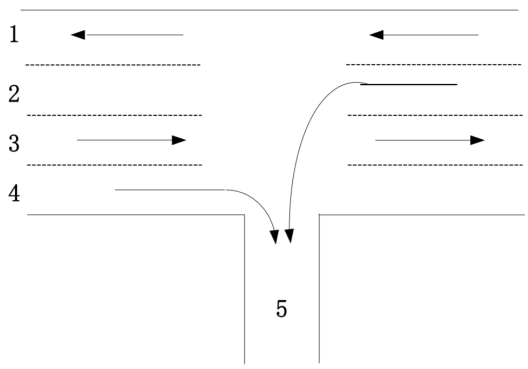
Report on problem 2

Haifei Wang

October 17, 2021

1 Problem

There is a T-junction shown below:



Every lane(1, 2, 3, 4, 5) can accommodate only one car at a time.
Given the traffic flow:

lane	1	2	3	4
number of cars per minute	20	12	18	10

How to set up the traffic light (red and green, no yellow) to optimize the traffic? The answer should include:
- In which lanes should traffic lights be set up.
- The duration time of green light and red light.

2 Analysis

2.1 Boiling Down the Problem

Apparently, lane 1 does not need a traffic light for it does not interferes with other lanes for all intent and purposes.

Cars both in lane 2 and lane 4 wish to enter lane 5, hence there is a competition between them.

Another important competition to notice is that if cars in lane 2 are proceeding, cars in lane 3 will be put to a halt.

Therefore, lane 2, 3, 4 have to be set up with traffic lights.

Also, with a simple reflect on the situation, we can conclude that **there are only 2 scenarios**:

1. Lane 2: red; Lane 3 and 4: green
2. Lane 2: green; Lane 3 and 4: red

Hence, the problem reduces to how to allocate time for these two situations.

2.2 Assumptions

The following assumptions are made:
Assumption 1 Cars arrive at the junction at a constant rate.
Assumption 2 The traffic light cycle T is fixed and known. For example, 90 seconds.
Assumption 3 After the light turns green, it still takes some time for the vehicle to start proceeding. This includes the time for cars from other lanes to fully get through (since there is no yellow light), and the time needed for acceleration.

2.3 Objective function

The quantity to be maximized / minimized is a function of the input parameters. This is called the objective function. The solution of the problem depends on the choice of objective function.

I select the objective function as ”the sum of the detention time of every car per cycle”.

3 Modeling

We can plot “The number of cars waiting at the junction” versus time t.

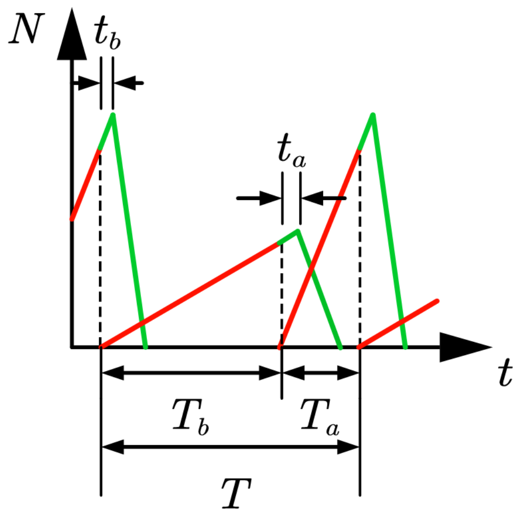


Figure 1

Quantities in Fig. (1) are listed as follows:

1. N: The number of cars waiting at the junction.
2. T_a: The duration time of green light of lane 2.
3. T_b: The duration time of green light of lane 3 and lane 4.
4. $\nu_{a,b}$: The slope of rising lines, i.e., the rate at which cars arrives at the junction.
5. $k_{a,b}$: The slope of declining lines, i.e., the rate at which cars leaves the junction (when the light is green) minus the rate at which cars arrives at the junction.
6. $t_{a,b}$: Time needed for the vehicle to start proceeding after the light turns green.

   All quantities are measured in the unit of seconds. Subscript a indicates lane 2, and subscript b indicates lane 3 and 4.

In Fig. (1), after the light has turned green (after the vertical dashed lines), N still increases for a while ($t_{a,b}$) according to the assumption 3.

The sum of the detention time S is just the area under the curves and can be analytically calculated:

$$S = \int_0^T N(t)\mathrm{d}t = \frac{1}{2}\nu_b(T_a + t_b)^2(1 + \frac{\nu_a}{k_a}) + \frac{1}{2}\nu_a(T - T_a + t_a)^2(1 + \frac{\nu_b}{k_b}) \tag{1}$$

4 Solving Using Matlab

We can arbitrarily assign values to the quantities declared above. The pratical value of these quantities need to be attained by real world investigation.

For example, if $T = 90\mathrm{s}$, $\nu_a = \frac{12}{60}\mathrm{s}^{-1}$(given), $\nu_b = \frac{28}{60}\mathrm{s}^{-1}$(given), $k_a = 0.5\mathrm{s}^{-1}$, $k_b = 1.5\mathrm{s}^{-1}$, $t_a = 5\mathrm{s}$, $t_b = 3\mathrm{s}$, then using Matlab we get:

$$\arg\max_{T_a \in [0,1]} S(T_a) \approx 72(\mathrm{s}), \quad \arg\max_{T_a/T \in [0,1]} S(T_a) \approx 28\%, \tag{2}$$

meaning that if the cycle is 90 seconds, we should assign about 28% to lane 2 (25 seconds), and the remaining 72% to lane 3 and lane 4 (65 seconds). This is reasonable because there are more than twice as many cars in lane 3 and 4 as in lane 2.

The code are shown below:

```
syms Ta;
syms T;
syms va;
syms vb;
syms ka;
syms kb;
syms ta;
syms tb;

T_ = 90; % you can modify this
S = 0.5 * vb * (Ta + tb)^2 * (1 + va / ka) + 0.5 * va * (T - Ta + ta)^2
    * (1 + vb / kb);
S = subs(S, T, T_);
S = subs(S, va, 0.2);
```

```
S = subs(S, vb, 28/60);
S = subs(S, ka, 0.5);
S = subs(S, kb, 1.5);
S = subs(S, ta, 5);
S = subs(S, tb, 3);
step = 0.01; % the step used for solving minimum
[min_val, idx] = min(subs(S, Ta, 0:step:T_)); % get the minimum value and the corresponding index
Ta_rst = idx * step; % the Ta at which S reaches minimum

% show the result
fprintf("The time assigned to lane3 is: %.2f seconds \n", vpa(Ta_rst));
fprintf("Percentage: %.2f%", vpa(Ta_rst / T_ * 100, 2));
fprintf("\n");
```