Semester 2 2018
COMP3702/7702 ARTIFICIAL INTELLIGENCE
ASSIGNMENT 2: Game Changing Technology

Note:

- This assignment consists of two parts: Programming and report.
- You can do this assignment in a group of **at most** 3 students. This means you can also do the assignment individually.
- For those who choose to work in a group:
  - All students in the group must be enrolled in the same course code, i.e., all COMP3702 students or all COMP7702 students.
  - Please register your group name in before **5pm on Tuesday, 23 Oct 2018 in** http://robotics.itee.uq.edu.au/~ai/a2Group.html. If you have not registered your group by the said time, you will need to work on the assignment individually.
  - All group members are expected to work in both programming and report. The demo will involve Q&A to each group member, individually.
- Submission Instruction:
  - Your program should compile from command prompt and generate an executable that can be run from command prompt as:

    ```
    > java ProgramName inputFileName outputFileName
    ```

  - You should submit **only the source code** required to compile the program, i.e., remove all object files and executable before submission.
  - The report should be in .pdf format and named a2-[courseCode]-[ID].pdf.
    If you work individually, ID is your student number.
    If you work in a group, ID is the student number of all group members separated by a dash. For instance, if you work in a group of two, and the student number is 12345 and 45678, then
    [ID] should be replaced with `12345-45678`
  - The report and all the source codes necessary to compile your program should be placed inside a folder named a2-[courseCode]-[ID].
  - The folder should be zipped under the same name, i.e., a2-[courseCode]-[ID].zip, and the zip file should be submitted via turnitin before **5pm on Tuesday, 13 November 2018**. Grace period until Wednesday 8am, Wednesday 14 November 2018.
- Demo Instruction:
  - Demo will be on Wednesday, 14 Nov 2018 to Friday, 16 Nov 2018.
  - Please register for a demo slot before **5pm on Tuesday, 13 Nov 2018** in http://robotics.itee.uq.edu.au/~ai/a2Demo.html
  - If you work in a group, all group members must be present for the demo.
  - You must use the lab PC for the demo, and therefore you must make sure that your code does compile and run well in the lab PC.

In this assignment, you will develop an MDP agent to play a game, named *Changing Technology*. In a nutshell, game Changing Technology is a suburban and off-road touring car racing game, where the emphasize is on the strategy that a team should take —that is, deciding which driver, car models, and car components should be used when and where—, so that the team can reach the goal within a given number of steps.

Photo taken from http://4wdguys.com.au/opposite-lock-challenge-mudgee-2012/

In game Changing Technology, a car operates in an environment with varying terrain types. The possible terrain types are: 1. dirt-straight-hilly, 2. dirt-straight-flat, 3. dirt-slalom-hilly, 4. dirt-slalom-flat, 5. asphalt-straight-hilly, 6. asphalt-straight-flat, 7. asphalt-slalom-hilly, and 8. asphalt-slalom-flat.

The environment map, including the terrain types, is known perfectly and is represented as 1-dimensional grid cell of size N, where column-1 is the starting point and column-N is the goal region.

At each step, the team might be able to do one of the following:

A1. Continue moving. Depending on the terrain, driver, car models, and car components, each movement might move the car k cells, where $k \in [-4, 5] \cup \{slip, breakdown\}$. The result of the exact movement is uncertain and represented as a probability distribution. This distribution is conditioned on the components discussed in action #2—#7 below. To simplify, ***this game assumes conditional independence*** and prior distribution is assumed to be uniform over the set $[-4, 5] \cup \{slip, breakdown\}$.
If the movement causes the car to have non-zero probability mass of being in cell > N, then all probability mass for cells beyond N will be added to the probability mass of being at cell-N. The same is applied to cell-1.
The car might also slip, which will incur a substantial time before it can start moving again. The car might also break down, which will incur a repair time.

A2. Change the car type. Different type will have different speed in different terrains and different fuel efficiency. After the change, the car fuel is at full tank (50 liters) and the pressure of all of its tires are at 100% capacity.

A3. Change the driver. Different drivers will have different speed in different terrains.

A4. Change the tire(s) of existing car. The team has an option to change its entire tires with another model. The available tire types are: all-terrain, mud, low-profile, performance. The different types of tires and terrains will influence how far the car will move in a single step. After the change, the pressure of all its tires is at 100% capacity.

