Summary

We put forward a model for the traffic flow. This abstract model for the traffic circle is discrete, which well describes the continuous traffic flow in reality. This feature enables us to simulate the the movements of vehicles in the model by a program which uses a Monte Carlo algorithm.

Based on this model, we come up with two metrics, which measure the efficiency and the fairness of a traffic circle design. We consider not only the efficiency but also the fairness, which prevents our design from an extreme situation (such as the vehicles in one road have to wait for a long period of time).

Besides the greedy strategy which is commonly used by drivers, that is to enter the traffic circle as soon as possible, we will also discuss other two strategies, namely the classified-lane design and the buffering design. In the former design, we direct cars into different lanes. And in the latter, we use the outmost lane as a buffering driveway, by which conflicts between vehicles are avoided. At last, we conclude that these two designs are better than the greedy one in most of the time.

Then, for every design, three patterns of traffic flow are used to test every method we have taken into account. These three patterns have their own meaning in real world, i.e, a random flow from all roads and two oriented ones. Furthermore, we increase the intense of the these three representative flows to test the sensitivity and stability of each design. Thus, through this process, we can sift the best design out.

Finally, according to the test result of each design calculated by the algorithm, we give a specific and detailed guide to realize our methods, which are written in a technical summary in the appendix.

Traffic Circle

Team #4984

February 10, 2009

Contents

Part I
Introduction

1 Description of the Problem

Our task is to provide a mathematical stimulation of the traffic flow of a traffic circle, and determine the optimal model that best approximates the real scenario. Specifically, our task consists of following parts:

1. Provide a model for a traffic flow of a traffic circle.
2. Put forward several general methods of control traffic flow in, around and out of the traffic circle concerned.
3. Offer various metrics at the traffic flow and method level.
4. Present an algorithm to work out the approximation of the traffic flow of a specific traffic circle.
5. Compare the merits and demerits of respective specific methods generated from general methods. Choose the best way to control the traffic flow under certain conditions.

2 What is a Traffic Circle?

A traffic circle is a central island around which there are several lanes, ranging from one to three generally. Four or five roads are connected to the outmost lane. Vehicles follow specific rules to drive in, around and out of the circle, which are of vital importance in determining the efficiency and fairness of a traffic circle.

Part II
Model and Metrics

3 Model of Traffic flow in a Traffic Circle

3.1 Assumptions

1. The speed of all vehicles running in, around and out of the circle as well as on roads connected to the circle, is a fixed constant unless a vehicle has to stop due to traffic rules. And all vehicles have the same vehicle length.

2. Vehicles run on the right side of the road, driving in an anticlockwise direction. In this case, to a certain road, vehicles run out first, and then others run into the circle.

3. There is a upper limit of the number of vehicles running in a certain lane of a traffic circle.

   We say a lane is "full" if the number of vehicles reaches the limit, otherwise it is "available". The time for a vehicle to drive from its current lattice to the next is called, in short, one "TCS"(one traffic circle second).

4. In order to drive from a certain road to the nearest one, a vehicle has to enter a traffic circle instead of merely making a turn as in the case of a normal crossroad.

5. In the case of a traffic circle with more than one lanes, vehicles can enter or exit an inner lane even if all the outer lanes are full, but penalty time of one "TCS" will be added for such behaviors for the following reason:

   It seems that all the lattices of the outer lane is occupied, but due to the safety distance specified in traffic rules, vehicles in the inner lane are able to pass through the outer lane as long as vehicles in the outer lane remain stopped or run slower, the speed of the vehicle must be reduced, and thus cause the penalty time.

6. No people walk in, around or out the traffic circle. Only cars are considered in our model.

7. Twenty-four hours in a day can be divided into several periods, according to the traffic conditions. Within each period, the probability

that there is one vehicle driving on a certain road following a certain direction in a TCS is a constant. While the probability varies among different periods.

The probability of a certain kind of vehicles of a one-driveway road is the number of all such vehicles on the road over the number of moving vehicles that the road can contain most. It is the same to the expectation of such vehicles emerging per TCS.It is easy to extend the notion of probability of a one-driveway road to a multi-driveway one.

8. All drivers observe the traffic rules correctly.

3.2 Abstract Model of Traffic Flow

Suppose that the n lanes in a traffic circle are denoted by $c_0, c_1, \ldots, c_{n-1}$ from the inmost to the outmost, and the limit of the number of vehicles in these lanes (mentioned in assumption 4) are $l_0, l_1, \ldots, l_{n-1}$ respectively. The m roads are noted as $r_0, r_1, \ldots, r_{m-1}$.

According to assumption 1, the maximal number of vehicles that a given lane can contain can be calculated from the length of the lane and the speed of vehicles.

For each $i = 0, 1, ..., n-1$,

$$l_i = \frac{L(c_i)}{L(vehicle) + D(speed)}$$

where $L(c_i), L(vehicle), D(speed)$ represent the length of c_i, the average length of a vehicle and the safety distance between two adjacent vehicles running at the fixed speed mentioned in assumption 1.

We will discuss the patterns of the traffic flow (moving patterns) during various periods of time and decide whether traffic lights should be introduced or not.

As mentioned in assumption 4 and the Figure1, we conceive all lanes of the traffic circle as composed of discrete lattices. A vehicle can enter a lane if and only if a proper lattice in this lane is left empty, the traffic light is green (if there is one) and other specific conditions of designed rules are satisfied. Unless exact rules are designed to prevent potential conflict, we do not take into account the fact that vehicles may alter lane when turning around the circle for such behavior may cause traffic jam or accidents without strict rules of proper behavior of altering lane. As is mentioned in assumption 5, a vehicle can always exit the circle. Figure 2, 3, 4 illustrate the situations within 2 TCSs.

