Comet Backup

Practical Candidate development skills

This test is designed to give flexibility for you to demonstrate your skills in an unrestricted way. It is open-book and you may use any resources you with to reference, however your work must be your own. Please don't copy and paste any code.

You may use any language to complete the following tasks. However, we would prefer submissions in Go/PHP/Javascript or C++. Build Scripts, makefiles, data files are also allowed.

Time allowance 2.0 hours from opening this document

Please submit your application as a git repository or zipped directory to hello@cometbackup.com

Task

You have been employed by the City Council to determine the most suitable type of traffic control to build, for an intersection where two roads cross.

Each of the 4 intersecting roads has a Cars-Per-Minute (CPM) rate variable. Your program should allow the following variable inputs:

- Road 1 (North) CPM
- Road 2 (East) CPM
- Road 3 (South) CPM
- Road 4 (West) CPM

There are three different control methods that can be used. There are different reasons why each control method might be used.

- Roundabout
    - Free flowing at low CPM, but at higher CPMs, there is extra congestion and inefficiency
- Stop signs
    - No waiting if the intersection is clear, but all users must stop, so it is inefficient at low CPM
- Traffic lights
    - Efficient at high CPM, but users may have to wait, even at low CPM

At this stage, the City Council do not have enough information to determine the most efficient parameters for a given road intersection. Each control method will perform differently, depending on the total CPM of the 4 roads.

Therefore, you will also need to factor in the following constants. The table below shows the performance of each type of intersection at different total CPM levels. A high efficiency percentage means high performance. A low efficiency percentage means the control method is unnecessarily impeding car movements.

Control Method	"High Throughput" (Total CPM >= 20)	"Medium Throughput" (20 > Total CPM >= 10)	"Low Throughput" (Total CPM < 10)
Roundabout	50% efficient	75% efficient	90% efficient
Stop Signs	20% efficient	30% efficient	40% efficient
Traffic Lights	90% efficient	75% efficient	30% efficient

Your program should come up with an "efficiency score" for the intersection based on the given CPMs to choose the best control method.

It is recommended that you start out by hardcoding some values for these (e.g., an intersection with 5 CPM in each direction), and focus on a working solution. Afterwards, you can work on the extension activities.

Extensions

Here are some ideas to extend the project. If you have time, you can work on these in any order you prefer.

1. The Council would like to be able to manipulate the CPM values for each four roads without recompiling every time. An interactive prompt would be helpful. You may choose to prompt for input values on stdin, or accept them as command-line parameters.
2. You might want to use visual formatting to make it more readable and visually appealing, or implement a GUI interface using Qt. If your program is a command-line program, you could add support for a "/?" or "--help" or "--usage" command-line flag.
3. You might want to implement a batch system that produces the result for a large number of CPM values, read from a CSV file on disk or stdin.
4. You might want to extend the system to support arbitrary traffic control methods.
5. Roundabouts tend to be more efficient when one through-road is busy, and the other is quiet. So, if roads 1 and 3 have a high CPM relative to 2 and 4, then a bonus efficiency score of +10% might be earned for the roundabout control method.
6. You might have noticed that stop signs never get the best score. However, they are very cheap to make. Your program could allow the user to enter costs of each of the 3 control methods. You can then output a CPM-per-dollar value, indicating the number of cars the intersection can support, per dollar of construction cost. Roundabouts cost $100k, stop signs cost $40k and traffic lights cost $200k.
7. Rather than three static states of throughput (low, medium, high), could these states be a continuous curve? Modify the algorithm to smoothly transition between these states.
8. Add support for dual carriageway and/or pedestrian lanes. Can you allow a variable number of lanes in each direction?
9. Traffic lights can allow north-to-south and south-to-north car movement at the same time. What are all the other path combinations? What is the smallest covering set of paths that can be travelled simultaneously? Explore the maximum efficiency when each path combination has a different CPM rate.
10. Each of the 3 control methods has a different average latency before a new car is able to pass. At high throughput levels, traffic lights are more efficient, but a car might be able to

enter an empty roundabout sooner. Invent some realistic latency numbers for each control method, and then investigate the latency-vs-throughput curve. What is the effect of cars “piling up” on average latency? Simulate this.