EECS 126 Project – Fall 2019

For the project this semester, you will explore applications of EECS 126 toward either **Catan**, **Markov Chain Monte Carlo**, **Interacting Particle Systems**, or **Digital Communication**.

1 Logistics

This project is worth 5% of your course grade. You must work in groups of 3 or 4, and are encouraged to use Piazza's teammate finding feature if you are lacking group members. We strongly recommend forming teams as soon as possible. Only one student in each team is to submit the files to Gradescope. If your proposed idea isn't in line with what we (the staff) want, we will let you know shortly after the proposal due date, but this is rare. However, you are strongly encouraged to talk to the staff members handling your category of projects about your ideas before submitting.

Each team should only work on one project. You have a few days to form a team and decide on a topic, after which you have to submit a Google Form informing us about your team and the topic you will be working on (tentatively). Over the next week, you must formulate a concrete proposal in line with your topic, and this is due by November 15. Each topic has different requirements for the proposal, and these are listed under the topics in more detail. Your final submission will be due December 6.

1.1 Timeline

- November 1: Projects are released
- November 8: (11.59PM): Teammates and Topic Form Due (5%)
- November 15: (11:59 PM): Proposal due on Gradescope (15%)
- December 6 (11:59 PM): Entire project due on Gradescope (80%)

1.2 Extra Credit

Standout projects in each category will be awarded bonus points.

2 Catan

We found a way to simulate a (simplified) game of Catan[1]! Awesome! To ensure your domination on the island of Catan, you will need to design an action policy to win in the fewest number of turns possible, finding optimal settlement locations and determining the tradeoff between building additional settlements versus development cards. To start this project, download the Catan trailer lab (`Project_Catan.ipynb`) and our implementation of Catan (`catan.py`).

2.1 Task 1

Write up a proposal which describes your proposed strategy. Follow the instructions in the Catan lab to get acquainted with the platform. There is some starter code at the end that you might find useful.

2.2 Task 2

2.2.1 Optimal strategy

You should come up with a strategy to minimize the expected winning time by developing a better strategy for the **provided** (to be released November 8th or earlier) model of Catan. You are responsible for creating a new strategy (implemented as an action function, takes in 1 argument, the player), a planBoard function, that takes in the board and can be used to produce an overall plan for all trials run on that boards, and a dumpPolicy which determines what resources you will drop if you have too resources and a 7 is rolled. You are only allowed to modify these three methods, and your action / planBoard / dumpPolicy functions must operate within a time constraint (under 3 minutes for 100 trials on a fixed board). Make sure to follow the rules of Catan.

Your planBoard, action, and dumpPolicy methods should be submitted in a separate file `catanAction.py` (see sample), along with all relevant helper methods. This means that you should not put any of your work in `catan.py`, as we will be using a plain `catanAction.py` when we test your action method.

Important: To grade your code, we will be importing `planBoard`, `action`, and `dumpPolicy` using the following code:

```
catan_bot = __import__(filename)
action = catan_bot.action
dumpPolicy = catan_bot.dumpPolicy
planBoard = catan_bot.planBoard
```

Please make sure your code runs properly when imported.

2.2.2 Report

At the end, please write up a 3 to 5 page report that details your assumptions, attempted strategies, motivations behind these strategies, and how you ended up developing your final strategy. At this point, please include the mean time your algorithm takes to win, averaged over 10 randomly generated boards, running 100 trials per board.

[1]TA contact: Eric, Justin, Katie, Tae Hyung Kim

2.3 Grading

We will randomly generate boards and run your policy on each board multiple time. Your score will be weighted by the average time it took you to win across all boards. Code that does not run properly when imported or which errors out (such as divide by zero errors) will receive lower grades.

2.4 Submission

Please submit `catanAction.py` and your report to Gradescope.

