CS747 – Assignment 4 Report

Implementing Sarsa, Expected Sarsa and Q-Learning in Windy Grid worlds

Mahesh Abnave (203059010)

1. Algorithms

- Implemented following three algorithms:
    1. Sarsa
    2. Expected Sarsa
    3. Q Learning

2. Grid worlds

- The grid size is fixed to 7x10.
- The start state is fixed to [3,0] and end state to [3,7]
- The wind strength is fixed in follow pattern: [0,0,0,1,1,1,2,2,1,0]
- You can also choose from one for four different types of world characteristics:
    1. **Type 1:** windy + four moves (parameter value: windy)
    2. **Type 2:** windy + eight moves (parameter value: windy-king)
    3. **Type 3:** stochastic wind + eight moves (parameter value: stoch-wind-king) - in this noise exists only in windy columns
    4. **Type 4:** stochastic noise everywhere + wind + eight moves (parameter value: stochallcol-wind-king) - in this noise exists on all column

3. Corner cases

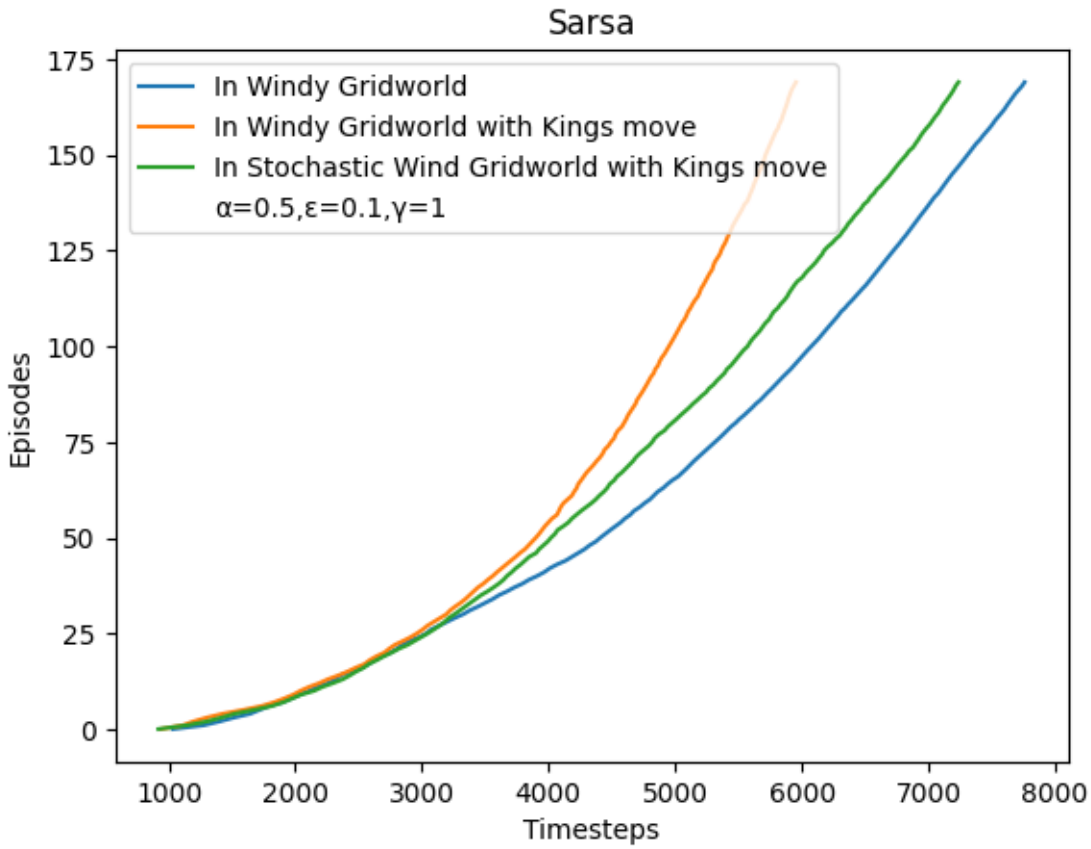
We don't allow movements to go outside grid world even after wind and stochastic noise effect.
If a final move (after considering wind and noise effect) is above grid top boundary, we bring it back to top most row of grid and likewise for all four edges of grid.

