CS 533 Intelligent Agents and Decision Making, Spring 2018

Homework #3

Yathartha Tuladhar

Instructions to run the code:

The file to build policy is “build_parking_mdp.py”
You can open it in a python IDE, and set the parameters in load_args()
Or, run it from the terminal.

To view the arguments, run
Run: python build_parking_mdp.py -h

To run the value iteration, run the “test_mdp.py”
Run “python test_mdp.py -h” for options

Part 1: Build a Planner

The planner was similar to the one in previous homework, but the algorithm was slightly modified from finite value iteration, to infinite value iteration. The program takes in a text file, and parses it to extract the MDP and all the necessary information. Then it calculates the value function and the policy for this MDP.

Here it uses epsilon as the stopping criteria. Whenever the norm of the difference of previous time-step values and the new ones are smaller than epsilon (which was set to 0.000001), the value iteration will stop.

Part 2: Run the Planner

The planner was run on the two sets of MDPs that were provided.

MDP1.txt
The results with **beta = 0.9** is shown below.
(Sutton and barto uses gamma instead of beta)

```
num_states = 10, num_actions = 4
Infinite horizon case since time-step provided was 0
Gamma = 0.9
epsilon = 1e-06
**************************************
Value : ['3.3210', '2.9233', '2.8914', '2.9233', '3.6900', '2.8407', '3.1564', '2.9071', '2.9889', '3.2482']
Policy : ['3.0000', '3.0000', '2.0000', '0.0000', '0.0000', '0.0000', '1.0000', '2.0000', '1.0000', '3.0000']
```

The bound on how suboptimal the resulting policy is can be calculated as:

bound = epsilon*beta / (1 – beta)
bound = 0.000001*0.9 / (1 – 0.9)
bound = 0.000009
Thus the value function is 0.000009 factor of the optimal value function.

The results with **beta = 0.1** is shown below.

```
num_states = 10, num_actions = 4
Infinite horizon case since time-step provided was 0
Gamma = 0.1
epsilon = 1e-06
**************************************
Value : ['0.1001', '0.0090', '0.0088', '0.0090', '1.0010', '0.0068', '0.0683', '0.0086', '0.0100', '0.0896']
Policy : ['3.0000', '3.0000', '2.0000', '0.0000', '0.0000', '0.0000', '1.0000', '2.0000', '1.0000', '3.0000']
```

The bound on how suboptimal the resulting policy is can be calculated as:

bound = epsilon*beta / (1 – beta)
bound = 0.000001*0.1 / (1 – 0.1)
bound = 1.1111e-7
Thus the value function is 1.1111e-7 factor of the optimal value function.

MDP2.txt

The results with **beta = 0.9** is shown below.

```
num_states = 10, num_actions = 4
Infinite horizon case since time-step provided was 0
Gamma = 0.9
epsilon = 1e-06
*************************************
Value : ['4.2632', '4.2576', '5.1885', '3.8440', '4.4169', '4.7368', '5.2632', '3.8351', '3.9752', '4.7368']
Policy : ['0.0000', '0.0000', '0.0000', '1.0000', '2.0000', '2.0000', '2.0000', '1.0000', '1.0000', '2.0000']
```

The bound on how suboptimal the resulting policy is can be calculated as:

bound = epsilon*beta / (1 – beta)
bound = 0.000001*0.9 / (1 – 0.9)
bound = 0.000009
Thus the value function is 0.000009 factor of the optimal value function.

The results with **beta = 0.1** is shown below.

```
num_states = 10, num_actions = 4
Infinite horizon case since time-step provided was 0
Gamma = 0.1
epsilon = 1e-06
*************************************
Value : ['0.0114', '0.0100', '0.5733', '0.0015', '0.0604', '0.1010', '1.0101', '0.0080', '0.0060', '0.1010']
Policy : ['3.0000', '0.0000', '0.0000', '1.0000', '2.0000', '2.0000', '2.0000', '3.0000', '1.0000', '2.0000']
```

The bound on how suboptimal the resulting policy is can be calculated as:

bound = epsilon*beta / (1 – beta)
bound = 0.000001*0.1 / (1 – 0.1)
bound = 1.1111e-7
Thus the value function is 1.1111e-7 factor of the optimal value function.

