Omar Farooq, Hunter Lambert, Tyler Barnett, Hans-Peter Hoellwirth
[revised April-23-2015]

Traffic Flow Simulator: Requirements Document

1. Background

A team of traffic engineers wants to use simulation software to gain intuition about traffic congestion. They are looking for simulation software that provides information on how traffic lights and stop signs impact congestion. The client's main interest is to find the median time it takes for a car to get through the city. The software will give the traffic engineers a good understanding of the practical and essential use of traffic lights and stop signs.

2. Description of Software

The purpose of the software is to simulate traffic flow in a city grid regulated by sensor-driven traffic lights. The simulation feeds the city roads with cars randomly and determines the median time it takes for cars to go through the grid.

3. Environmental Characteristics

Not yet specified by client.

4. Simulation Characteristics

4.1 Geographic layout

4.1.1 The city section is a square grid of between 2 to 10 north-south two-way streets and the same number of east-west two-way streets. The size of the grid is named numIntersectionsInOneDirection.

4.1.2 Every intersection is connected to 8 segments.

4.1.2.1 Every intersection is connected to 4 segments from which cars enter the intersection.

4.1.2.1.1 Every intersection is connected to a segment whose direction is north from which cars enter the intersection.

4.1.2.1.2 Every intersection is connected to a segment whose direction is south from which cars enter the intersection.

4.1.2.1.3 Every intersection is connected to a segment whose direction is east from which cars enter the intersection.

4.1.2.1.4 Every intersection is connected to a segment whose direction is west from which cars enter the intersection.

4.1.2.2 Every intersection is connected to 4 segments from which cars exit the intersection.

4.1.2.2.1 Every intersection is connected to a segment whose direction is north from which cars exit the intersection.

4.1.2.2.2 Every intersection is connected to a segment whose direction is south from which cars exit the intersection.

4.1.2.2.3 Every intersection is connected to a segment whose direction is east from which cars exit the intersection.

4.1.2.2.4 Every intersection is connected to a segment whose direction is west from which cars exit the intersection.

4.1.3 Each segment is comprised of exactly one queue.

4.1.4 No separate queues for left- or right-turning cars exist.

4.1.5 Every segment has the same maximum capacity of cars that it can hold at any point of time. The maximum capacity in the simulation is named maxSegmentCapacity.

4.2 Traffic control devices

4.2.1 Each intersection is controlled by either exactly two traffic lights or four stop signs.

4.2.2 Traffic lights at intersections

4.2.2.1 [revised April-21-2015] Each intersection controlled by traffic lights has exactly one traffic light.

4.2.2.2 [revised April-22-2015] Each traffic light has 4 mutually exclusive states:

4.2.2.2.1 All Red after Green in North-South Direction (ALL_RED_AFTER_GREEN_NS)

4.2.2.2.2 All Red after Green in East-West Direction (ALL_RED_AFTER_GREEN_EW)

4.2.2.2.3 Green in North-South Direction (GREEN_NS)

4.2.2.2.4 Green in East-West Direction (GREEN_EW)

4.2.2.3 The traffic light at each intersection acts independently from the traffic light at other intersections.

4.2.2.4 If for a given intersection the traffic light is green in the north-south direction, it must be red in the East-west direction.

4.2.2.5 If for a given intersection the traffic light is green in the East-West direction, then it must be red in the North-South direction.

4.2.2.6 Traffic lights in the north-south direction are green at the beginning of the simulation.

4.2.2.7 No separate traffic lights for left or right turns exist.

4.2.2.8 Sensors at intersections

4.2.2.8.1 For a given intersection, at each segment from which cars enter the intersection, a sensor determines whether or not there is a car in that segment's queue.

4.2.2.8.2 Each sensor only measures whether there is a car waiting or not, but not the number of cars waiting in the queue.

4.2.2.8.3 [revised April-22-2015] When a car is sensed at a red light:

4.2.1.8.3.1 The traffic light at that intersection turns to all red (from GREEN_NS to ALL_RED_AFTER_GREEN_NS

and from GREEN_EW to ALL_RED_AFTER_GREEN_EW) after minTimeLightGreen has been reached for this traffic light and immediately if minTimeLightGreen has been already exceeded at that point of time.

4.2.1.8.3.2 After the traffic light has turned to all red, the traffic light will stay in this status for exactly minTimeToChangeLight time units and then switch from ALL_RED_AFTER_GREEN_NS to GREEN_EW or from ALL_RED_AFTER_GREEN_EW to GREEN_NS.