4. Plots

4.1. Sarsa plots without Type 4 grid world (as required in assignment statement)

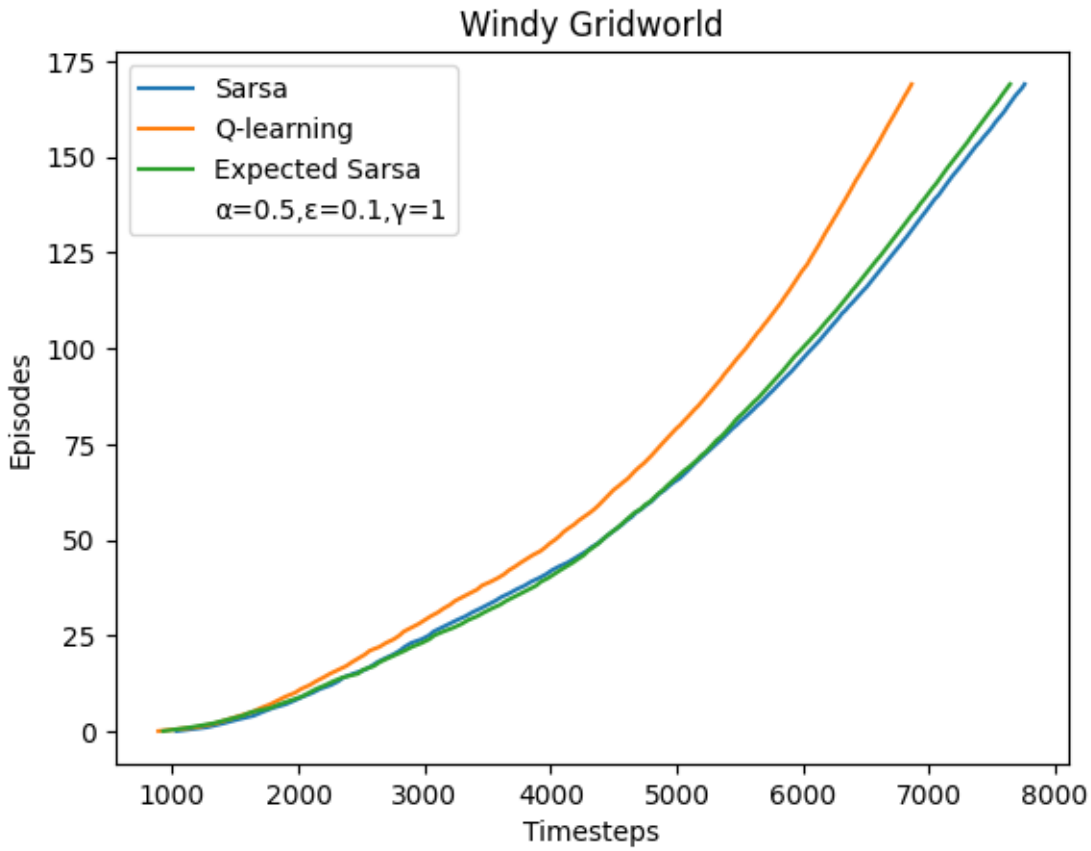
(Included in directory: 'six graphs without stochastic-everywhere/Sarsa.png')

These are plots which contains for all grid words except type 4 grid world (in which stochastic noise exists in all columns and not just in windy columns):



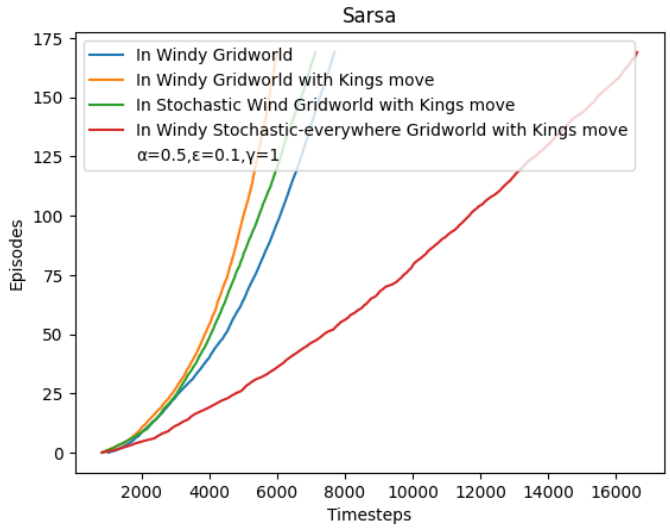
4.2. All algorithms plots in windy grid world (as required in assignment statement)