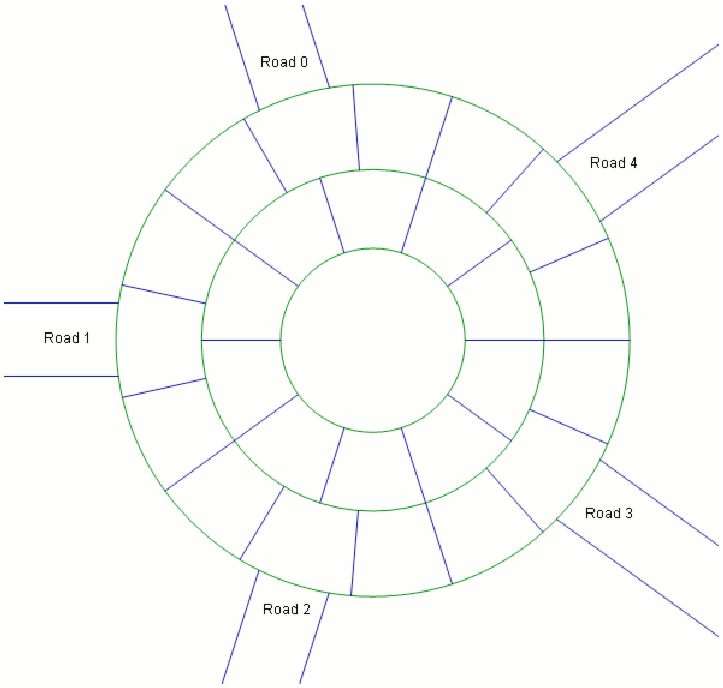

Figure 1: a 2-lane traffic circle with 5 roads

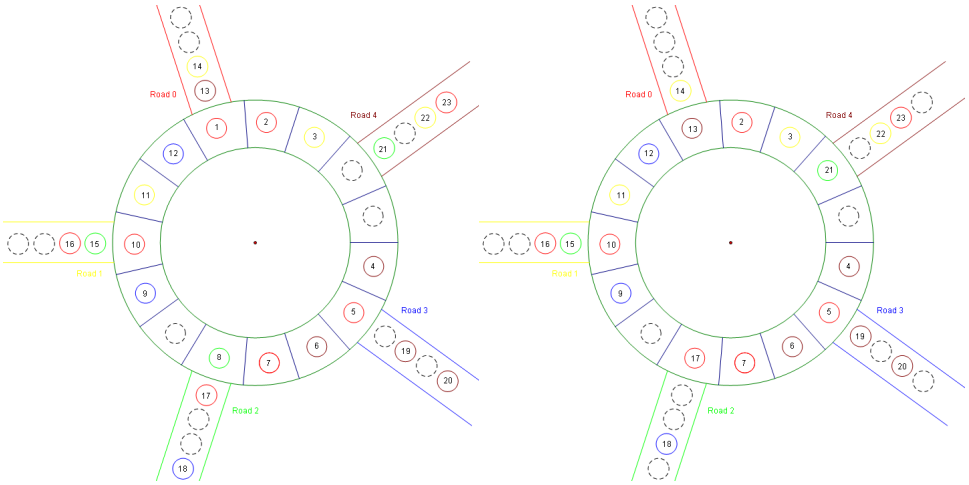

Figure 2: original state and during the first TCS

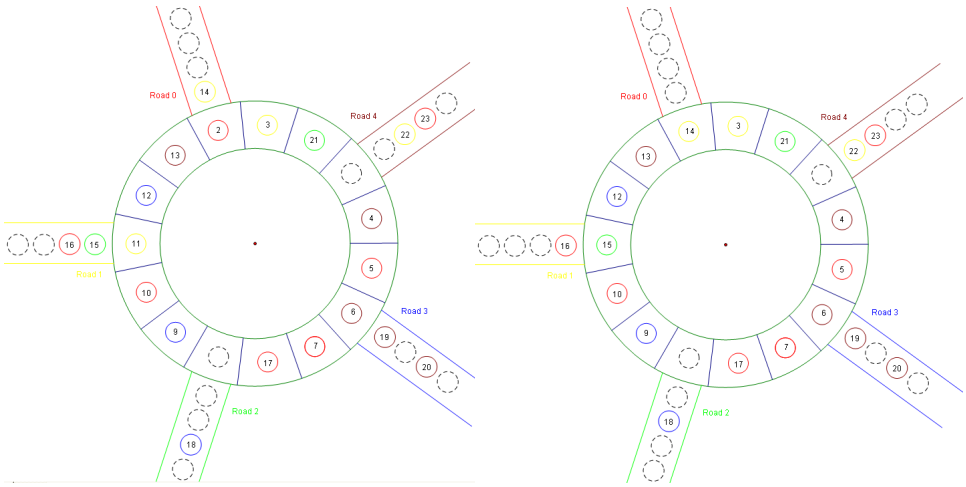

Figure 3: at the end of the first TCS and during the second TCS

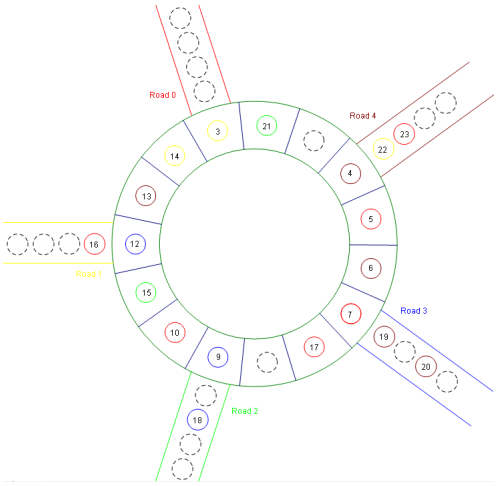

Figure 4: at the end of the second TCS

4 Metrics of Good Traffic Flows

4.1 Efficiency

The time wasted on waiting is primarily used to measure the efficiency of a traffic circle. Thus, we take as a metric the average waiting time of all vehicles, "AWT" for short. In a mathematical expression:

$$AWT = \frac{\sum t_i}{N(vehicle)}$$

where t_i is the time wasted by the ith vehicle, including the waiting time and the penalty time and $N(vehicle)$ the number of vehicles passed through the traffic circle.

4.2 Fairness

It is quite likely that the most efficient method of controlling traffic flow may add to the benefit of vehicles in one road, while hampering that of vehicles in other roads, the example of which will be provided in the later chapters. So we introduce another important metric, that is the maximum of the average waiting time of every road (MWR). However, efficiency remains the first priority. We will take the fairness into consideration when the efficiencies of two different methods are approximately equal.

$$MWR = \max_{1 \le j \le n} \frac{\sum_{i \in r_j} t_i}{N_j(vehicle)}$$

where t_i is the time wasted by the ith vehicle and $N_j(vehicle)$ the number of vehicles coming from raod r_j.

5 Some Analysis of Efficiency

5.1 Maximize the Speed

Theorem 5.1 The faster the vehicles travel, the more efficient the traffic flow is, which means every vehicle in the traffic flow should drive at the speed mentioned in the assumption 1.

Proof The time for a certain vehicle in the circle to pass through determines how long other vehicles have to wait before entering the circle. A second waited by vehicles outside the circle is caused by all the vehicles in the circle.

Thus, the waiting time caused by a certain vehicle is its passing time divided by the number of vehicles in the circle.

Moreover, the time for a vehicle to pass through the circle equals the length it should run divided by the speed.

In a random case, we consider the length that a certain vehicle travels as the expectation of the length traveled by all vehicles, which is a constant.

As far as efficiency is concerned, we should maximize the product of the number of vehicles in the circle and the speed.

Generally, the safety distance prescribed by laws is positively related to the speed. We suppose that

$$sd = \alpha s$$

where sd and s is the safety distance and the speed of a vehicle.

