
<td>3.43</td>
<td>264.15</td>
<td>20.16</td>
<td>772.78</td>
<td>81.58</td>
<td>380.27</td>
<td>62.51</td>
<td>196.92</td>
<td>0.96</td>
</tr>
<tr>
<td>FF7</td>
<td>282.59</td>
<td>55.59</td>
<td>410.92</td>
<td>41.53</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>208.47</td>
<td>1.52</td>
</tr>
<tr>
<td>FF8</td>
<td>195.31</td>
<td>1.91</td>
<td>417.11</td>
<td>29.03</td>
<td>221.67</td>
<td>11.68</td>
<td><strong>171.41</strong></td>
<td>26.18</td>
<td>183.52</td>
<td>1.26</td>
</tr>
<tr>
<td>FF9</td>
<td>238.38</td>
<td>19.69</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>229.05</td>
<td>4.79</td>
</tr>
<tr>
<td>FF10</td>
<td>381.49</td>
<td>122.85</td>
<td><strong>202.38</strong></td>
<td>1.57</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>Failed</td>
<td>197.14</td>
<td>0.86</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bests</th>
<th>2</th>
<th>3</th>
<th>0</th>
<th>1</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failures</td>
<td>1</td>
<td>6</td>
<td>14</td>
<td>15</td>
<td>0</td>
</tr>
</tbody>
</table>
Decentralized Traffic Signal Control With Intersection Communication

The graph shows the average trip time (in steps) for different traffic distributions. The x-axis represents the traffic distribution, while the y-axis shows the average trip time. The data points indicate the performance of the Best Other Algorithm Average and the Proportional Algorithm Average.
Moving to a Traffic Simulator

• The traffic model within NetLogo is very basic
• A real traffic simulator allows for much more realistic behavior
• Many open source traffic simulators are available (SUMO\(^9\) is used here)
• The algorithm and traffic model were moved to SUMO for testing in a more realistic environment
Adding Intersection Offsets to the Algorithm

- The addition of offset values allow intersections to further coordinate.
- The main goal is to have a traffic light turn green as a group of cars approaches, allowing them to travel through without stopping.
- Each intersection has times within the cycle at which it will turn green in the W/E or N/S direction (these values are calculated using known/observed data).
Adding Intersection Offsets to the Algorithm

- Steps to creating an offset value
  1) Determine which neighbor to offset with
  2) Determine the estimated trip time from that neighbor to the intersection in question (distance/speed)
  3) Set the intersections green switch time to the sum of the neighbor’s green switch time and the estimated trip time (taking cycle length into account)
New Parameters/Operations

• Two different methods for calculating average cars were implemented (straight average and time-weighted average)

• Three different methods for counting vehicles on a road were implemented (number of vehicles per unit of road, number of vehicles on the road, and number of stopped vehicles)

• Two offset methods (average over all neighbors, offset with neighbor in most saturated direction)
Results from Simulation in SUMO

Performance Comparison of Different Control Strategies/Parameter Values

PM0: Time insensitive data, PM1: Time sensitive data
EM0: Vehicles/m, EM1: Vehicles on street, EM2: Number of stopped cars
Testing with Real World Data

• Data supplied by the City of Ottawa has allowed for a real world area to be modeled within SUMO
• Includes a realistic road network (7x9 blocks)
• Hourly traffic counts have been supplied and integrated into the model
• Turning rates have been calculated using the given data, allowing for vehicle route generation
• Signal plans currently used within the City are also available to test against
Real World Difficulties

- Using real world data complicates signal generation
- Example #1: Offsetting lights is much more difficult
- Example #2: More complicated intersection logic (e.g. turning lanes, advanced green lights, etc.)
Real World Difficulties

• Several things need to be addressed by an algorithm within the real world:
  – All possible offsets need to be considered
  – Different network structures must be taken into account
  – Different signal logics may be required
Possible Improvements to the Algorithm

- Prediction of future traffic volumes
- Different strategies for different volume levels (e.g. offset vs. clearing)
- Dynamic cycle lengths
- Improved information propagation
- Dynamic speed limits
Future Work

- Improvements in traffic modelling (perhaps some sort of standard?)
- Improvements in traffic simulation
- Inclusion of pedestrians into models
- More testing on real data
- Dynamic traffic routing
References


References


References