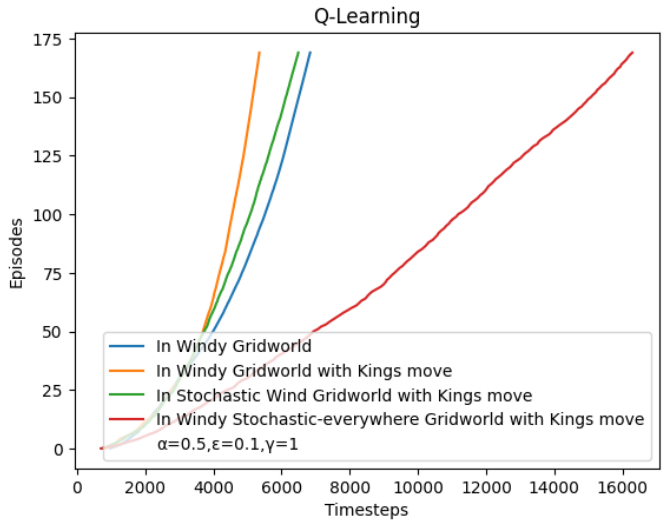
(Included in directory: 'six graphs without stochastic-everywhere/Windy Gridworld.png')

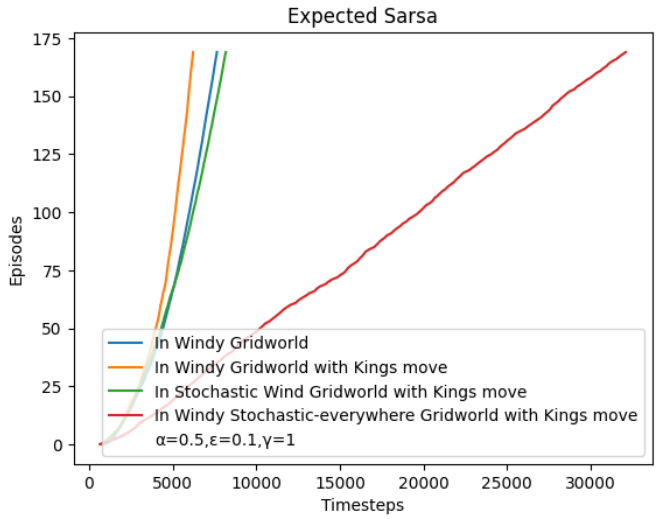


4.3. All algorithms in all grid words (including Type 4 grid world)

(Included in directory: 'seven graph with stochastic everywhere')







Observations

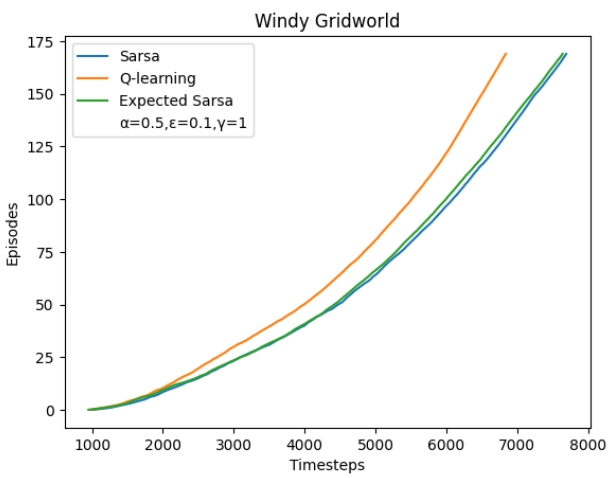
- Expected Sarsa is most affected by stochastic noise as it takes almost double times steps for Type 4 grid world (stochastic noise in all columns). This might be because of Expected Sarsa's stochastic (wighted) approach to calculate target value:

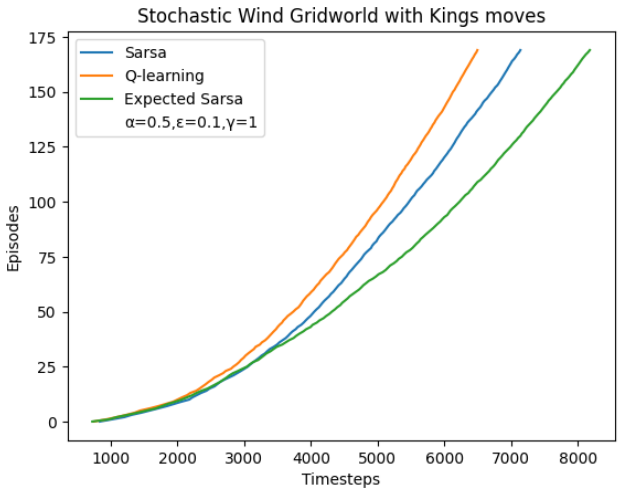
$$Target = r_t + \gamma \sum_{a \in A} \pi^t(s^{t+1}, a)\, \hat{Q}^t(s^{t+1}, a)$$