Subsequently, our task is reduced to maximize

$$\frac{s \times l}{a + sd}$$

or

$$\frac{s \times l}{a + \alpha \times s}$$

where l, a stand for the total length of all lanes in the traffic circle and the vehicle length of vehicles. Since l, a, α are positive, the result reaches its maximum while s is arrives at its upper bound.

□

5.2 A Seemingly Better Strategy Fails

People will argue that sometimes we should set aside the vacant lattices in the traffic circle strategically, so that vehicles in the road, lining up and waiting, is able to drive into the circle. Actually, after a delicate analysis, we come to a conclusion which betrays people's intuition.

Theorem 5.2 Any empty position in the circle should be occupied at once if there is a vehicle ready to get in, which implies that a vehicle shall fill the unoccupied position in the traffic circle in time when the rules designed allow so.

Proof We calculate the total time wasted in the following two cases:

1. The vehicle queuing in the first place of a road always drives into the nearest empty position in the traffic circle.

2. A vehicle other than the one mentioned above occupies the position.

The total time wasted by all vehicles in the first case differs from that of the second one in the following aspects:

1. All vehicles waiting on the busy road do not have to wait after the vehicle mentioned in the second case runs into the circle.

2. All vehicles outside the circle have to wait while the empty position runs to the busy road and therefore the vehicle mentioned in the second case fills in.

3. The difference between the passing time of the might-be-in vehicles in circle in the two scenarios causes the waiting time of vehicles outside the circle to differ.

Since, in reality, the number of vehicles that a lane of a traffic circle can contain is more than twice the number of the roads that connects to the circle, so the time in the second situation is twice more than that in the first one. Additionally, more vehicles in the latter situation have to wait than in the former one. Therefore, the total time wasted in the second aspect is twice more than that of the first one. As mentioned in the proof of theorem 5.1, after calculating the difference in the third aspect divided by the number of vehicles in the circle, we can take it as the time wasted by each vehicles waiting outside. It is obvious that this time is less than one TCS. The time mentioned in the first aspect, one TCS, is less than the half of the time in the second aspect. So the time wasted in the third aspect is less than the half of that of the second one. No matter whether the third one is wasted or saved by the second case, the discussion above ensures that the efficiency in the first case is higher than that in the second case.

□

Example 5.1 As mentioned in chapter 4.2, efficiency may cause extreme unfairness. We now use Theorem 5.2 to give an example.

Consider a simple case of a traffic circle with one lane and three roads. All vehicles from r_1 running to r_2 and all vehicles from r_2 running to r_1. Suppose the probability of the appearance of vehicles of r_1, r_2 are both 1. Since no empty lattice would be created at the conjunction of r_0 and the circle as well as the premise that any empty lattice created in other places would be occupied at once, no matter what positive probability of the appearance of vehicles from r_0, we conclude that no vehicles from r_0 can run into the circle once the lane becomes full according to Theorem 5.2. In brevity, the vehicles

from r_0 have to wait until the probability(i.e. the time period) changes, which could require a considerably long time.

Though the example is a bit unusual in our real life, we can learn the importance of fairness as a metric from it.

6 Various Types of Traffic Circles and Corresponding Methods

6.1 Traffic Circles with One Lane

Using theorem 5.1, vehicles already in a circle should not stop . Moreover, vehicles can run out with no influence on others. Therefore, we have only to consider on the rules of how vehicles run into the circle.

Using Theorem 5.2, empty position should be occupied at once, that is when the empty lattice appears on the entrant from a road where vehicles are waiting, the first vehicle should run in. If we set up some traffic lights in the circles, sometimes empty lattice will not be filled at once. So traffic lights are not recommended when efficiency is of primary concern.

6.2 Traffic Circles with Several Lanes

As mentioned in chapter 3.2, we now do not consider the situation where vehicles change from one lane to another.

Traffic lights are not recommended for the same reason of chapter 6.1.

Two optional methods are offered here:

1. No difference exists between lanes. Vehicles run into the circle if at least one lane is not full. We assume that if more than one lane is not full, the vehicle runs into the inmost one available.

2. Vehicles with different distance to travel in the circle run into different lanes, which suggests that a certain vehicle has one and the only one lane on which it could run around the traffic circle.

Part III
Solution for Various Kinds of Traffic Circles

7 Classified-Lane Design and Programme for Testing

In the model given above, we design a series of rules to guide drivers to a certain circle. For example, a rule specifies that drivers who intend to driver the next road should drive in the outer lane and others the inner. We will experiment with all possible rules and find out the best.

In order to approach the real scenario, we employ a Monte Carlo algorithm to stimulate a traffic flow and find out the best rule-design.

Figure 5 shows the flowchart in calculating the efficiency and fairness given the traffic flow pattern and the rules are given.

The source code refers to the appendix B.

We use this programme to calculate the two metrics of every possible method in order to find the optimal one.

8 Traffic Circles with Four Roads

We consider six most representative situations in our real life:

1. N1(normal 1): the probabilities of all kinds of vehicles are the same and the traffic is not busy. Table 1 shows probability matrix.

↗	r_0	r_1	r_2	r_3
r_0	0	0.1	0.1	0.1
r_1	0.1	0	0.1	0.1
r_2	0.1	0.1	0	0.1
r_3	0.1	0.1	0.1	0

Table 1: N1

The jth element in row i of the matrix means the possibility of vehicles from road i to road j. This convenience will also be taken in the following situations.

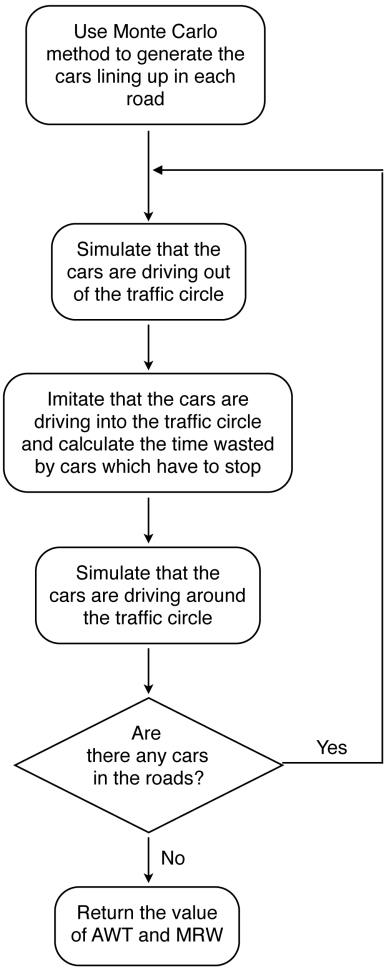

Figure 5: Algorithm Flowchart

2. N2(normal 2): the probabilities of all kinds of vehicles are the same and the traffic is busy. Table 2 shows probability matrix.

