CS440: State and Action Spaces

Drew Kristensen

Action Space

In the SUMO environemnt, there are four possible light values for each direction of travel. Each light can take on a value from the set {GreenStraight, GreenLeft, Yellow, Red}. The default SUMO traffic lights run on a 90 second loop which cycles through these four values. Yellow, however, is an intermediate light which comes immediatley after a green phase such that the light progression goes from green to yellow to red. The maximum number of phases in a single intersection is 12, since I assume there to be no intersection with more than 6 incoming streets and there are four phases for each pair of streets opposite each other. However, these include the yellow phases, which aren't needed to specify in the network, since a yellow phase should follow a green phase and preceed the red phase. Including this in the network might improve the result, since there is no need to have a drawn out yellow if there is no car on the controlled lane while there are cars waiting elsewhere, but it will increase training time, since that is another variable we need to train towards. Initially, I plan to attempt to train the network with the yellow phases but this may prove too broad and may need to narrow down the output. From this, we will take the action space to be these 12 phases. In most cases, such as for any intersection with less than 6 incoming streets, there will be less than 12 actions to take. In these cases, we can still use the action space, but train it to never choose the actions it cannot take. For training, we can say that any output higher is clipped at whatever the maximum action value is. We can represent choosing an action by outputting an *argmax* over the vector given by each light.

State Space

As for the state space, Genders and Razavi developed their discrete traffic state encoding (DTSE), which holds three vectors which signal presensce of a vehicle in a cell, the speed of a vehicle in the cell, and the current traffic signal. While this is information rich and provides a lot of the information you could gather in an ideal world, collecting the first two values seem challenging enough that it would be incredibly difficult to implement (especially easily) in a real world environment. Instead, I propose using a different state space. This state space holds the number of cars curently at the light, the number of cars going under a certain speed threshold which signifies the number of cars waiting at the light, and the current traffic signal. By not requiring specific locations of vehicles, we lose spatial information in the problem but we gain the ability to more quickly implement the example in the real world. However, knowing the raw number at the light leaves too much information out. Therefore, using the number of cars in each lane (but not their exact locations) may be the perfect balance of opnipotence and realism. Furthermore, knowing the exact speed of vehicles helps with determining future spatial information but again, would be difficult without advanced equipment. Instead, I would like to use the average speed over three reference frames. While this sounds complicated, the thought behind it is this: for a camera mounted on a traffic light, being able to use the angle downwards which it is pointing, and the visual change in size of vehicles over a few fractions of second, it might be plausable to have a sufficeiently accurate estimate of the speed of the vehicles in the visual frame.

Conclusion

In conclusion, the action space is a vector of length 12 where each entry corresponds to a different traffic light signal. The tentative state space uses the number of vehicles in each lane, the average estimated speed for each vehicle, and the current traffic light signal.