5-2005

Modeling Traffic Patterns using Java

Kim Meek
The College at Brockport

Follow this and additional works at: http://digitalcommons.brockport.edu/cmst_lessonplans

Part of the Physical Sciences and Mathematics Commons, and the Science and Mathematics Education Commons

Repository Citation
http://digitalcommons.brockport.edu/cmst_lessonplans/284

This Lesson Plan is brought to you for free and open access by the CMST Institute at Digital Commons @Brockport. It has been accepted for inclusion in Lesson Plans by an authorized administrator of Digital Commons @Brockport. For more information, please contact kmyers@brockport.edu.
Modeling Twelve-Corners Traffic
Project Background

Many students must pass a crosswalk across Winton Rd near Twelve Corners in order to reach Brighton High School each day.

A trusty crossing guard ensures their safety by operating a traffic light that stops cars so that students can safely cross.

Each time the light turns red, all road traffic stops, and cars build up at the intersection.
Problem:

Can the timing of the crosswalk’s traffic light be improved to minimize traffic congestion while still ensuring the safety of the students?
Data Collection

Team members BW and DG observed pedestrian and motor vehicle traffic at the crosswalk for 10 minutes during peak traffic time. And collected the following data:

- # of Cars: 53
- # of students: 17
- Time light was red: 30 seconds
- Average time light was green: 2 minutes
Computer Technology Chosen: Java programming

- All three team members are AP Computer Science students, and have a strong understanding of the Java language.
- Java’s object-oriented nature allows for classes to be designed for each type of object that interacts at the crosswalk.
- Java’s Marine Biology Case Study, part of the AP Computer Science Curriculum, provides an excellent framework to simulate the crosswalk.
The Marine Biology Case Study (MBS)

- Provides a simulation of fish randomly moving about an environment.
- Fish can be added to any square and facing any direction.
- Different speeds of simulation can be chosen, or it can proceed through single steps.
Important MBS Classes

Fish – The code for the fish in the simulation. Contains an act method that tells the fish what to do in each step of the program.

Environment – Provides a grid for objects in the simulation. Only one object can inhabit each square. Only fish objects can be added.

MBSGUI – Provides a Graphics User Interface for the simulation. Allows for easy tweaking of the simulation as well as a way to view it.
New Classes we Added

Since only Fish can be added to environment, we faced a problem. Fortunately, Java offered a solution. We made all our new classes extend Fish so that they could be used.

By overriding the act method in these classes, we could also design specific behaviors for all of them.
Class Descriptions: Car

- Detects whether it is in the westbound or eastbound lane and moves accordingly.
- If the light is red, the Car stops before the crosswalk.
- Even if light is green, if a child is in the way, the Car will stop.
Children

Moves on sidewalk towards crossing guard.  
If light is green, waits in line for light to turn red.  
Once light turns red, runs across the crosswalk and continues to walk on other side of road.
CrossingGuard

Can be seen as “God” within the simulation.
Adds new cars and children randomly.
Has a probability of turning light red based on how many kids are waiting.
Once light is red, turns light green after a number of steps.
Grass

Grass has nothing in its act method. It simply sits where it is and doesn’t move. Its sole purpose is to stay in its spot and make the simulation more aesthetically pleasing.
The Simulation Working
Solving the Problem

At all times, we kept the number of kids and cars at the real life ratio the same, and we also kept timing constant.

The part of the simulation that we changed between trials was the behavior of the CrossingGuard in turning the light red and green.

By analyzing the effects of the different methods, we determined which ones worked better than others, and which one worked the best.
Solution 1: Set Times

Our first Solution was to have a fixed time for the light to stay green between red lights.

This meant that the light change was completely independent of its surroundings.

Pros: This method provided regularity, and the timings could be set to the ratio of cars to children.

Cons: Since car and child creation was completely random, the light needed more ability to respond to the environment. With this method, large groups of children could bunch at the light, or traffic could be stopped when no children needed to cross.
Solution 2: One-Student Trigger

Our next solution was to turn the light red whenever there was any child waiting for the light.

We created the trigger by monitoring whether or not there were children in the CrossingGuard’s row.

Pros: This method ensured that children would always be able to cross in a timely manner.

Cons: This method resulted in a lot of traffic slowdown since the light would turn red any time there was a student waiting, which would happen frequently. A large number of cars would have to stop in order to ensure that just one student could cross.
Solution 3: Multi-Student Trigger

This method was very similar to the last in that it measured whether or not there were children in the CrossingGuard’s row.

This time, there needed to be more than one child to set off the trigger. We set the needed number to three children.

Pros: Traffic could stop less frequently because more students would cross during each red light.

Cons: Because children do not always come in groups, a single student could end up waiting at the light forever, or at least a very long time.
Solution 4: Probability Trigger

Our final solution used random probability to decide whether or not to turn the light red.

Whenever a child was waiting, there was a 1/6 chance that the light would change in the next movement.

Also, for each additional child another 1/6 was added to the probability.

Pros: This method combined the pros of solutions 2 and 3. The light usually waited until multiple children were waiting until it changed, but single children could also cross on their own eventually.

Cons: Since this method throws more random chance into the mix, there is always the possibility of rare abnormalities occurring. eg: a group as large as 5 in theory faces the chance of waiting forever.
Final Decision: Solution 4

The probability trigger created a system that was fair to both the cars and the children.

The possible abnormalities would be very very rare, and we never encountered them in our simulations, so we consider them to be a moot point.
Questions?