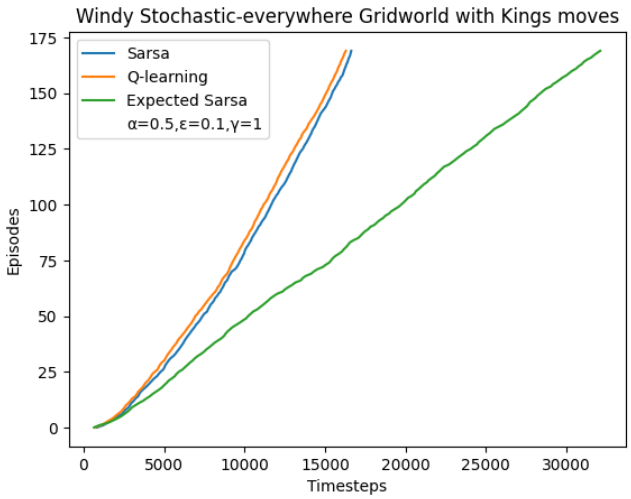
  Other algorithms does not seem to get affected as much as Expected Sarsa dur to stochastic noise. Thus Expected Sarsa is worst for stochastic environments.
- Surprisingly, Sarsa and Q Learning performs better with stochastic noise only in windy columns (with Kings move) than without any stochastic noise (with no Kings move). Thus, Kings moves help these two algorithms overcome performance impact of stochastic noise in windy columns. Expected Sarsa continues to perform poor with stochastic noise even in just windy columns (possibly due to same reason explained in above point).
- All algorithms performs their best in Windy Grid world with eight (Kings) moves
- All algorithms performs their worst in Windy Grid world with eight (Kings) moves and stochastic noise in all columns.
- For Sarsa, average time steps for last ten episodes in windy grid world with four actions is 17.4.
- For Q-Learning, average time steps for last ten episodes in windy grid world with eight action is 7.5

4.4. All grid worlds with all algorithms

(Included in directory: ‘seven graph with stochastic everywhere’)







Observations

- Adding stochastic noise “only to windy columns” have lesser performance impact on all algorithms than adding stochastic noise to “all columns”, as it can be seen that the later takes (approx.) four times more time steps than earlier for expected sarsa and (approx.) twice as much times steps for other algorithms.
- Expected Sarsa performs significantly poor than Sarsa when stochastic noise is added (be it to only windy columns or to all columns), possibly due to the same reason explained in first point in earlier set of observations.
- Q-Learning perform better than both other algorithms in all grid world.

5. Running Experiment Utility

5.1. Obtaining help for running utility (runs only for single seed)

Utility help can be obtained by running script with `-h` option:

 # python gridworld.py -h

```
usage: gridworld.py [-h] [-a ALPHA] [-e EPSILON] [-g GAMMA] [-s SEED] [-p EPISODES] -l {sarsa,ql,esarsa} -w
{windy,windy-king,stoch-wind-king,stochallcol-wind-king} [-pf PLOT_FILE] [-of OUTPUT_DATA_FILE]

Simulates Windy Gridworld problem. The grid size is fixed to 7x10.
The start state is fixed to [3,0] and end state to [3,7]
The wind strength is fixed in follow pattern: [0,0,0,1,1,1,2,2,1,0]
with leftmost 0 corresponding to index-0 column.
You can specify which of three algorithms to run:
1. sarsa (sarsa)
2. expected sarsa (esarsa)
3. q-learning (ql)
You can also choose from one for four different types of world characteristics:
1. windy + four moves (windy)
2. windy + eight moves (windy-king)
3. stochastic wind + eight moves (stoch-wind-king) - in this noise exists only in windy columns
4. stochastic noise everywhere + wind + eight moves (stochallcol-wind-king) - in this noise exists on all columns

optional arguments:
  -h, --help            show this help message and exit
  -a ALPHA, --alpha ALPHA
                        Learning rate (default: 0.5)
  -e EPSILON, --epsilon EPSILON
                        Used for epsilon greedy policy (default: 0.1)
  -g GAMMA, --gamma GAMMA
                        Discount factor (default: 0.5)
  -s SEED, --seed SEED  Seed for random number generator (default: 42)
  -p EPISODES, --episodes EPISODES
                        Number of episodes to run (default: 170)
  -l {sarsa,ql,esarsa}, --algo {sarsa,ql,esarsa}
                        Algorithm to run
  -w {windy,windy-king,stoch-wind-king,stochallcol-wind-king}, --gridworld {windy,windy-king,stoch-wind-
king,stochallcol-wind-king}
  -pf PLOT_FILE, --plot-file PLOT_FILE
                        Name of png file for storing plot (default: plot.png)
  -of OUTPUT_DATA_FILE, --output-data-file OUTPUT_DATA_FILE
                        Name of output file for storing simulation output (default: output.txt)
```