4.2.2.8.4 A traffic light will not switch for any reason other than a waiting car has been sensed at a red light.

4.2.3 Implicit control devices at entrance to city section

4.2.3.1 Each entrance is controlled by a regular traffic light.

4.2.3.2 The entering segments are not restricted by the segment capacity constraint.

4.2.4 Stop signs at intersections

Not yet specified by client.

4.3 Traffic flow

4.3.1 Speed of traffic

4.3.1.1 All cars move at the same speed when they move.

4.3.1.2 A car traverses a segment in no less than minTimeToTraverseSegment.

4.3.2 Patterns of traffic flow

4.3.2.1 There are no collisions.

4.3.2.2 A car can not overtake another car.

4.3.2.3 [revised April-15-2015] There can be at most one car in an intersection at a time.

4.3.2.3.1 There is no “right of way” apart from the set order in which incoming segments are processed by an intersection. (See 4.7.3)

4.3.2.4 [revised April-15-2015] A car can make a right turn only under the following conditions:

4.3.2.4.1 A car can make a right turn on a green light only when:

4.3.2.4.1.1 There is no car in the intersection.

4.3.2.4.1.2 The segment to enter is not full.

4.3.2.4.2 A car can make a right turn on a red light only when:

4.3.2.4.2.1 There is no car in the intersection.

4.3.2.4.2.2 The segment to enter is not full.

4.3.2.4.2.3 There is a green light in the direction perpendicular to the car.

4.3.2.5 [revised April-15-2015] A car can make a left turn only under the follwing conditions.

4.3.2.5.1 The traffic light is green.

4.3.2.5.2 There is no car in the intersection.
4.3.2.5.3 The segment to enter is not full.
4.3.2.6 [revised April-15-2015] A car can go straight only under the following conditions:
4.3.2.6.1 The traffic light is green.
4.3.2.6.2 There is no car in the intersection.
4.3.2.6.3 The segment to enter is not full.

4.4 Stochastic properties

4.4.1 Random numbers are used to determine if a car enters at a given time unit.
4.4.2 The probability of a car entering the grid per time unit is constant and is given by the variable probabilityOfArrivalPerIteration.
4.4.3 The portion of the grid the car enters and exits from is determined randomly before the car enters the grid.
4.4.4 The user gets different results from the simulation each time yet is able to run the exact same simulation with the same results again if he/she wants by providing the idNumber of the earlier simulation.

4.5 Properties of cars

4.5.1 When entering the grid, each car already knows its destination (that is the exit segment from the grid) and its path to the destination.
4.5.2 A car does not change its initially planned route in case of traffic congestions.
4.5.3 A car does not exit the same direction that it entered.
4.5.4 Each car only turns at most once (once or not at all).
4.5.5 If a car is going to exit the opposite side of the grid from which it entered, the car stays on the same road and so exits on the road it entered from.

4.6 Additional assumptions

4.6.1 The following elapsed time is insignificant: the time required for two cars at the same intersection that are traveling in opposite directions to both change their directions by making left turns within the same step of the simulation.

4.7 [revised April-08-2015] **Order of "Execution"**

4.7.1 The intersections and segment are processed in the same way every time.
4.7.2 The intersections are processed with column number increasing faster than the row number. That is, after intersection (1,1) we visit (2,1),(3,1) and so on.
4.7.3 The incoming segments are processed first in the order southward, eastward, northward, westward, then the outgoing segments are processed in the same order.

5. Input/Output

5.1 Input

5.1.1 Input Format

5.1.1.1 There is a prompt for input parameters for the simulation.

5.1.2 [revised April-22-2015] Initial Input Parameters

5.1.2.1 [revised April-23-2015] *idNumber*: gives the simulation identification number of a previous simulation and forces the new simulation to use the same distinct random seed, except for the case where the value is -1. If -1, the simulation should use a new random seed.

5.1.2.2 *lengthOfSimulation*: desired length of simulation in time units. The number has to be greater than 0 and smaller than 60,000.

5.1.2.3 *numIntersectionsInOneDirection*: gives the dimension of the square grid, ranging from 2 to 10 inclusive.

5.1.2.4 *beginMarking*: gives the number of time units after start the of the simulation before starting to mark entering cars. The number has to be greater than 0 and smaller than *lengthOfSimulation.*

5.1.2.5 *endMarking*: gives the number of time units after the start of the simulation before stopping to mark entering cars. The number has to be greater than *beginMarking* and smaller than *lengthOfSimulation*.

