Optimizing Trafic Light algorithms in Urban Street Grids

Introduction to Computational Science

Jan Kramer
Klaas Kliffen

January 28, 2016

Abstract

Abstract

Contents

1 Introduction **2**

2 Related work **2**

3 Concept **2**

3.1 Grid of intersections 2

3.2 Algorithms 2

4 Implementation **3**

5 Results **3**

6 Conclusion **3**

6.1 Future work 3

1 Introduction

2 Related work

Wiering [1]

3 Concept

3.1 Grid of intersections

3.2 Algorithms

For this assignment two different algorithms will be used. There will also be a version of both algorithms with loop detection. A time stamp will be used as an input for each of the algorithms, along with a variable for setting the amount of time steps for a light to switch.

Simple

The simple algorithm is based on old-fashioned traffic handling by a human in the center of the intersection pointing at the lane which may move. At each time step, one of the four directions will get the green light and cars can move. After a given amount of time steps, the next direction will be set to green and the current direction set to red. In pseudo code:

```
SET direction TO (time stamp DIV switch time) MOD number of
    directions
SET lights of lanes in direction TO green
```

Two Sided

The two sided algorithm uses the fact that two opposing lanes can move at the same time. Cars moving straight forward or turning right can move at the same time as the cars on the opposing lane moving forward and right. The same is true for the lane turning left. In pseudo code:

```
SET direction TO (time stamp DIV switch time) MOD number of
    directions
SET light of the lane in the direction TO green
SET the light in the opposing lane TO green
```

Loop detection

The loop detection can be applied to both of the previous algorithms. For this, each traffic light need to keep track of which lanes currently have green light and at what time stamp the last change happened. It also needs to keep track of the front of each lane and whether a car is present in it. It will only set lights to green if the next lanes according to the algorithm are not empty. If it encounters an empty lane, it will look for the next lane. If the next lane is not empty, set those lights to green. If all lanes are empty, keep all lights red and wait for a car to appear. If the change time has not yet expired, check if there are still cars present. If not, schedule a new change. In pseudo code:

```
SET direction TO stored direction
IF time stamp - last change time >= switch time THEN
  SET direction TO direction from algorithm
  IF front of lanes are filled THEN
    SET lights of the lanes from the algorithm TO green
    SET last change time TO time stamp
    SET stored direction TO direction
  ELSE
    FOREACH other direction
      IF front of other lanes in direction is filled THEN
        SET lights of that direction TO green
        SET last change time TO time stamp
        SET stored direction TO direction
        END LOOP
ELSE
  IF front of lanes in direction are empty THEN
    SET last change time TO time stamp - switch time
```

4 Implementation

5 Results

6 Conclusion

6.1 Future work

- Setting chance choosing between the left lane or the other independently. Now all is equal
- Giving cars incentive to drive to a certain destination
- Make the amount of cars change over time (rush hour effect)

References

[1] Wiering, M. Multi-agent reinforcement learning for traffic light control, 2000.