5.2. Obtaining all plots

These plots are included in all-plots directory.

All plots can be obtained by running all following script:

 # python multirun.py

This script runs all three algorithms on all four gridworlds each with ten seeds. The output will be seven plots (formed from average outputs from ten seeds per algorithm per grid world):

1. **Windy Gridworld.png** – contains plots of all three algorithms in windy, four moves grid world without any stochastic noise
2. **Windy Gridworld with Kings move.png** – contains plots of all three algorithms in windy, eight move grid world without any stochastic noise
3. **Stochastic Wind Gridworld with Kings move.png** – contains plots of all three algorithms in windy, eight moves grid world with stochastic noise **only in windy columns**
4. **Windy Stochastic-everywehre Gridworld with Kings move.png** – contains plots of all three algorithms in windy, eight move grid world with stochastic noise **in all columns**
5. **Sarsa.png** – Plots of Sarsa in all above four types of grid worlds
6. **Expected Sarsa.png** – Plots of Expected Sarsa in all above four types of grid worlds
7. **Q-Learning.png** – Plots of Q Learning in all above four types of grid worlds

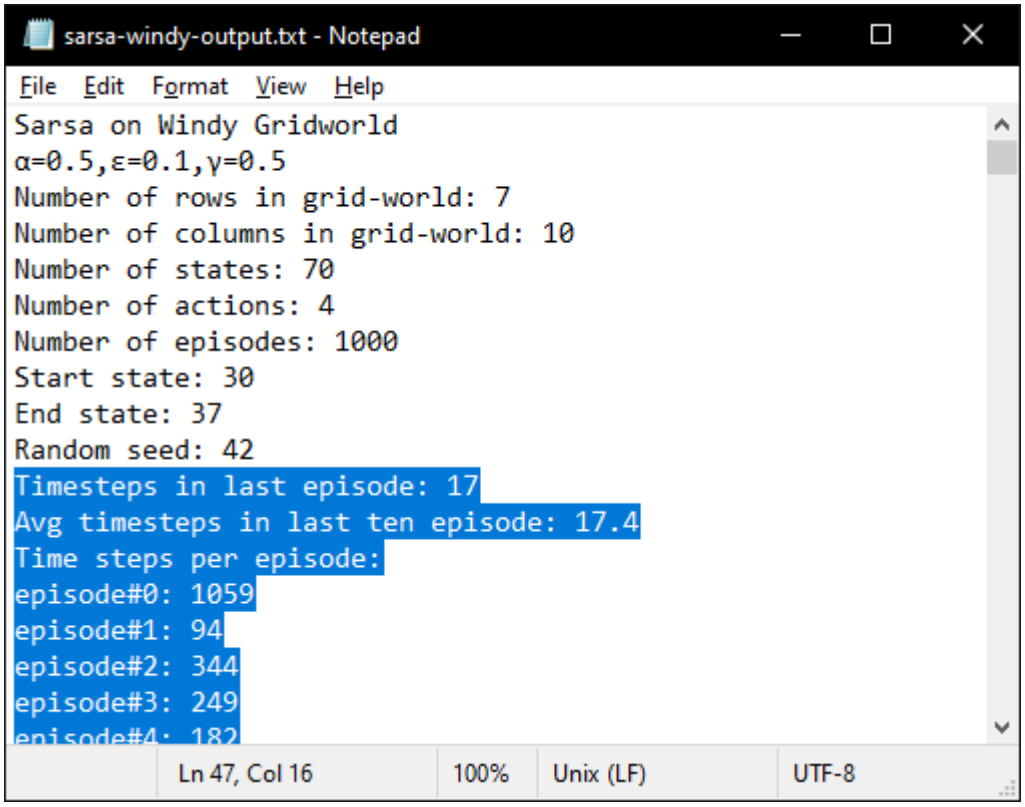
5.3. Example – sarsa in windy + four moves gridworld