↗	r_0	r_1	r_2	r_3
r_0	0	0.2	0.2	0.2
r_1	0.2	0	0.2	0.2
r_2	0.2	0.2	0	0.2
r_3	0.2	0.2	0.2	0

Table 2: N2

3. N3(normal 3): the probabilities of all kinds of vehicles are the same and the traffic is extremely busy. Table 3 shows probability matrix.

↗	r_0	r_1	r_2	r_3
r_0	0	0.3	0.3	0.3
r_1	0.3	0	0.3	0.3
r_2	0.3	0.3	0	0.3
r_3	0.3	0.3	0.3	0

Table 3: N3

4. E1(east 1): the probability of vehicles to one road, say r_0, is much more than that of any other road, which might happen when there is a business center on r_0 during the rush hours in the morning. Table 4 shows probability matrix.

↗	r_0	r_1	r_2	r_3
r_0	0	0.1	0	0.1
r_1	0.3	0	0	0.1
r_2	0.3	0.1	0	0.1
r_3	0.3	0.1	0	0

Table 4: E1

5. E2(east 2): the probability of vehicles to one road, say r_0, is much more than that of any other road and the traffic is busier than the situation above. Table 5 shows probability matrix.

6. SW1(southwest 1): the probability of vehicles to two nearby roads, say r_2, r_3, is much more than that of the other two road, which might happen when there is a business center between r_0, r_1 during the rush hours in the morning. Table 6 shows probability matrix.

↗	r_0	r_1	r_2	r_3
r_0	0	0.2	0	0.2
r_1	0.6	0	0	0.2
r_2	0.6	0.2	0	0.2
r_3	0.6	0.2	0	0

Table 5: E2

↗	r_0	r_1	r_2	r_3
r_0	0	0.1	0.2	0.3
r_1	0.1	0	0.3	0.2
r_2	0.1	0	0	0.1
r_3	0	0.1	0.1	0

Table 6: SW1

7. SW2(southwest 2): the probability of vehicles to two nearby roads, say r_2, r_3, is much more than that of the other two road and the traffic is busier than the situation above. Table 7 shows probability matrix.

↗	r_0	r_1	r_2	r_3
r_0	0	0.15	0.3	0.45
r_1	0.15	0	0.45	0.3
r_2	0.15	0	0	0.15
r_3	0	0.15	0.15	0

Table 7: SW2

Figure 6 visualizes these situations.

Without losing of generality, we suppose that the outmost circle can contain 20 vehicles, the inner 16(if the traffic circle has 2 lanes) and the inmost 12(if 3 lanes).

Fairness will be taken into account if the efficiency of a method is 10 percent deviated from the best.

Table 8 shows the codes of optimal methods(method are encoded into numbers). In appendix A, we will show how to convert the codes into their corresponding methods. And in appendixC, the methods corresponding to each traffic condition are given.

Note that the method used in the one-lane situation is the most natural way mentioned in the discussion in chapter 6.1.

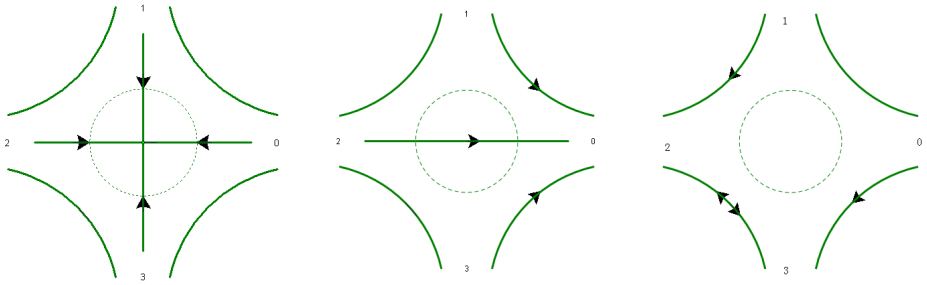

Figure 6: Normal, East, Southwest

	1 lane(TCS)		2 lanes(TCS)			3 lanes(TCS)		
Type	AWT	MRW	AWT	MRW	Code	AWT	MRW	Code
N1	0.79	0.95	0.23	0.31	61	0.14	0.24	143+3
N2	122	140	1.21	1.78	51	0.51	1.18	47+35
N3	412	426	92.3	125	51	32.5	90.4	63+13
E1	43.9	130	0.38	0.84	204	0.10	0.15	143+13
E2	514	978	92	332	53	0.88	2.23	143+13
SW1	76.8	206	0.27	0.43	140	0.15	0.39	11+9
SW2	311	698	9.11	22.95	156	0.28	1.15	227+3

Table 8: Four Roads

After analyzing the data here, we can abstract a criterion for rebuilding the traffic circle.

- For the normal situation (such as N1, N2 and N3), we build more lanes to resolve a severe traffic jam.
- For a oriented traffic flow (such as E1, E2, SW1 and SW2), we should set a specific rule to control the traffic. In addition, different number of lanes resolves different tense of traffic flow.

9 Traffic Circles with Five Roads

We consider six similar most representative kinds of situations as discussed in chapter8. The tables of probability are shown in Table 9,10,11,12,13 and 14:(the meaning of labeled types is similar)

↗	r_0	r_1	r_2	r_3	r_4
r_0	0	0.1	0.1	0.1	0.1
r_1	0.1	0	0.1	0.1	0.1
r_2	0.1	0.1	0	0.1	0.1
r_3	0.1	0.1	0.1	0	0.1
r_4	0.1	0.1	0.1	0.1	0

Table 9: N1

↗	r_0	r_1	r_2	r_3	r_4
r_0	0	0.2	0.2	0.2	0.2
r_1	0.2	0	0.2	0.2	0.2
r_2	0.2	0.2	0	0.2	0.2
r_3	0.2	0.2	0.2	0	0.2
r_4	0.2	0.2	0.2	0.2	0

Table 10: N2

By analogy with the chapter 8, we suppose that the outmost circle can contain 20 vehicles, the inner 15(if the traffic circle has 2 lanes) and the inmost 10(if 3 lanes). And fairness will be taken into account if the efficiency of a method is 10 percent deviated from the best.

Table 15 shows the codes of optimal methods(method are encoded into numbers).

