CMPSC 310
Artificial Intelligence
Fall 2018
Janyl Jumadinova

Lab 2
10 September, 2017
Due: 17 September by 2:30 pm
This is an individual lab.

Objectives

To learn how to use GitHub to access the files for a laboratory assignment. To continue experimenting with an agent-based modeling tool called, NetLogo, to design agent-based models. To learn how to an A* search can be implemented in NetLogo. To reflect on the Netlogo A* search model and the implementation, its potential uses for the AI problems and to explore making small modifications to the model in order to gain more practice programming in NetLogo.

Reading Assignment

If you have not done so already, please read all of the relevant "GitHub Guides", available at https://guides.github.com/, that explain how to use many of the features that GitHub provides. In particular, please make sure that you have read guides such as "Mastering Markdown" and "Documenting Your Projects on GitHub"; each of them will help you to understand how to use both GitHub and GitHub Classroom. To help you understand the search problems and the A* solution, you should also read Chapter 3 in the FCA book, concentrating particularly on Section 3.6.

Configuring Git and GitHub

During this and the subsequent laboratory assignments, we will securely communicate with the GitHub servers that will host all of the project templates and your submitted deliverables. In this assignment, you will perform all of the steps to configure your account on GitHub, so that you can complete lab assignments using GitHub Classroom. You can also learn more about GitHub Classroom by visiting https://classroom.github.com/. As you will be required to use Git, an industry standard tool, in all of the laboratory and remaining practical assignments and during the class sessions, you should keep a record of all of the steps that you complete and the challenges that you face. You may see the course instructor or one of the teaching assistants if you are not able to complete a certain step or if you are not sure how to proceed.

1. If you do not already have a GitHub account, then please go to the GitHub website (https://github.com/) and create one, making sure that you use your "allegheny.edu" email address so that you can join GitHub as a student at an accredited educational institution. You are also encouraged to sign up for GitHub's "Student Developer Pack" at https://education.github.com/pack, qualifying you to receive free software development tools. Additionally, please add a description of yourself and an appropriate professional photograph

to your GitHub profile. Unless your username is taken, you should also pick your GitHub username to be the same as Allegheny's Google-based email account.

2. If you have never done so before, you must use the "ssh−keygen" program to create secure-shell keys that you can use to support your communication with GitHub. Open the terminal and type the "ssh−keygen" command in it. Follow the prompts to create your keys and save them in the default directory. That is, you should press "Enter" after you are prompted to "Enter file in which to save the key ... :" and then type your selected passphrase whenever you are prompted to do so or press "Enter" for no passphrase to be used. Please note that a "passphrase" is like a password that you will type when you need to prove your identify to GitHub.

3. Now, in the GitHub's website and look in the right corner for an account avatar with a down arrow. Click on this link and then select the "Settings" option. Now, scroll down until you find the "SSH and GPG keys" label on the left, click to create a "New SSH key", and then upload your ssh key to GitHub. You can copy your SSH key to the clipboard by going to the terminal and typing "`cat ~/.ssh/id_rsa.pub`" command and then highlighting this output. When you are completing this step in your terminal window, please make sure that you only highlight the letters and numbers in your key—if you highlight any extra symbols or spaces then this step may not work correctly. Then, paste this into the GitHub text field in your web browser.

To access the laboratory assignment, you should go into the #labs channel in our Slack team and find the announcement that provides a link for it. Copy this link and paste it into your web browser. Now, you should accept the laboratory assignment and see that GitHub Classroom created a new GitHub repository for you to access the assignment's starting materials and to store the completed version of your assignment. Specifically, to access your new GitHub repository for this assignment, please click the green "Accept" button and then click the link that is prefaced with the label "Your assignment has been created here". If you accepted the assignment and correctly followed these steps, you should have created a GitHub repository with a name like 'lab-2-cmpsc-310-fall-2018-allegheny-college-jjumadinova". Unless you provide the instructor with documentation of the extenuating circumstances that you are facing, not accepting the assignment means that you automatically receive a failing grade for it.

Now you are ready to download the starting materials to your laboratory computer. Click the "Clone or download" button and, after ensuring that you have selected "Clone with SSH", please copy this command to your clipboard. By typing "git clone" in your terminal and then pasting in the string that you copied from the GitHub site you will download all of the code for this assignment.

After this command finishes, use "cd" to change into the new directory. Now you are ready to investigate the starter files for this lab.