3 MCMC Applications

Now that you understand the idea that led to the success of message decoding through the Metropolis-Hastings algorithm, seeing in lab how MCMC could be used to sample from distributions and solve ciphers, we want you to explore other applications of the powerful MCMC. [2].

3.1 Task 1: Proposal

We want to you decide another application of MCMC via a method not introduced in lecture or lab (i.e. NO message decoding); examples include Gibbs sampling, Slice sampling, Reversible-jump, Hamiltonian (or Hybrid) Monte Carlo, etc. Some examples in the past are:

- Model events using a Hierarchical Bayesian model and estimate the parameters using MCMC
- Perform special cases of MCMC (such as Gibbs sampling) for parameter estimation over a hierarchical model for a real world dataset
- Sample from the posterior distribution given the data of the weights of a neural network using MCMC instead of gradient descent to find the optimal solution

Please specify: What is the motivation behind your idea? What is the dataset you will be exploring/analyzing? What application and/or method of MCMC will you be using? What do you expect to see from the results?

The proposal should be about one page in length and include the names of all team members.

3.2 Task 2: Code and Report

Include all source code you wrote in a zip folder in your submission. Make sure to cite any code you may have copied from the web. You may use any programming language; Python is not necessary.

Summarize your results in a 3 to 5-page report (preferably typeset in LaTeX). We'd like you to include a working demo, preferably via a link to your recorded YouTube video. An example structure to the report includes the following sections:

- Introduction
- Methods (theory and code)

[2]TA contact: Sean, Michael, Aditya, Alan

- Experiments
- Results/analysis (with figures and a video demo)
- Discussion of limitations

3.3 Grading

The proposal will be graded on detail, clarity and innovation.

For the final check-off, a score of 7.5 out of 10 reflects a good understanding of the algorithm and implementation. A score higher than 7.5 out of 10 will require some creativity.

4 Interacting particle systems

A good model for understanding the behaviour of agents in a world is via simplified particle models. [3] You can essentially view these as ”advanced Markov chain projects”. Having seen a variety of Markov Chains so far, we expect that you will be able to understand and simulate various particle systems. By a particle system, we mean a system where each agent has a set of potential actions and chooses one uniformly at random. The following examples should give you a feel for what we mean.

4.1 Traffic Modeling: TASEP

Totally asymmetric exclusion processes are simple models that people use to model traffic. For the simple finite-length one-dimensional model, you have an N-cell road, and you have cars that are trying to get from point A to point B. Each car has an exponential clock and every time their clock goes off, they are trying to move forward with probability p. However, if there is a car in front of them, they don't move.

Figure 1: Taken from `https://arxiv.org/pdf/0803.2625.pdf`

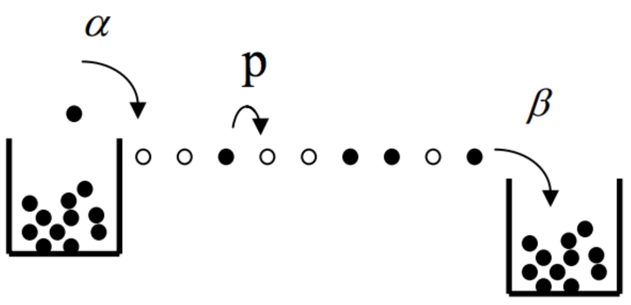


Figure 1. Schematic representation of the one-dimensional TASEP

Additionally, at the first and last cells, there are exponential clocks with rates α and β that introduce and remove cars from the system, respectively.
Things to look at are flux/current/how many cars move through a space at a given point, the number of contiguous cars (which indicate the blockages), number of cars in the system at a given time; stationary distributions of each cell (density profile) and so on.
If you choose to specialize on this model, some interesting experiments include:

- If you investigate the literature on one-dimensional TASEP, you will see that it includes these ”phase changes”. Run an experiment to show us what those phases are. (Bonus points for clearly explaining the theory, though this requires an understanding of basic partial differential equations).