↗	r_0	r_1	r_2	r_3	r_4
r_0	0	0.1	0	0	0.1
r_1	0.2	0	0	0	0.1
r_2	0.3	0.1	0	0	0.1
r_3	0.3	0.1	0	0	0.1
r_4	0.2	0.1	0	0	0

Table 11: E1

↗	r_0	r_1	r_2	r_3	r_4
r_0	0	0.2	0	0	0.2
r_1	0.4	0	0	0	0.2
r_2	0.6	0.2	0	0	0.2
r_3	0.6	0.2	0	0	0.2
r_4	0.4	0.2	0	0	0

Table 12: E2

↗	r_0	r_1	r_2	r_3	r_4
r_0	0	0.1	0.1	0	0
r_1	0.1	0	0	0	0.1
r_2	0.2	0.2	0	0	0
r_3	0.3	0.3	0	0	0
r_4	0.2	0.2	0	0	0

Table 13: SW1

↗	r_0	r_1	r_2	r_3	r_4
r_0	0	0.15	0.15	0	0
r_1	0.15	0	0	0	0.15
r_2	0.3	0.3	0	0	0
r_3	0.45	0.45	0	0	0
r_4	0.3	0.3	0	0	0

Table 14: SW2

	1 lane(TCS)		2 lanes(TCS)			3 lanes(TCS)		
Type	AWT	MRW	AWT	MRW	Code	AWT	MRW	Code
N1	15.0	25.8	0.66	0.81	1698	0.35	0.64	1374+103
N2	267	285	69.0	134	548	19.3	60.2	499+398
E1	83.9	280	0.61	0.98	1994	0.23	0.36	2599+86
E2	374	911	83.8	153	2269	3.19	5.93	1999+105
SW1	113	313	0.59	0.81	1974	0.22	0.49	1374+108
SW2	269	595	21.8	46.7	453	0.43	0.81	2749+237

Table 15: Five Roads

Note that the method of one lane is the most natural way due to the discussion in chapter 6.1.

We can abstract almost the same conclusion mentioned at the end of the chapter 8. However, as the vehicles become more in a 5-road traffic circle than in a 4-road one, a two-lane design can not make significant improvement to a one-lane design. This phenomenon implies us to find a better design to handle this situation which we will discuss in chapter 10.

Part IV
Further Discussion

10 The Effect of Changing Lanes

We now take changing lanes into account. We assume that every road has two driveways in order to classify vehicles with different destinations.

10.1 The Buffering Design for Traffic Circles with Two Lanes

In Figure 7, vehicles from r_i to r_{i+1} drive into the outer lane and others drive into the inner one. Vehicles in the inner lane change to the outer lane at the road before its destination, i.e., vehicles to r_i changes to the outer lane at the connection of the circle and r_{i-1}.

The situation of five roads is similar.

Our method concerning changing lanes are deigned as follow. In a TCS, four events that may happen are listed below,

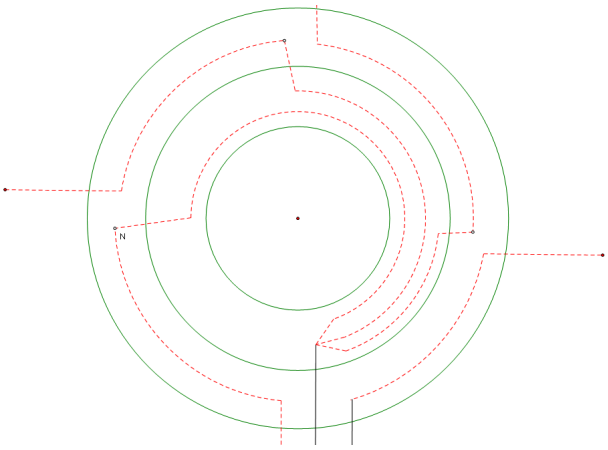

Figure 7: Changing Lanes

1. A vehicle arrived at its destination of this lattice drives out immediately.
2. A vehicle in the inner lane changes to the outer. As describe in the model of changing lanes above, the vehicles in the outer lane never cross the conjunction of a road and the circle, the vehicle in the inner lane will not conflict with others in the outer lane.
3. A vehicle whose destination is the next road drives in if the second event does not take place there.
4. A vehicle at the head of a road drives into the circle if there is a vacant lattice for it.

In assumption 5, we introduce a notion, penalty time, to punish the vehicles conflict with the vehicles in the outer lane while they are entering to the inner. However, the penalty time could be avoided if the road consists of two driveways. Here is the new rule added to the rules above. It requires drivers to drive into a specific driveway according to their destination, that is to say, the drivers whose destination is the next road drive on the right way and others the left. This ensures that no conflict will happen as is shown in the Figure 8, 9, 10and 11 on page 21 and 22.

10.2 Efficiency

The programme for calculating the AWT here are almost the same as the one in chapter 7. The source code refers to the appendix B.

Also, we use three kinds of situations to test this Buffering design.