Shortest Path Finding Problem

During our last class we have explored the idea of solving problems by projecting them on the state spaces and then using a search algorithm to find the solution. A simple goal-oriented agent may use search algorithms to find a solution to a particular problem. For example, a path finding agent may use a search strategy to find the optimal path from the starting location to the end location.

In graphical terms, the shortest path problem is the problem of finding a path between two nodes in a graph so that the sum of the weights of its edges is minimized. In AI, the shortest path problem has many applications. Most obvious application is in robotics, where a robot has to navigate the environment and get from point A to point B using the most optimal path, while avoiding obstacles. Path finding is also widely utilized in the modern video games, where the concern is getting the player from one point on the map to another while considering the terrain, obstacles, and possibly collisions (with other players). Path finding is also utilized in other less obvious applications, such as algorithmic trading and social network analysis.

A* Implementation in NetLogo

If any global information about the structure of the space is available *informed search* algorithms can be utilized for a faster solution. In most cases we can get an approximation of how much it will cost to reach the goal, that is known as a *heuristic*. The idea behind a heuristic search is that we explore the node that is most likely to be nearest to a goal state. Among the different search algorithms that make use of partial information by using heuristics, the most famous is the A* algorithm. A* search uses both the cost of the path and the heuristic information. That is, in an A* algorithm for every state s we compute the estimate of the total path cost:
$f(s) = g(s) + h(s)$,
where $g(s)$ provides the cost of the path from the start state to current state, s, and $h(s)$ estimates the cost of the cheapest path from the current node to the goal. A* is the most popular choice for path finding because it is pretty flexible and can be used in a wide range of contexts. In this lab you will explore one implementation of the A* algorithm in Netlogo and make modifications to it.

First of all, study the A* implementation for the shortest path finding problem found in your cloned lab 2 repository inside `src` directory. This implementation is a slight modification of the model from the Netlogo community page:
http://ccl.northwestern.edu/netlogo/models/community/Astardemo1
Complete the next sequence of steps using this model:

1. Read the documentation in the Info tab. Then run the model as is. Next, build at least three different mazes (from simple to more complex), save them and run the model using those mazes.

2. Now, turn to the source code to see how model was set up and how path finding was implemented. Since there are varients of A* implementations, study at the `find-a-path` procedure and make a note of the steps and data structures used in this implementation. Use NetLogo's documentation https://ccl.northwestern.edu/netlogo/docs/ to understand various functions. For example, note that `distance` gives the distance from this agent to the

specified turtle/patch. How could you describe this implementation in the pseudocode form or in plain English? In a `writing/reflection.md` file add your pseudocode (preferably) or an explanation in English of the A* implementation of this model.

3. Now make at least three changes to the model you have just ran. This portion of the lab is completely open-ended, you are free to make visual changes to the model and/or make changes to the implementation of the A* algorithm, depending on your comfort-level with NetLogo and your previous programming experience. For example, is there a better visual representation of the generated model? Is there a better layout to use for the model?Can you generate a more informative output, for example, by creating graph(s)? Can you improve the implementation of the A*? The goal of this part of the lab is to allow you to improve the given model.

4. Modify documentation in the applicable sections under the Info tab to describe all of the changes you have made.

Reflection on the A* Implementation for Path Finding

Write a reflection report in the document `writing/reflection.md` found in the lab 2 repository that contains the following:

- A pseudocode or a description (in English) of the original A* implementation.
- Detailed description of your modifications.
- From your observations, comment on the general performance of the A* algorithm. Is A* algorithm an appropriate algorithm for path finding problems? Are there any cases when it may fail?
- Describe how A* search may be used in one or two realistic applications, other than the ones mentioned in this assignment sheet.

Please remember that your reflection is a Markdown file that must adhere to the standards described in the [Markdown Syntax Guide](https://guides.github.com/features/mastering-markdown/). Remember, you can preview the contents of a committed Markdown file by clicking on the name of the file in your GitHub repository.

Required Deliverables

This assignment invites you to submit electronic versions of the following deliverables through your GitHub repository.

1. Modified NetLogo model with the A* implementation, including your well-documented and commented modifications. Add comments to the program and your documentation to the "Info" tab appropriately.

2. Reflection document including details on the A* algorithm and the performance of the A* implementation, description of the model changes and an application scenario for potential usage of such a solution.