MASSACHVSETTS INSTITVTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.01—Introduction to EECS I
Fall Semester, 2007

Assignment 11, Issued: Tuesday, Nov. 13

To do this week

...in Tuesday software lab

1. Start writing code and test cases for the numbered questions (1–6) in the software lab. Paste all your code, including your test cases, into the box provided in the "Software Lab" problem on the on-line Tutor. What you submit to the tutor will not be graded; what you hand in next Tuesday will.

...before the start of lab on Thursday

1. Read the lecture notes.
2. Do the Pre-Lab homework problems for week 11 that are due on Thursday (Questions 7-13).
3. Read through the entire description of Thursday's lab.

...in Thursday robot lab

1. Do the nanoquiz at the start of the lab. It will be based on the material in the lecture notes and the homework due on Thursday.
2. Answer the numbered questions (Questions 14–25) in the robot lab and demonstrate the appropriate ones to your LA.

...before the start of lecture next Tuesday

1. Do the lab writeup, providing written answers (including code and test cases) for **every** numbered question in this handout.

On Athena machines make sure you do:
`athrun 6.01 update`
so that you can get the `Desktop/6.01/lab11` directory which has the files mentioned in this handout.

- You need the file `search.py` for the software lab.

During software lab, if you are using your own laptop, download the file(s) from the course Web site (on the Calendar page).

Planning

So far, our robots have always chosen actions based on a relatively short-term or "myopic" view of what was happening. They haven't explicitly considered the long-term effects of their actions. One way to select actions is to mentally simulate their consequences: you might plan your errands for the day by thinking through what would happen if you went to the bank after the store rather than before, for example. Rather than trying out a complex course of action in the real world, we can think about it and, as Karl Popper said, "let our hypotheses die in our stead." We can use state-space search as a formal model for planning courses of action, by considering different paths through state space until we find one that's satisfactory.

This week, we'll assume that the robot can know exactly where it is in the world, and plan how to get from there to a goal. Generally speaking, this is not a very good assumption, and we'll spend the next two weeks trying to see how to get the robot to estimate its position using a map. But this is a fine place to start our study of robot planning.

We will do one thing this week that (at first) doesn't seem strictly necessary, but will be an important part of our software structure as we move forward: we are going to design our software so that the robot might, in fact, change its idea of where it is in the world as it is executing its plan to get to the goal. This is very likely to happen if it is using a map to localize itself (you've probably all had the experience of deciding you weren't where you thought you were as you were navigating through a strange city). This week, the only way it can happen is if, in the simulator, a malicious person drags the robot to another part of the world as it is driving around. However, we will see later in this lab that the robot can in fact be in a similar situation if it fails to predict the effect of its actions accurately.

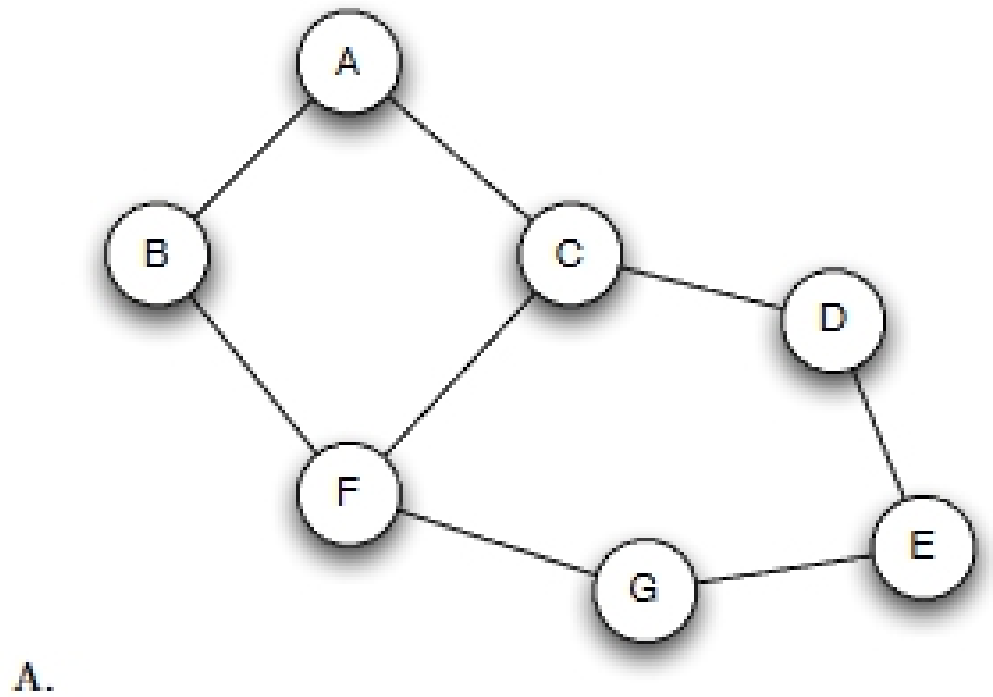
There are still a lot of details to be ironed out before we get this all to work, which we'll talk about later.

Tuesday Software Lab

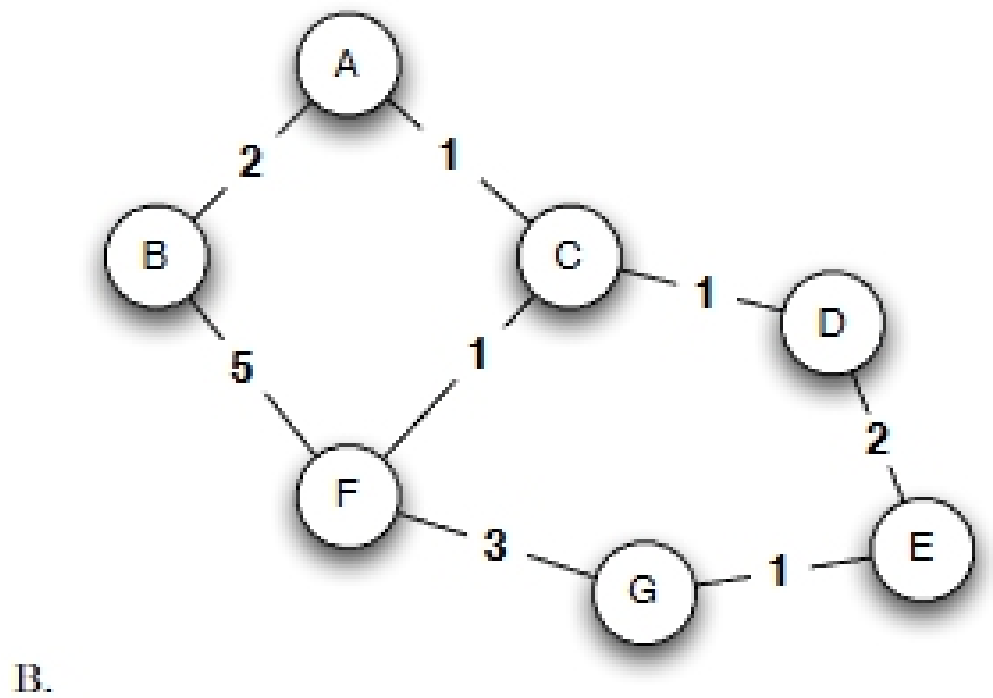
Please do the following programming problems.

Experimenting with search

The code file `search.py` contains the code for the search algorithms and the `numberTest` domain discussed in lecture. Load this into Python (not Soar, just ordinary Python, in Idle) so you can experiment with it.



A.



B.

Figure 1: A. A simple map. B. A map with each road labeled by the time it takes to traverse it.