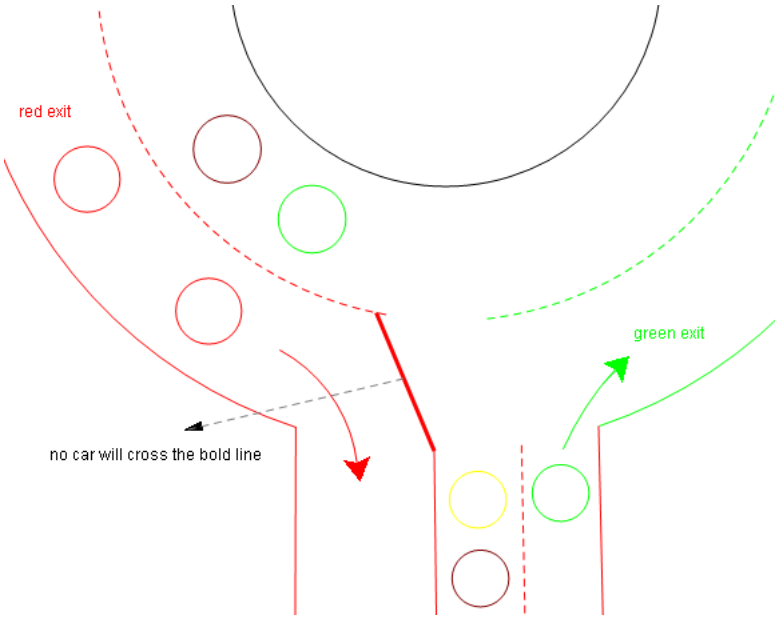

Figure 8: Illustration 1

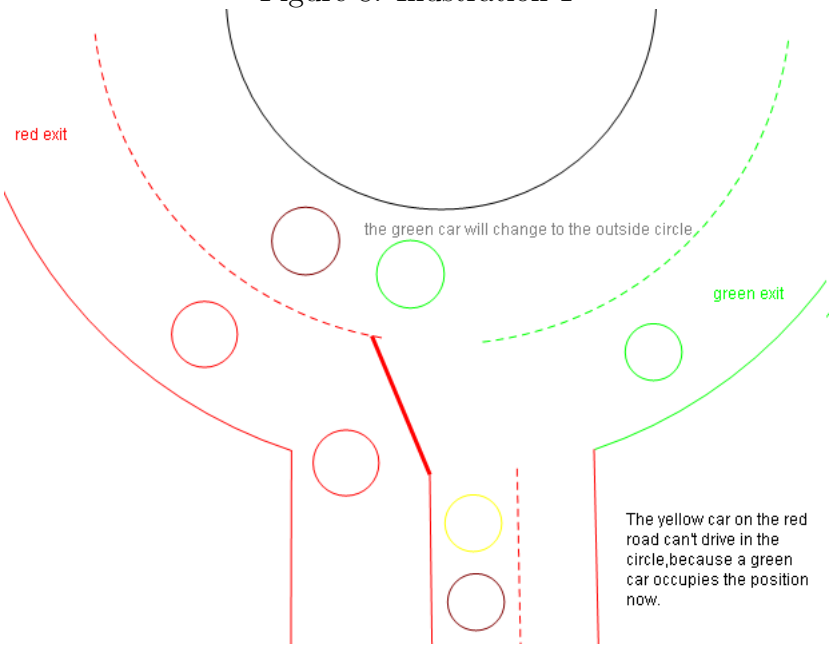

Figure 9: Illustration 2

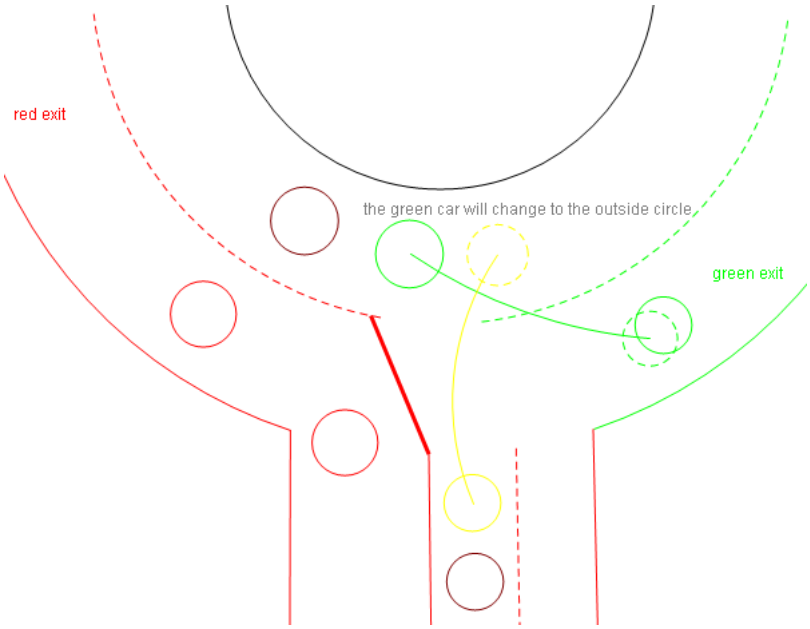

Figure 10: Illustration 3

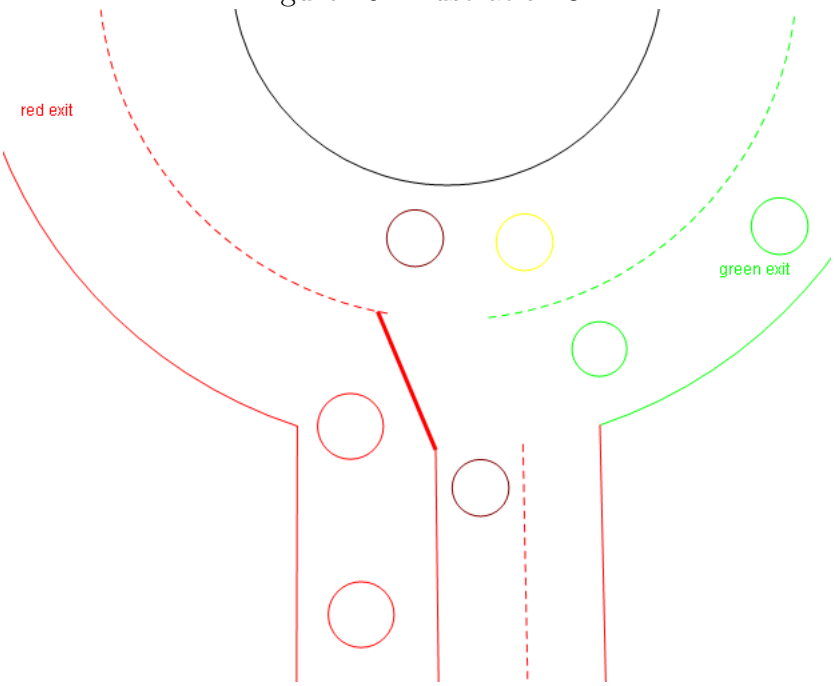

Figure 11: Illustration 4

1. N(normal): the probabilities of all kinds of vehicles are the same. The table 16 shows the probability matrix on page 23.

↗	r_0	r_1	r_2	r_3
r_0	0	0.2	0.2	0.2
r_1	0.2	0	0.2	0.2
r_2	0.2	0.2	0	0.2
r_3	0.2	0.2	0.2	0

Table 16: Normal

2. E(east): the probability of vehicles to one road, say r_0, is much more than that of any other road. The table 17 shows the probability matrix.

↗	r_0	r_1	r_2	r_3
r_0	0	0.2	0.2	0.2
r_1	0.6	0	0.2	0.2
r_2	0.6	0.2	0	0.2
r_3	0.6	0.2	0.2	0

Table 17: East

3. SW(southwest): the probability of vehicles to two nearby roads, say r_0, r_1, is much more than that of the other two roads. The table 18 shows the probability matrix.

↗	r_0	r_1	r_2	r_3
r_0	0	0.2	0.4	0.4
r_1	0.2	0	0.4	0.4
r_2	0.2	0	0	0.2
r_3	0	0.2	0.2	0

Table 18: Southwest

Table 19 on page 24 shows the AWT of the method in chapter 10. The unit of data in the table is one TCS.

11 Comparison between Different Traffic Circles

After discussing several kinds of traffic circles, the result can be summarized into three points.

	N	E	SW
AWT	0.50	46.2	6.47

Table 19: Result

1. AWT and MWT of optimal design increase faster than the probability of the appearance of vehicles. The bigger the probability, the faster AWT and MWT increases, which implies that the efficiency of a traffic circle depends sensitively on the tense of the traffic flow.
2. AWT and MWT decrease dramatically when there is one more lane, which implies that the influence by adding a new lane to the circle is remarkable. So if it is possible, we recommend the government to build more lanes.
3. It is highly recommended to use the buffering design if the traffic circle has already two lanes.

Finally, we have got the criteria for whether to set proper traffic rules or rebuild the traffic circle which is stated here and in chapter 8 and chapter 9.

12 Strengths

- The simulation programme can be applied to a variety of situations.
- We use two metrics to judge a method and choose the most efficient ones without losing fairness.
- Various situations are discussed to extend to common conclusions.
- We have strictly proved that traffic lights is not a necessity, a statement also verified by the stimulation programme.
- The buffering design is easier to understand and more practical.

13 Weakness

- Some of the methods are a bit difficult to remember for drivers.
- The patterns of the traffic flow discussed are a little bit limited, though several main situations have been involved. It can not be ensured that a more complicated pattern will occur in real life.