5.1.2.6 *maxSegmentCapacity*: gives the maximum number of cars in one segment at a single point of time The number has to be greater than 0 and smaller than 100.

5.1.2.7 *minTimeLightGreen*: gives the minimum number of time units that a traffic light must be green before switching to all red. The number has to be greater than 0 and smaller than *lengthOfSimulation*.

5.1.2.8 [revised April-08-2015] *minTimeToChangeLight*: gives the number of time units a traffic light stays in status all red. The number has to be greater than 0 and smaller than *lengthOfSimulation*.

5.1.2.9 *minTimeToTravelSegment*: gives the number of time units it takes a car to traverse a segment. The number has to be greater than 0 and smaller than *lengthOfSimulation*.

5.1.2.10 [revised April-08-2015] *minTimeToChangeSegment*: gives the number of time units it takes a car between entering an intersection from one segment and exiting theintersection to go into a different segment. The number has to be greater than 0 and smaller than *lengthOfSimulation*.

5.1.2.11 *choiceOfThreeOptions*: defines the list of allowed turns and must be between 1 and 3.

If set to 1, you want each car to only go straight.

If set to 2, you want each car never to turn left, and either go only straight or to turn right exactly once.

If set to 3, you want each car to only go straight or to turn right exactly once or

to turn left exactly once.

5.1.2.12 *probabilityOfCreatedCarAlwaysMovingStraight*: gives the the probability of a created car always moving straight. The probabilty must be 1.0 if *choiceOfThreeOptions* is 1.

5.1.2.13 *probabilityOfArrivalPerIteration*: gives the the probability that a car enters the grid in a particular time unit has to be in the range 0 to 1 inclusive.

5.2 Output

5.2.1 Number of cars that were marked.

5.2.2 [revised April-23-2015] Median time of marked cars within the grid.

5.2.2.1 Each car has an elapsed time that it spends in the grid.

5.2.2.2 [revised April-23-2015] The software tracks the time units each car spends within the grid, that is, the number of time units between entering the grid and the moment when the car is put into an exit segment.

5.2.2.3 The calculation of the median time of cars in the grid only considers those cars that enter the grid starting at iteration number beginMarking and before iteration number endMarking. The simulation refers to those cars as “marked cars”.

5.2.2.4 After the simulation has stopped, the system determines the median time that marked cars were in the grid.

6. Life Cycle Considerations

6.1 [revised April-15-2015] **Subsets**

6.1.1 The first deliverable implemented the Car class. The program allowed the user to construct instances of Car.

6.1.2 The second deliverable enhanced the first deliverable by implementing the Segment class and allowing the user to construct instances of Segment.

6.1.3 The third deliverable enhanced the second deliverable by implementing the Intersection class and allowing the user to construct instances of Intersection.

6.1.4 The fourth deliverable enhanced the third deliverable by assigning Segments to Intersections. There were no Cars put in Segments in this subset. This subset checked the top-level logic of the traffic simulation.

6.1.5 The fifth deliverable enhanced the fourth by introducing Cars in Segments in a restricted way. Cars were only placed in the incoming Northward segment of intersection which had the same column and row number.

6.1.6 The sixth deliverable enhanced the fifth by implementing turns. This subset also allowed cars to leave the grid.

6.1.7 The seventh deliverable allowed cars to be placed in specific locations in the grid. This subset also checked if the segment was full before placing a car into that segment. The segment capacity was set to 3. However, cars were only allowed to move straight.

6.1.8 The eighth deliverable expanded the simulation by encorporating minTimeToChangeSegment and minTimeToTraverseSegment.

6.1.9 The ninth deliverable will expand the simulation by incorporating maxSegmentCapacity.

6.1.10 The tenth deliverable will enhance the ninth deliverable by incorporating traffic lights for each intersection. Cars will only go straight.

6.1.11 The eleventh deliverable will enhance the tenth deliverable by incorporating cars that will turn right (11a) and left (11b).

6.1.12 The twelfth deliverable will enhance the eleventh deliverable by incorporating randomization and calculating the median time of marked cars in the grid.

6.2 Possible extensions

6.2.1 Adding stop signs

6.2.2 Adding graphical interface to see how many cars are on the grid and where.

6.2.3 Adding a real-time visualization of the traffic which will allow the user to see cars moving and traffic signals changing.