Part 3: Parking Domain

The parking lot is designed as follows:

```
----------------------------
        STORE
----------------------------
    A[0]        B[0]       : handicapped row
    A[1]        B[1]
    A[2]        B[2]
    A[3]        B[3]
    .           .
    .           .
    .           .
    A[end]      B[end]
```

The MDP was designed as suggested in the assignment. However, one modification was made to separate the two aisles (to setup the problem easier). Thus now the MDP is a four tuple {Aisle, Row, Occupied, Parked}.

Probabilities for a row in an aisle being occupied is set up as:

- handicap spot = 0.001
- For other spots, the gradient is as:
  (total_rows_in_aisle – current_row) / total_rows_in_aisle

  Thus, nearer the current row (to the store), lesser the probability of being occupied.

These probabilities are kept constant. Thus for the two parameter scenarios we will be looking at, the probability of a parking spot being occupied is set the same way.

In the first parameter (safe driver) scenario, the rewards are as follow:

- Driving (not being parked) = -1
- Parking in handicapped = -100
- Crashing = -1000
- In other states there is a gradient of reward:
  (total_rows_in_aisle – current_row) * 10

  Thus smaller the current_row, higher the reward.
- The end state has a reward of 1

The results for the first parameter (safe driver) are shown in the next page.

Parameters 1: Safe driver Value function

(Aisle, Row, O, P)	Value
['A', 0, 'Unoccupied', 'NotParked']	63.1607075197
['A', 0, 'Unoccupied', 'Parked ']	-99.1
['A', 0, 'Occupied ', 'NotParked']	63.1607075197
['A', 0, 'Occupied ', 'Parked ']	-999.1
['A', 1, 'Unoccupied', 'NotParked']	80.81
['A', 1, 'Unoccupied', 'Parked ']	90.9
['A', 1, 'Occupied ', 'NotParked']	55.8446367672
['A', 1, 'Occupied ', 'Parked ']	-999.1
['A', 2, 'Unoccupied', 'NotParked']	71.81
['A', 2, 'Unoccupied', 'Parked ']	80.9
['A', 2, 'Occupied ', 'NotParked']	53.753938472
['A', 2, 'Occupied ', 'Parked ']	-999.1
['A', 3, 'Unoccupied', 'NotParked']	62.81
['A', 3, 'Unoccupied', 'Parked ']	70.9
['A', 3, 'Occupied ', 'NotParked']	52.2536812368
['A', 3, 'Occupied ', 'Parked ']	-999.1
['A', 4, 'Unoccupied', 'NotParked']	53.81
['A', 4, 'Unoccupied', 'Parked ']	60.9
['A', 4, 'Occupied ', 'NotParked']	49.8285878676
['A', 4, 'Occupied ', 'Parked ']	-999.1
['A', 5, 'Unoccupied', 'NotParked']	45.6373645402
['A', 5, 'Unoccupied', 'Parked ']	50.9
['A', 5, 'Occupied ', 'NotParked']	45.6373645402
['A', 5, 'Occupied ', 'Parked ']	-999.1
['A', 6, 'Unoccupied', 'NotParked']	40.073628086
['A', 6, 'Unoccupied', 'Parked ']	40.9
['A', 6, 'Occupied ', 'NotParked']	40.073628086
['A', 6, 'Occupied ', 'Parked ']	-999.1
['A', 7, 'Unoccupied', 'NotParked']	35.0662652773
['A', 7, 'Unoccupied', 'Parked ']	30.9
['A', 7, 'Occupied ', 'NotParked']	35.0662652773
['A', 7, 'Occupied ', 'Parked ']	-999.1
['A', 8, 'Unoccupied', 'NotParked']	30.5596387494
['A', 8, 'Unoccupied', 'Parked ']	20.9
['A', 8, 'Occupied ', 'NotParked']	30.5596387494
['A', 8, 'Occupied ', 'Parked ']	-999.1
['A', 9, 'Unoccupied', 'NotParked']	26.5036748741
['A', 9, 'Unoccupied', 'Parked ']	10.9
['A', 9, 'Occupied ', 'NotParked']	26.5036748741
['A', 9, 'Occupied ', 'Parked ']	-999.1
['B', 0, 'Unoccupied', 'NotParked']	71.2896750226
['B', 0, 'Unoccupied', 'Parked ']	-99.1
['B', 0, 'Occupied ', 'NotParked']	71.2896750226
['B', 0, 'Occupied ', 'Parked ']	-999.1
['B', 1, 'Unoccupied', 'NotParked']	80.81
['B', 1, 'Unoccupied', 'Parked ']	90.9
['B', 1, 'Occupied ', 'NotParked']	31.9961136273