- In the model that allows vehicles to change between lanes, we just come up with a solution named the buffering design. Though it is strong enough to handle almost every situation of a traffic circle which has two lanes. Still, we did not step further to construct a stronger design which involves the ideas in the buffering one.

Part V
Conclusion

In this paper, we, firstly, have put forward a model for the traffic flow and imitated it by a Monte Carlo algorithm. This model is a good approximation to the real traffic flow in life. Based on this model, we come up with two metrics, which measures the efficiency and the fairness of a traffic circle design. Also, two designs, i.e, classified-lane design and buffering design, are provided to be choose. Then for every design, three patterns of traffic flow are used to test every methods we take into account to solve this problem. Thus, through this process, we can sift the best design out. Finally, according to the metrics data of each design calculated by the algorithm, we give a specific and detailed guide to realize our methods.

A Decoding Method

In a four-road and two-lane situation, we encode every classified-lane design into a distinct number. For convenient, we consider the circle consists of 4 arcs. If the design declares that a vehicle covering only one arc in the circle has to drive in the outer lane and others the inner can be converted into a quaternary number 1111. The number 1 in the lowest position means the vehicle from road 0 should enter the outer lane if it is going to exit at the road 2 or 3 or 4. And the number in other position would be interpreted in the same way. Therefore the decimal number 85 is the code of this design. Decoding steps can be done in a reverse direction of the encoding.

For the three-lane situation, we encode a specific design into a pair of numbers, for example 85+170 in decimal system or 1111+2222 in quaternary system. It means that the vehicle from r_0 to r_1 shall enter the outmost lane and to r_2 the middle and r_3 the inmost.

In a similar way, a design for a five-road traffic circle can be converted into a quinary number or a pair of quinary numbers.

B Source Code

The programme for the classified-lane design.

```
#include <stdio.h>
#include <stdlib.h>
#define N 2
#define CL 20
#define M 5
#define MM 3125
#define T 500
#define C 1
#define X (0.1 * C)
#define Y (0.2 * C)
#define Z (0.3 * C)
int main() {
    int i, j, k, t, temp, sa, c, sc, state, tstate;
    int s[N][CL];
    const int cl[N] = {15, 20};
    const int tc[N][M] =
                {
                    0, 3, 6, 9, 12,
                    0, 4, 8, 12, 16
                };
    int st[M][M] =
                {
                };
    int q[M][T], qq[M], cc[M];
    double dtemp;
    double sm = 1000;
    const double p[M][M] =
                {
                    0, X, 0, 0, X,
                    Y, 0, 0, 0, X,
                    Z, X, 0, 0, X,
                    Z, X, 0, 0, X,
                    Y, X, 0, 0, 0
                };
    for (state = 0; state < MM; state++)
    {
    for (i = 0; i < M; i++)
```

```
{
    for (j = 0; j < M; j++)
        st[i][j] = 0;
    qq[i] = cc[i] = 0;
}
for (i = 0; i < M; i++)
{
    for (j = tstate % M; j > 0; j--)
        st[i][(i + j) % M]++;
    tstate /= M;
}
t = sa = c = 0;
for (i = 0; i < M; i++)
    for (j = 0; j < T; j++)
    {
        dtemp = (double)rand() / RAND_MAX;
        for (k = 0; k < M; k++)
            if (dtemp < p[i][k]) break;
            else dtemp -= p[i][k];
        if (k < M)
        {
            q[i][j] = k;
            cc[i]++;
            c++;
        }
        else q[i][j] = -1;
    }
for (i = 0; i < N; i++)
    for (j = 0; j < cl[i]; j++)
        s[i][j] = rand() % (M + 1) - 1;
sc = c
while (c)
{
    for (j = 0; j < M; j++)
    {
        for (i = N - 1; i >= 0; i--)
            if (s[i][tc[i][j]] == j)
            {
                s[i][tc[i][j]] = -1;
                for (k = i + 1; k < N; k++)
                    if (s[k][tc[k][j]] != -1) sa++;
```

```
            }
        if (q[j][0] != -1 &&
            s[st[j][q[j][0]]][tc[st[j][q[j][0]]][j]] == -1)
        {
            s[st[j][q[j][0]]][tc[st[j][q[j][0]]][j]]
            = q[j][0];
            q[j][0] = -1;
            c--;
        }
        if (q[j][0] != -1)
        {
            sa++;
            qq[j]++;
        }
        for (k = 1; k < T; k++)
            if (q[j][k] != -1)
                if (q[j][k - 1] == -1)
                {
                    q[j][k - 1] = q[j][k];
                    q[j][k] = -1;
                }
                else
                {
                    sa++;
                    qq[j]++;
                }
    }
    for (j = 0; j < N; j++)
    {
        temp = s[j][cl[j] - 1];
        for (k = cl[j] - 1; k > 0; k--)
            s[j][k] = s[j][k - 1];
        s[j][0] = temp;
    }
    t++;
};
if ((double)sa / sc < sm)
{
    sm = (double)sa / sc;
    printf("%4d  ", state);
    printf("%3.2f  ", (double)sa / sc);
```

```
        dtemp = 0;
        for (i = 0; i < M; i++)
            if (dtemp < (double)qq[i] / cc[i])
                dtemp = (double)qq[i] / cc[i];
        printf("%3.2f\n", dtemp);
    }
    }
    return 0;
}
```