A5. Add fuel to existing car. The car fuel level is an integer, with range 0 (empty tank) to 50 liters (full tank). The time it takes to fill up the fuel by x amount is ceil(x/10) steps. For example: If 8 liters is added, at the next step the agent can do something else, but if 19 liters is added, only at the next next step, the agent can choose another different action.
A6. Change pressure to the tires. The team can add/reduce the pressure to 50% capacity, 75% capacity, or 100% capacity. The fuel consumption when the pressure is at 75% capacity is double that of 100% capacity. The fuel consumption when pressure is at 50% capacity is triple that of 100% capacity. However, the chances of slip at 75% capacity is double that of 50% capacity, and the chances of slip at 100% capacity is triple that of 50% capacity.
A7. Combination of options A2 and A3.
A8. Combination of options A4 —A6. The time to do the change follows the slowest step (i.e., if fuel is added, this will be fuel addition).

The game has 5 levels, which differ in the possible terrain types and available actions, as follows:

Level-1: Only two types of terrain (i.e., dirt and asphalt) with $N \leq 10$, and only actions A1—A4 are possible, with two different car models and two different drivers to choose from. Fuel, pressure, and oil quality are assumed to remain the same throughout the race.

Level-2: Only four types of terrain (i.e., dirt-straight, dirt-slalom, asphalt-straight, and asphalt-slalom) with $N \leq 10$, and only actions A1—A6 are possible, with three different car models and two different drivers to choose from.

Level-3: All eight terrain types with $N \leq 30$, and only actions A1—A6 are possible, with five different car models and five different drivers to choose from.

Level-4: All eight terrain types with $N \leq 30$, and actions A1—A7 are possible, with five different car models and five different drivers to choose from.

Level-5: All eight terrain types with $N \leq 30$, and all eight classes of actions are possible, with five different car models and five different drivers to choose from.

Probability Hint:

Suppose D, B, and C are random variables. If we know P(D=d | B = b), P(D=d | C = c), and P(D = d). Then, we can compute P(D = d | B = b, C = c) as follows:

$$P(D = d|B = b, C = c) = \frac{P(D = d|B = b)P(D = d|C = c)P(D = d)}{\sum_{all\ possible\ d} P(D = d|B = b)P(D = d|C = c)P(D = d)}$$

What you need to do

Your tasks in this assignment can be classified into 3 parts:

1. Design the solution. This task contains two main components, i.e., framing the problem as an MDP problem and deciding how to solve the problem. Note that you should design the MDP components, i.e., what are the states, actions, transition, and reward functions, manually. However, the exact parameters of your MDP problem will depend on the given input file.
2. Implement your design. Your program is allowed a maximum of 2 minutes computation prior to each simulation run. If you use an online method for solving MDP, at each step during run-time, your program is allowed a maximum of 15 seconds computation time. Your program will be run on a PC in the tutorial room. The time requirement is for a program that runs as a single-threaded process. If you use multi-threading, then we will divide the aforementioned time limit with the number of threads you use. Note: ***You are not allowed to use any library for linear algebra, optimization, and MDP solver***.
3. Write a report, which answers the questions we post for the Report section (see the last part of this document). Note that to answer the questions well, you need experimental results, which means you do need to do #2.

Input and Output format

Input format. The input is a single .txt file, containing information about the race and how the terrain types affect performance of the car, driver, and car components. The format is as follows.

1. The first line is the level of the game, i.e., 1, 2, 3, 4, or 5.
2. The second line consists of three numbers separated by a white space. The first number is the discount factor, the second number is the time to recover from a slip, the last number is repair time to recoer from a breakdown.
3. The third line consists of two numbers. The first number is the umber of cells in the map, i.e., *N*. The second number is the maximum number of time-steps the agent is allowed to take for reaching the goal region. Let's denote this number as *MaxT*.
4. Line 4 to (3+NT), where NT is the number of terrain for this level, has two components separated by a colon (':'). To the left of the colon is the type of terrain, written as all small letters as per description in p. 2-3. To the right of the colon is the index of the cells with such a terrain type, separated by a comma. A continuous sequence of cells with the same terrain type can be written as beginningCellIdx-endCellIdx. For example:

   ```
   dirt-straight-hilly:1-3,5,7
   ```

   means cell index 1, 2, 3, 5, and 7 has dirt-straight-hilly as its terrain type.
   Note: If there's no cell with such terrain type, then the right side of the colon is empty. The cell index starts from 1.
5. Line 3+NT+1 is the number of car types. Let's denote this as CT.
6. Each line in the subsequent CT lines has two components separated by a colon (':'). To the left of the colon is the type of car. The type of the car is always a

single word. To the right of the colon is 12 numbers separated by a white space. The numbers represent conditional probability mass that given this particular car is used, the results of moving (i.e., *k*) is -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, slip, and breakdown, respectively.