['B', 1, 'Occupied ', 'Parked ']	-999.1
['B', 2, 'Unoccupied', 'NotParked']	71.81
['B', 2, 'Unoccupied', 'Parked ']	80.9
['B', 2, 'Occupied ', 'NotParked']	32.7570538608
['B', 2, 'Occupied ', 'Parked ']	-999.1
['B', 3, 'Unoccupied', 'NotParked']	62.81
['B', 3, 'Unoccupied', 'Parked ']	70.9
['B', 3, 'Occupied ', 'NotParked']	31.182297977
['B', 3, 'Occupied ', 'Parked ']	-999.1
['B', 4, 'Unoccupied', 'NotParked']	53.81
['B', 4, 'Unoccupied', 'Parked ']	60.9
['B', 4, 'Occupied ', 'NotParked']	28.0215841262
['B', 4, 'Occupied ', 'Parked ']	-999.1
['B', 5, 'Unoccupied', 'NotParked']	44.81
['B', 5, 'Unoccupied', 'Parked ']	50.9
['B', 5, 'Occupied ', 'NotParked']	23.8703409809
['B', 5, 'Occupied ', 'Parked ']	-999.1
['B', 6, 'Unoccupied', 'NotParked']	35.81
['B', 6, 'Unoccupied', 'Parked ']	40.9
['B', 6, 'Occupied ', 'NotParked']	19.4574244159
['B', 6, 'Occupied ', 'Parked ']	-999.1
['B', 7, 'Unoccupied', 'NotParked']	26.81
['B', 7, 'Unoccupied', 'Parked ']	30.9
['B', 7, 'Occupied ', 'NotParked']	16.6111789603
['B', 7, 'Occupied ', 'Parked ']	-999.1
['B', 8, 'Unoccupied', 'NotParked']	19.5679766392
['B', 8, 'Unoccupied', 'Parked ']	20.9
['B', 8, 'Occupied ', 'NotParked']	19.5679766392
['B', 8, 'Occupied ', 'Parked ']	-999.1
['B', 9, 'Unoccupied', 'NotParked']	22.8533073845
['B', 9, 'Unoccupied', 'Parked ']	10.9
['B', 9, 'Occupied ', 'NotParked']	22.8533073845
['B', 9, 'Occupied ', 'Parked ']	-999.1

The value of the terminal state is 1.

The resulting policy is shown in the next page.