The programme for the buffering design.

```
#include<stdio.h>
#include<stdlib.h>

#define length 1000
#define ccc1 4 //ccc1 = 1/4 circumference inside

int roadl[4][length+1]={0},roadr[4][length+1]={0},
    circle1[4*ccc1]={0}, wait,car[4],carnumber=0,carnum[4]={0},
    // the matrix of probability
    prob[4][4]={0,20,20,20,
                20,0,20,20,
                20,20,0,20,
                20,20,20,0};
double waitpercar;

void initial () {
    int i,j,p,pro[4];
    wait=0;
    for (i=0;i<4;i++) {
        pro[0]=prob[i][0];
        pro[1]=prob[i][0]+prob[i][1];
        pro[2]=prob[i][0]+prob[i][1]+prob[i][2];
        pro[3]=prob[i][0]+prob[i][1]+prob[i][2]+prob[i][3];

        for (j=0;j<length;j++) {
            p=rand()%100;
            if (p<pro[0]) {
                if (i==3) roadr[i][j]=1;
                else roadl[i][j]=1;
```

```
                carnum[i]++;
            }
            else if (p<pro[1]) {
                if (i==0) roadr[i][j]=2;
                else roadl[i][j]=2;
                carnum[i]++;

            }
            else if (p<pro[2]) {
                if (i==1) roadr[i][j]=3;
                else roadl[i][j]=3;
                carnum[i]++;
            }
            else if (p<pro[3]) {
                if (i==2) roadr[i][j]=4;
                else roadl[i][j]=4;
                carnum[i]++;
            }
            else {roadl[i][j]=0;}
        }
    }
    carnumber=carnum[0]+carnum[1]+carnum[2]+carnum[3];
}

int inroad (int i) {
    int j=0,k,n=0;
    if (carnum[i]==0) return 0;

    if ((circle1[ccc1*i]==0)&&(roadl[i][0]!=0)) {
            circle1[ccc1*i]=roadl[i][0];
            roadl[i][0]=0;
            carnum[i]--;
            carnumber--;

    }
    else if (roadr[i][0]!=0) {
        roadr[i][0]=0;
        carnum[i]--;
        carnumber--;
    }
```

```
    for (j=0;j<length;j++) {
        if (roadl[i][j]==0) break;
        car[i]++;
        wait++;
    }
    if (j<length) {
        for (k=j;k<length;k++)
            roadl[i][k]=roadl[i][k+1];
    }

    for (n=0;n<length;n++) {
        if (roadr[i][n]==0) break;
        car[i]++;
        wait++;
    }
    if (n<length) {
        for (k=n;k<length;k++)
            roadr[i][k]=roadr[i][k+1];
    }
    return 0;
}

int incir () {
    int i;
    for (i=0;i<4;i++) {
        inroad(i);
    }
    return 0;
}

int outcir () {
    int i,m;

    m=circle1[4*ccc1-1];
    for (i=4*ccc1-1;i>0;i--) {
        circle1[i]=circle1[i-1];
    }
    circle1[0]=circle1[4*ccc1-1];
    for (i=0;i<4;i++) {
        if (circle1[i*ccc1]==(i+2)%4)
            circle1[i*ccc1]=0;
```

```
    }
    if (circle1[2*ccc1]==4) circle1[2*ccc1]=0;
    return 0;
}

int main () {
    int n;
    initial();
    n=carnumber;
    while (carnumber>0) {
        outcir();
        incir();
    }

    waitpercar=1.00*wait/n;
    printf("car=%d\nwait=%d\nwait per car=%f\n",n,wait,waitpercar);

    return 0;
}
```

C Technical Summary

We use the symbols denoted in chapter 3.2 for convenience. Moreover,$r_i \to r_j$ refers to vehicles from r_i to r_j and "all" refer to all roads.

Four Roads Case

1. Two Lanes:
   (a) If all roads are equal
   not busy: we denote the inmost lane as c_0 and outmost c_1.
   $c_1 : r_0 \to r_1; r_1, r_2 \to all.c_0 : else$
   busy: $c_0 : r_1, r_3 \to all.c_1 : else$
   (b) If more vehicles rush to one road, say r_0
   not busy: $c_0 : r_0, r_2 \to all.c_1 : else$
   busy: $c_0 : r_1, r_3 \to all.c_1 : else$
   (c) If more vehicles rush to two roads,say r_2, r_3
   not busy: $c_0 : r_0, r_2 \to all; r_3 \to r_2.c_1 : else$
   busy: $c_0 : r_0 \to all; r_2 \to r_0, r_1; r_3 \to r_2.c_1 : else.$
2. Three lane:
   (a) If all roads are equal
   not busy: $c_0 : r_2 \to all; r_3 \to r_2.c_1 : r_1 \to all; r_3 \to r_0, r_1.c_2 : r_0 \to all.$
   busy: $c_0 : r_2 \to all; r_3 \to r_2.c_1 : r_1 \to all.c_2 : r_0 \to all; r_3 \to r_0, r_1.$
   Extremely busy: $c_0 : r_3 \to all.c_1 : r_1 \to r_0, r_3; r_2 \to all.c_2 : r_0 \to all; r_1 \to r_2.$
   (b) If more vehicles rush to one road, say r_0
   $c_0 : r_2 \to all; c_3 \to c_2.c_1 : r_0 \to r_2, r_3; r_3 \to r_0, r_1.c_2 : r_0 \to r_1; r_1 \to all.$
   (c) If more vehicles rush to two nearby roads, say r_2, r_3
   not busy: $c_0 : r_2, r_3 \to all.c_1 : r_0 \to r_2, r_3.c_2 : r_0 \to r_1; r_1 \to r_2, r_3.$
   busy: $c_0 : r_2, r_3 \to all.c_1 : r_0 \to r_2, r_3.c_2 : r_0 \to r_1; r_1 \to r_2, r_3.$

Five Roads Case

1. Two lanes:

(a) If all roads are equal
not busy: $c_0 : else.c_1 : r_0 \to r_1, r_2, r_3; r_1 \to all; r_2 \to r_3, r_4; r_3 \to r_4, r_0, r_1; r_4 \to r_0, r_1$.
busy: $c_0 : else.c_1 : r_0 \to r_1, r_2, r_3; r_1 \to all; r_2 \to r_3; r_3 \to all$.

(b) If more vehicles rush to one road, say r_0
not busy: $c_0 : else.c_1 : r_0 \to all; r_1 \to r_2, r_3, r_4; r_2 \to all; r_4 \to r_0, r_1, r_2$.
busy: $c_0 : r_1 \to r_0; r_2 \to all; r_3 \to r_2; r_4 \to r_3.c_1 : else$.

(c) If more vehicles rush to two road, say r_0, r_1
not busy: $c_0 : r_2 \to r_1; r_3 \to all; r_4 \to r_3.c_1 : else$.
busy: $c_0 : r_0 \to r_4; r_1 \to all; r_2 \to r_1; r_3 \to r_2; r_4 \to all.c_1 : else$.

2. Three lanes:

(a) If all are equal
not busy: $c_0 : r_3 \to all; r_4 \to r_2, r_3.c_1 : r_0 \to r_4; r_1 \to all; r_4 \to r_0, r_1.c_2 : else$.
busy: $c_0 : r_3 \to r_2; r_4 \to all.c_1 : r_0 \to r_4; r_2 \to all.c_2 : else$.

(b) If more vehicles rush to one road, say r_0
not busy: $c_0 : r_2 \to r_1; r_3 \to all.c_1 : r_0 \to r_2, r_3, r_4; r_1 \to r_0, r_4; r_4 \to all.c_2 : else$.
busy: $c_0 : r_3 \to all; r_4 \to r_3.c_1 : r_0 \to all; r_1 \to r_0, r_3, r_4; r_4 \to r_0, r_1, r_2.c_2 : else$.

(c) If more vehicles rush to two nearby road, say r_0, r_1
not busy: $c_0 : r_3 \to all; r_4 \to r_2, r_3.c_1 : r_0 \to r_4; r_1 \to r_0, r_3, r_4; r_4 \to r_0, r_1.c_2 : else$.
busy: $c_0 : r_3 \to r_0, r_1, r_2.c_1 : r_0 \to r_3, r_4; r_1 \to r_0, r_4; r_4 \to all.c_2 : else$.