6.2.4 Adding support for configuration file input for simulation parameters

7. Glossary

7.1 A “queue” is a sequence of cars awaiting their turn to enter an intersection. The cars enter the intersection in order of their queue arrival.

7.2 A "segment" connects two intersections in a single direction and consists of exactly one queue.

7.3 A “street section” consists of two segments, one of which goes in the opposite direction of the other segment, and thus connects two intersections in both directions.

7.4 A “street” consists of one or more connected street sections, all of them either going north-south or east-west.

7.5 “Grid” is the system of streets that the simulation is run on.

7.6 An “intersection” is a place where a north-south two way street and a east-west two way street intersect.

7.7 “Direction” is the course in which a car can move in a segment and can be northward, southward, eastward or westward.

7.8 “Opposite direction” describes a direction in terms of another direction: the opposite direction of northward is southward, of southward is northward, of eastward is westward, and of westward is eastward.

7.9 “Cross direction” describes a street in terms of a direction: the cross direction of northward is eastward-westward and of southward is eastward-westward, and the cross direction of eastward is northward-southward, and of westward is northward-southward.

7.10 A “marked car” is a car that has entered the simulation at or after beginMarking and before endMarking.

7.11 A car being “at a red light” means that:

7.11.1 The car is in a segment from which cars enter an intersection

7.11.2 There are no other cars ahead of the car in the segment

7.11.3 The car has been in the segment for at least minTimeToTraverseSegment

7.11.4 [revised April-23-2015] The traffic light in the direction of the segment is red, meaning that the traffic light is in either of the following status:

7.11.4.1 ALL_RED_AFTER_GREEN_NS or ALL_RED_AFTER_GREEN_EW

7.11.4.2 GREEN_NS and car is in a segment with directition eastward or westward

7.11.4.3 GREEN_EW and car is in a segment with directition northward or southward

7.12 A car being “at a green light” means that:

7.12.1 The car is in a segment from which cars enter an intersection

7.12.2 There are no other cars ahead of the car in the segment

7.12.3 The car has been in the segment for at least minTimeToTraverseSegment

7.12.4 [revised April-23-2015] The traffic light in the direction of the segment is green, meaning that the traffic light is in either of the following status:

7.12.4.1 GREEN_EW and car is in a segment with directition eastward or westward

7.12.4.2 GREEN_NS and car is in a segment with directition northward or southward

7.13 “Time unit” refers to one iteration and is the smallest unit of time used in the simulation. It is used for both inputs and outputs.

7.14 “Point of time” is the number of elapsed time units since the start of the simulation.

7.15 A car is “in the intersection” if the car has exited the segment from which it came and is waiting to enter the segment to which it is going.

7.16 [revised April-22-2015] The “median” of all of the marked cars’ times in the grid is calculated as follows: If the number of marked cars is odd, the “median” is the middle element of the sorted marked cars’ times in the grid. If the number of marked cars is even, the “median” is the average of the two elements closest to the middle of the sorted list of marked cars’ times in the grid.

7.17 [revised April-22-2015] A segment is said to be “full” if the current number of cars in this segment equals the maximum number of cars allowed in this segment (defined as *maxSegmentCapacity*).

7.18 [revised April-22-2015] Names of values:

Name of Value	**Intended Meaning**
idNumber	simulation identification number related to random seed

lengthOfSimulation	length of simulation in time units
numIntersectionsInOneDirection	the dimension of the grid, ranging from 2 to 10
beginMarking	number of time units to start marking entering cars
endMarking	number of time units to stop marking entering cars
maxSegmentCapacity	maximum number of cars in one segment
minTimeLightGreen	minimum number of time units that a traffic light must be green
minTimeToChangeLight	number of time units between a green traffic light turning red and the red light in the cross direction turning green.
minTimeToTravelSegment	minimum number of time units for one car to traverse one segment
minTimeToChangeSegment	the minimum number of time units for a car to move to a new segment given that the car can enter the new segment.
choiceOfThreeOptions	option code (1 to 3): [1] You want each car to only go straight. [2] You want each car never to turn left, and either to only go straight. [3] You want each car to only go straight or to turn right exactly once or to turn left.
probabilityOfCreatedCarAlways MovingStraight	probability that a created car always moves straight
probabilityOfArrivalPerIteration	probability that a car enters the grid in a particular time unit

8. Questions

No questions to the client.