Parameters 1: Safe driver Policy

['A', 0, 'Unoccupied', 'NotParked']	Drive
['A', 0, 'Unoccupied', 'Parked ']	Exit
['A', 0, 'Occupied ', 'NotParked']	Drive
['A', 0, 'Occupied ', 'Parked ']	Exit
['A', 1, 'Unoccupied', 'NotParked']	Park
['A', 1, 'Unoccupied', 'Parked ']	Exit
['A', 1, 'Occupied ', 'NotParked']	Drive
['A', 1, 'Occupied ', 'Parked ']	Exit
['A', 2, 'Unoccupied', 'NotParked']	Park
['A', 2, 'Unoccupied', 'Parked ']	Exit
['A', 2, 'Occupied ', 'NotParked']	Drive
['A', 2, 'Occupied ', 'Parked ']	Exit
['A', 3, 'Unoccupied', 'NotParked']	Park
['A', 3, 'Unoccupied', 'Parked ']	Exit
['A', 3, 'Occupied ', 'NotParked']	Drive
['A', 3, 'Occupied ', 'Parked ']	Exit
['A', 4, 'Unoccupied', 'NotParked']	Park
['A', 4, 'Unoccupied', 'Parked ']	Exit
['A', 4, 'Occupied ', 'NotParked']	Drive
['A', 4, 'Occupied ', 'Parked ']	Exit
['A', 5, 'Unoccupied', 'NotParked']	Drive
['A', 5, 'Unoccupied', 'Parked ']	Exit
['A', 5, 'Occupied ', 'NotParked']	Drive
['A', 5, 'Occupied ', 'Parked ']	Exit
['A', 6, 'Unoccupied', 'NotParked']	Drive
['A', 6, 'Unoccupied', 'Parked ']	Exit
['A', 6, 'Occupied ', 'NotParked']	Drive
['A', 6, 'Occupied ', 'Parked ']	Exit
['A', 7, 'Unoccupied', 'NotParked']	Drive
['A', 7, 'Unoccupied', 'Parked ']	Exit
['A', 7, 'Occupied ', 'NotParked']	Drive
['A', 7, 'Occupied ', 'Parked ']	Exit
['A', 8, 'Unoccupied', 'NotParked']	Drive
['A', 8, 'Unoccupied', 'Parked ']	Exit
['A', 8, 'Occupied ', 'NotParked']	Drive
['A', 8, 'Occupied ', 'Parked ']	Exit
['A', 9, 'Unoccupied', 'NotParked']	Drive
['A', 9, 'Unoccupied', 'Parked ']	Exit
['A', 9, 'Occupied ', 'NotParked']	Drive
['A', 9, 'Occupied ', 'Parked ']	Exit
['B', 0, 'Unoccupied', 'NotParked']	Drive
['B', 0, 'Unoccupied', 'Parked ']	Exit
['B', 0, 'Occupied ', 'NotParked']	Drive
['B', 0, 'Occupied ', 'Parked ']	Exit
['B', 1, 'Unoccupied', 'NotParked']	Park
['B', 1, 'Unoccupied', 'Parked ']	Exit
['B', 1, 'Occupied ', 'NotParked']	Drive
['B', 1, 'Occupied ', 'Parked ']	Exit
['B', 2, 'Unoccupied', 'NotParked']	Park

['B', 2, 'Unoccupied', 'Parked ']	Exit
['B', 2, 'Occupied ', 'NotParked']	Drive
['B', 2, 'Occupied ', 'Parked ']	Exit
['B', 3, 'Unoccupied', 'NotParked']	Park
['B', 3, 'Unoccupied', 'Parked ']	Exit
['B', 3, 'Occupied ', 'NotParked']	Drive
['B', 3, 'Occupied ', 'Parked ']	Exit
['B', 4, 'Unoccupied', 'NotParked']	Park
['B', 4, 'Unoccupied', 'Parked ']	Exit
['B', 4, 'Occupied ', 'NotParked']	Drive
['B', 4, 'Occupied ', 'Parked ']	Exit
['B', 5, 'Unoccupied', 'NotParked']	Park
['B', 5, 'Unoccupied', 'Parked ']	Exit
['B', 5, 'Occupied ', 'NotParked']	Drive
['B', 5, 'Occupied ', 'Parked ']	Exit
['B', 6, 'Unoccupied', 'NotParked']	Park
['B', 6, 'Unoccupied', 'Parked ']	Exit
['B', 6, 'Occupied ', 'NotParked']	Drive
['B', 6, 'Occupied ', 'Parked ']	Exit
['B', 7, 'Unoccupied', 'NotParked']	Park
['B', 7, 'Unoccupied', 'Parked ']	Exit
['B', 7, 'Occupied ', 'NotParked']	Drive
['B', 7, 'Occupied ', 'Parked ']	Exit
['B', 8, 'Unoccupied', 'NotParked']	Drive
['B', 8, 'Unoccupied', 'Parked ']	Exit
['B', 8, 'Occupied ', 'NotParked']	Drive
['B', 8, 'Occupied ', 'Parked ']	Exit
['B', 9, 'Unoccupied', 'NotParked']	Drive
['B', 9, 'Unoccupied', 'Parked ']	Exit
['B', 9, 'Occupied ', 'NotParked']	Drive
['B', 9, 'Occupied ', 'Parked ']	Exit

Discussion for Safe Driver:

As you can see, in the policy for this set of parameters, the policy:

- never parks at A[0] or B[0], for O = occupied/unoccupied
- the states where O is occupied, and P is not parked, the action is always “Drive”
- It only parks in unoccupied states (excluding A[0] and B[0])

The value functions have high negative values for states where,
O = occupied and P = Parked

The highest value (90.9) is for states A[1] and B[1], where O=unoccupied, P = parked. As the row numbers get larger, the value decreases. The lowest value is -999.1, for states where) = occupied and P = parked.