7. Subsequent line is the number of drivers. Let's denote this as DT.
8. Each line in the subsequent DT lines has two components separated by a colon (':'). To the left of the colon is the driver's name. The driver's name is always a single word. To the right of the colon is 12 numbers separated by a white space. The numbers represent conditional probability mass that given this particular driver, the results of moving (i.e., *k*) is -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, slip, and breakdown, respectively.
9. Each line in the subsequent 4 lines has two components separated by a colon (':'). To the left of the colon is the tires model, written as per A4. To the right is 12 numbers separated by a white space. The numbers represent conditional probability mass that given this particular model of tires are used, the results of moving (i.e., *k*) is -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, slip, and breakdown, respectively.
10. The next line consists of NT*CT (NT: #types of terrain, CT: #types of car) numbers. Each number represents the amount of fuel used by a single moving action (i.e., A1) when pressure of the tires is at 100% capacity. The first is the number for the first terrain type (as per Input #4) and the first car type (as per Input #6), the second is the number for the first terrain type and the second car type, etc..
11. The next line consists of NT*CT (NT: #types of terrain, CT: #types of car) numbers. Each number represents the slip probability of a single moving action (i.e., A1) when pressure of the tires is at 50% capacity. The first is the number for the first terrain type (as per Input #4) and the first car type (as per Input #6), the second is the number for the first terrain type and the second car type, etc..

Output format. The output is a single .txt file. The number of lines in the output file is the lower among the *number of steps to reach the goal* +1 or *MaxT+1* (*maxT* refers to #3 on Input format) lines.

Each line-*i* consists of 2 tuples separated by a semi-colon (';'). Each tuple is written inside a bracket, start with '(' and ends with ') '.
The first tuple consists of 8 components separated by a comma. In sequential order, the components are: the cell index where the car is currently located, whether the car is in slip condition or not (1 slip, 0 not slip), whether the car is in breakdown condition or not (1 breaks down, 0 good condition), the current car type, the current driver's name, the current tires type, the current fuel level, the current tire pressure.
The second tuple consists of the index (e.g., A1) of the action(s) performed at time-*i* (in ascending order) followed by the value (if values are required). The action index and its value are separated by a colon (':'). If more than one actions are performed, the actions are separated by a comma. If there's no action performed, the second tuple is written as '(n.a.)'.

Grading for the Programming Part (total points: 60/100)

The details of the grading scheme is as follows. If your situation satisfies more than one marking band, we will use the higher band.

COMP3702:

- >= 1 & < 10: The program does not compile nor run (staff discretion).
- >= 10 & < 20: The program runs but fails to solve any query within the given time limit (staff discretion).
- 30: The program solves a query involving up to 2 level-1 cases.
- 40: The program solves a query involving up to 2 level-2 cases.
- 50: The program solves a query involving up to 2 level-3 cases.
- 60: The program solves a query involving up to 2 level-4 cases.

COMP7702:

- >= 1 & < 5: The program does not compile nor run (staff discretion).
- >= 5 & < 10: The program runs but fails to solve any query within the given time limit (staff discretion).
- 20: The program solves a query involving up to 2 level-1 cases.
- 30: The program solves a query involving up to 2 level-2 cases.
- 40: The program solves a query involving up to 2 level-3 cases.
- 50: The program solves a query involving up to 2 level-4 cases.
- 60: The program solves a query involving up to 2 level-5 cases.

Report (total points: 40/100)

Your report must contain answers to the following questions:

1. [10 points] Please define your MDP problem.
2. [10 points] Please explain the method you use to solve the problem at the conceptual level (i.e., pseudo code and what abstract data structure is used for the container).
3. [20 points] Please analyse your algorithm's time and memory complexity, via comparison study. For this purpose, at the very least, you need to analyse the performance as the size of the problem (i.e., size of state space and action space) increases. Please note that in this question, you will get higher mark for stronger arguments. A strong argument should at least include a logical explanation of why and include experimental results to back your explanation. Also, please also note that good explanation is NOT equal to long explanation!!!

oOo That's All, Folks oOo

☺ Once you do this assignment, you can write in your CV that you have worked on a GAME CHANGING TECHNOLOGY ☺