- Increase the dimension. Have cars going go through the crossroads with rules that you may choose to define yourself. Have cars park and then leave after some time.

[3]TA contact: Alex, Kevin, Nikita

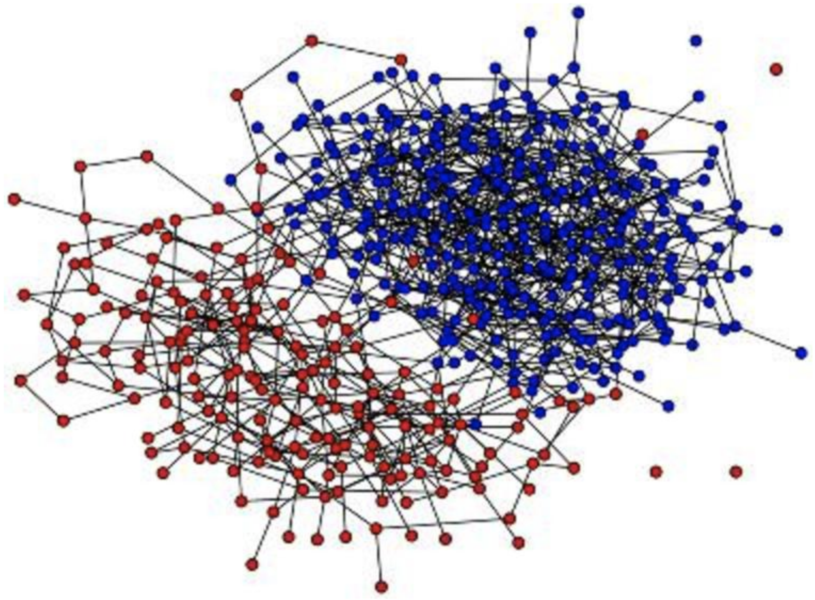
- Add cars that behave differently; for instance, add the notion of speed to cars. How does this change your density profiles and your current?

- Add accidents. With some probability, make cars break or make them hit other cars (and give them some time to clear off).

4.2 Spreading: Voter Models

Voter models are commonly used to model how contagions happen. This could be for instance a disease epidemic, a new emerging meme, or how incorrect news articles can travel far in the internet.

Figure 2: Taken from `https://en.wikipedia.org/wiki/Voter_model`



The model constitutes a graph where the nodes refer to people, and the directed edges are essentially influences. Now, each person has an exponential clock, and when their clocks go off, they pick another person to spread whatever they have to.
You can investigate how fast concepts spread, who are the major influencers, how long does it take for an opinion to dominate all others and other metrics that you think are interesting.
There are many, many forms of these models, so if you choose to specialize on this model, some interesting adaptations of this model include:

- You can add a notions of resistance to explain the willingness of people to share or to adapt.

- Add quality to different concepts; after all, some memes are better than others.

- Add the requirement that a certain number of people need to adapt a concept before it spreads (you only spread if half of your neighbors are part of a trend, for instance)

- Meme-maker model. You can designate some people to occasionally randomly start a new concept, and then you can analyze how that spreads in relation to the other things.

For this model, it is much more straightforward to try to acquire real world data:

- You can play a game like charades or hot-potato with your friends for like an hour and form some graph structure to pass around the phone/buzzer and see if you get similar results to your simulations.

- You can try to spread a meme on social media and predict its spread over time based on your information about the network. (Please do not spread viral incorrect information online; I would be rather sad if we contribute to bad information pollution.)