This makes sense to my intuition.

In the second parameter (driver on rampage) scenario, the rewards are as follow:

- Driving (not being parked) = -1
- Parking in handicapped = 100
- Crashing = 1000
- In other states there is a gradient of reward:
  (total_rows_in_aisle – current_row) * 10

  Thus smaller the current_row, higher the reward.
- The end state has a reward of 1
- When in P = parked, the action is always exit.

The results for this parameter setting is shown below.

Parameters 2: Rampage driver Value function

['A', 0, 'Unoccupied', 'NotParked']	647.296407375
['A', 0, 'Unoccupied', 'Parked ']	100.9
['A', 0, 'Occupied ', 'NotParked']	899.81
['A', 0, 'Occupied ', 'Parked ']	1000.9
['A', 1, 'Unoccupied', 'NotParked']	786.102776663
['A', 1, 'Unoccupied', 'Parked ']	90.9
['A', 1, 'Occupied ', 'NotParked']	899.81
['A', 1, 'Occupied ', 'Parked ']	1000.9
['A', 2, 'Unoccupied', 'NotParked']	788.361699799
['A', 2, 'Unoccupied', 'Parked ']	80.9
['A', 2, 'Occupied ', 'NotParked']	899.81
['A', 2, 'Occupied ', 'Parked ']	1000.9
['A', 3, 'Unoccupied', 'NotParked']	778.737958945
['A', 3, 'Unoccupied', 'Parked ']	70.9
['A', 3, 'Occupied ', 'NotParked']	899.81
['A', 3, 'Occupied ', 'Parked ']	1000.9
['A', 4, 'Unoccupied', 'NotParked']	765.243065219
['A', 4, 'Unoccupied', 'Parked ']	60.9
['A', 4, 'Occupied ', 'NotParked']	899.81
['A', 4, 'Occupied ', 'Parked ']	1000.9
['A', 5, 'Unoccupied', 'NotParked']	748.273879343
['A', 5, 'Unoccupied', 'Parked ']	50.9
['A', 5, 'Occupied ', 'NotParked']	899.81
['A', 5, 'Occupied ', 'Parked ']	1000.9
['A', 6, 'Unoccupied', 'NotParked']	726.999494845
['A', 6, 'Unoccupied', 'Parked ']	40.9
['A', 6, 'Occupied ', 'NotParked']	899.81
['A', 6, 'Occupied ', 'Parked ']	1000.9
['A', 7, 'Unoccupied', 'NotParked']	699.958381752
['A', 7, 'Unoccupied', 'Parked ']	30.9
['A', 7, 'Occupied ', 'NotParked']	899.81

['A', 7, 'Occupied ', 'Parked ']	1000.9
['A', 8, 'Unoccupied', 'NotParked']	664.935834842
['A', 8, 'Unoccupied', 'Parked ']	20.9
['A', 8, 'Occupied ', 'NotParked']	899.81
['A', 8, 'Occupied ', 'Parked ']	1000.9
['A', 9, 'Unoccupied', 'NotParked']	618.580926141
['A', 9, 'Unoccupied', 'Parked ']	10.9
['A', 9, 'Occupied ', 'NotParked']	899.81
['A', 9, 'Occupied ', 'Parked ']	1000.9
['B', 0, 'Unoccupied', 'NotParked']	718.516405586
['B', 0, 'Unoccupied', 'Parked ']	100.9
['B', 0, 'Occupied ', 'NotParked']	899.81
['B', 0, 'Occupied ', 'Parked ']	1000.9
['B', 1, 'Unoccupied', 'NotParked']	798.44906351
['B', 1, 'Unoccupied', 'Parked ']	90.9
['B', 1, 'Occupied ', 'NotParked']	899.81
['B', 1, 'Occupied ', 'Parked ']	1000.9
['B', 2, 'Unoccupied', 'NotParked']	784.477372342
['B', 2, 'Unoccupied', 'Parked ']	80.9
['B', 2, 'Occupied ', 'NotParked']	899.81
['B', 2, 'Occupied ', 'Parked ']	1000.9
['B', 3, 'Unoccupied', 'NotParked']	764.523179692
['B', 3, 'Unoccupied', 'Parked ']	70.9
['B', 3, 'Occupied ', 'NotParked']	899.81
['B', 3, 'Occupied ', 'Parked ']	1000.9
['B', 4, 'Unoccupied', 'NotParked']	735.714369272
['B', 4, 'Unoccupied', 'Parked ']	60.9
['B', 4, 'Occupied ', 'NotParked']	899.81
['B', 4, 'Occupied ', 'Parked ']	1000.9
['B', 5, 'Unoccupied', 'NotParked']	696.713803719
['B', 5, 'Unoccupied', 'Parked ']	50.9
['B', 5, 'Occupied ', 'NotParked']	899.81
['B', 5, 'Occupied ', 'Parked ']	1000.9
['B', 6, 'Unoccupied', 'NotParked']	650.665119807
['B', 6, 'Unoccupied', 'Parked ']	40.9
['B', 6, 'Occupied ', 'NotParked']	899.81
['B', 6, 'Occupied ', 'Parked ']	1000.9
['B', 7, 'Unoccupied', 'NotParked']	606.913926229
['B', 7, 'Unoccupied', 'Parked ']	30.9
['B', 7, 'Occupied ', 'NotParked']	899.81
['B', 7, 'Occupied ', 'Parked ']	1000.9
['B', 8, 'Unoccupied', 'NotParked']	579.3098836
['B', 8, 'Unoccupied', 'Parked ']	20.9
['B', 8, 'Occupied ', 'NotParked']	899.81
['B', 8, 'Occupied ', 'Parked ']	1000.9
['B', 9, 'Unoccupied', 'NotParked']	581.033449961
['B', 9, 'Unoccupied', 'Parked ']	10.9
['B', 9, 'Occupied ', 'NotParked']	899.81
['B', 9, 'Occupied ', 'Parked ']	1000.9

Parameters 2: Rampage driver Policy

['A', 0, 'Unoccupied', 'NotParked']	Drive
['A', 0, 'Unoccupied', 'Parked ']	Exit
['A', 0, 'Occupied ', 'NotParked']	Park
['A', 0, 'Occupied ', 'Parked ']	Exit
['A', 1, 'Unoccupied', 'NotParked']	Drive
['A', 1, 'Unoccupied', 'Parked ']	Exit
['A', 1, 'Occupied ', 'NotParked']	Park
['A', 1, 'Occupied ', 'Parked ']	Exit
['A', 2, 'Unoccupied', 'NotParked']	Drive
['A', 2, 'Unoccupied', 'Parked ']	Exit
['A', 2, 'Occupied ', 'NotParked']	Park
['A', 2, 'Occupied ', 'Parked ']	Exit
['A', 3, 'Unoccupied', 'NotParked']	Drive
['A', 3, 'Unoccupied', 'Parked ']	Exit
['A', 3, 'Occupied ', 'NotParked']	Park
['A', 3, 'Occupied ', 'Parked ']	Exit
['A', 4, 'Unoccupied', 'NotParked']	Drive
['A', 4, 'Unoccupied', 'Parked ']	Exit
['A', 4, 'Occupied ', 'NotParked']	Park
['A', 4, 'Occupied ', 'Parked ']	Exit
['A', 5, 'Unoccupied', 'NotParked']	Drive
['A', 5, 'Unoccupied', 'Parked ']	Exit
['A', 5, 'Occupied ', 'NotParked']	Park
['A', 5, 'Occupied ', 'Parked ']	Exit
['A', 6, 'Unoccupied', 'NotParked']	Drive
['A', 6, 'Unoccupied', 'Parked ']	Exit
['A', 6, 'Occupied ', 'NotParked']	Park
['A', 6, 'Occupied ', 'Parked ']	Exit
['A', 7, 'Unoccupied', 'NotParked']	Drive
['A', 7, 'Unoccupied', 'Parked ']	Exit
['A', 7, 'Occupied ', 'NotParked']	Park
['A', 7, 'Occupied ', 'Parked ']	Exit
['A', 8, 'Unoccupied', 'NotParked']	Drive
['A', 8, 'Unoccupied', 'Parked ']	Exit
['A', 8, 'Occupied ', 'NotParked']	Park
['A', 8, 'Occupied ', 'Parked ']	Exit
['A', 9, 'Unoccupied', 'NotParked']	Drive
['A', 9, 'Unoccupied', 'Parked ']	Exit
['A', 9, 'Occupied ', 'NotParked']	Park
['A', 9, 'Occupied ', 'Parked ']	Exit
['B', 0, 'Unoccupied', 'NotParked']	Drive
['B', 0, 'Unoccupied', 'Parked ']	Exit
['B', 0, 'Occupied ', 'NotParked']	Park
['B', 0, 'Occupied ', 'Parked ']	Exit
['B', 1, 'Unoccupied', 'NotParked']	Drive
['B', 1, 'Unoccupied', 'Parked ']	Exit
['B', 1, 'Occupied ', 'NotParked']	Park
['B', 1, 'Occupied ', 'Parked ']	Exit