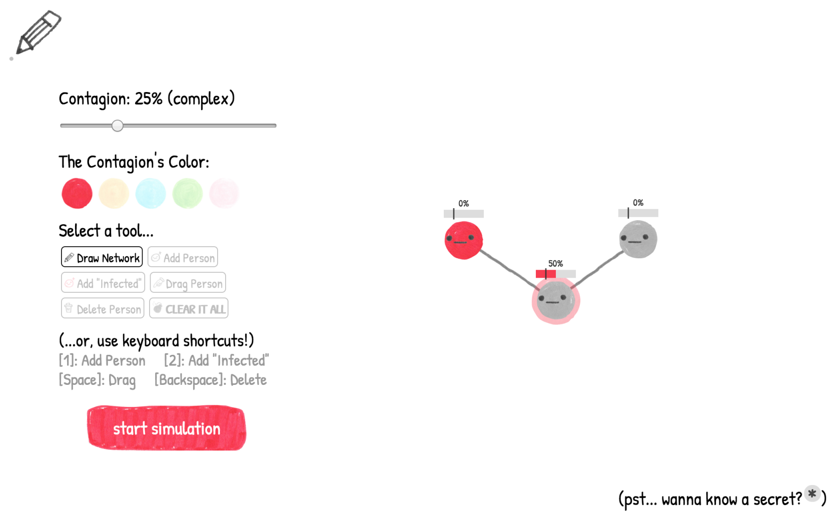
- You can look up data on how diseases spread, and use that data set to understand it.

4.3 Task 1: Proposal

We want you to fix a model you will work on, a set of interesting question(s) that your model will help you answer, and what experiments you will run. In particular, we would like to see a simple simulation up and running by this point.
The proposal should be less than two pages in length and include the names of all team members.

4.4 Task 2: Code and Final Report

Figure 3: Taken from `https://ncase.me/crowds/`



You will be building this system from scratch, so feel free to use any programming language of your choice. Please be sure to cite code snippets that you copied from other resources, and cite any academic work you build off of. Include all of your source code in a zip folder in your submission to Gradescope.

Ideally, we want an applet that one can tweak the parameters of and then see the simulation run in front of our eyes. The screenshot from ncase.me is a rather impressive applet that we have seen online and obviously a project of that high caliber is not what we expect. However, it should be giving you an idea on how to design such a mini applet. Some students have previously added sliders to a Jupyter notebook, which they can use to adjust the simulation.

In addition, write a three to five page report on the project. In the report, you should include questions you formed at the beginning, careful explanation of the experiments you designed to test your questions, some challenges you faced during the project and how you resolved them, and conclusions you drew. Also explicitly list out the technical outline of the project, and the specific results observed from various modeling strategies.

4.5 Grading

The written aspects of the project – the proposal and the final report – will be graded on detail and clarity, as well as technical prowess and accuracy.

5 Digital Communication

Our last project theme is digital communication[4], and for this project, you will build a system from scratch to transmit a text file between two laptops, using only the speakers and microphones of the computers. Your system should be able to transfer the file at a reasonable speed and it should also be robust to noise, both ambient noise and burst noise (like clapping). For full credit on the transfer speed aspect of the project, you will need to achieve a bitrate of at least 100 bits per second.

5.1 Task 1: Proposal

Your proposal should describe how you plan on implementing your digital communication system. You should also describe how your system will achieve the goals listed above (transfer speed and robustness to noise). The proposal should be about one to two pages in length, and remember to include the names of all team members.

5.2 Task 2: Code and Final Report

You will be building this system from scratch, so feel free to use any programming language of your choice. Please be sure to cite code snippets that you copied from other resources. Include all of your source code in a zip folder in your submission to Gradescope.

In addition, write a two to three page report on the project. In the report, you should include a description of your final implementation, some challenges you faced during the project and how you resolved them, and what you learned from the project.

5.3 Grading

The written aspects of the project – the proposal and the final report – will be graded on detail and clarity.

To grade your implementation, you will schedule a time to demonstrate your system to one or two members of the course staff. We will give you a text file (.txt format), and you will show us how your system transmits the file from one laptop to another. For the purposes of the competition, we will consider things like how fast of a bitrate you can achieve and how robust the system is to noise.

[4]TA contact: Ray, William, Raghav