['B', 2, 'Unoccupied', 'NotParked']	Drive
['B', 2, 'Unoccupied', 'Parked ']	Exit
['B', 2, 'Occupied ', 'NotParked']	Park
['B', 2, 'Occupied ', 'Parked ']	Exit
['B', 3, 'Unoccupied', 'NotParked']	Drive
['B', 3, 'Unoccupied', 'Parked ']	Exit
['B', 3, 'Occupied ', 'NotParked']	Park
['B', 3, 'Occupied ', 'Parked ']	Exit
['B', 4, 'Unoccupied', 'NotParked']	Drive
['B', 4, 'Unoccupied', 'Parked ']	Exit
['B', 4, 'Occupied ', 'NotParked']	Park
['B', 4, 'Occupied ', 'Parked ']	Exit
['B', 5, 'Unoccupied', 'NotParked']	Drive
['B', 5, 'Unoccupied', 'Parked ']	Exit
['B', 5, 'Occupied ', 'NotParked']	Park
['B', 5, 'Occupied ', 'Parked ']	Exit
['B', 6, 'Unoccupied', 'NotParked']	Drive
['B', 6, 'Unoccupied', 'Parked ']	Exit
['B', 6, 'Occupied ', 'NotParked']	Park
['B', 6, 'Occupied ', 'Parked ']	Exit
['B', 7, 'Unoccupied', 'NotParked']	Drive
['B', 7, 'Unoccupied', 'Parked ']	Exit
['B', 7, 'Occupied ', 'NotParked']	Park
['B', 7, 'Occupied ', 'Parked ']	Exit
['B', 8, 'Unoccupied', 'NotParked']	Drive
['B', 8, 'Unoccupied', 'Parked ']	Exit
['B', 8, 'Occupied ', 'NotParked']	Park
['B', 8, 'Occupied ', 'Parked ']	Exit
['B', 9, 'Unoccupied', 'NotParked']	Drive
['B', 9, 'Unoccupied', 'Parked ']	Exit
['B', 9, 'Occupied ', 'NotParked']	Park
['B', 9, 'Occupied ', 'Parked ']	Exit

Discussion for Rampage Driver:

As you can see, in the policy for this set of parameters, the policy:

- parks at A[0] or B[0], for O = occupied
  This is interesting. Since the reward for crashing is higher that for parking in handicapped spot, it will “Drive”, so that it can crash at a different state.

- the states where O is occupied, and P is not parked, the action is always “Park”

- In all states where O = unoccupied, for P = Parked/NotParked, the action is “Drive”

- When in P = parked, the action is always exit.

In this scenario, the driver just wants to crash into cars.

The value function has all positive rewards. The highes positive reward is 1000.9, which is for all states where O = occupiedm and P = Parked. The lowest value is for the farther most row, with O = unoccupied, and P = parked. This is because the probability of the row being occupied is lower farther from the store, and thus it cannot find enough cars to crash around. This value function and policy makes sense according to my intuition.