Command:

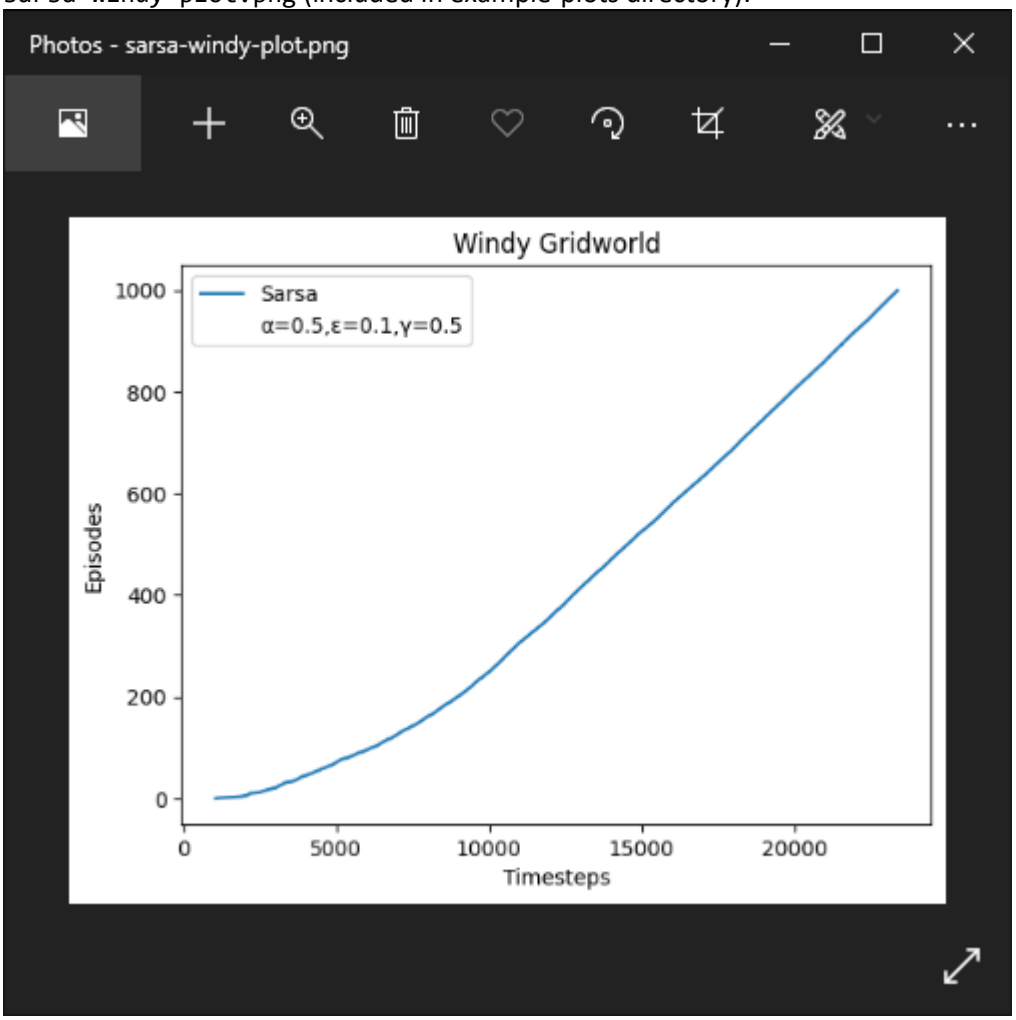
```
# python gridworld.py -l sarsa -w windy -p 1000 -pf='sarsa-windy-plot.png' -
of='sarsa-windy-output.txt'
```

This creates two files:

- sarsa-output.txt (included in example-plots directory):
  This file contains all inputs along with three possibly important observations: timesteps in last episode, avg timestep in last ten episodes and timesteps of all episodes as highlighted below



- `sarsa-windy-plot.png` (included in example-plots directory):



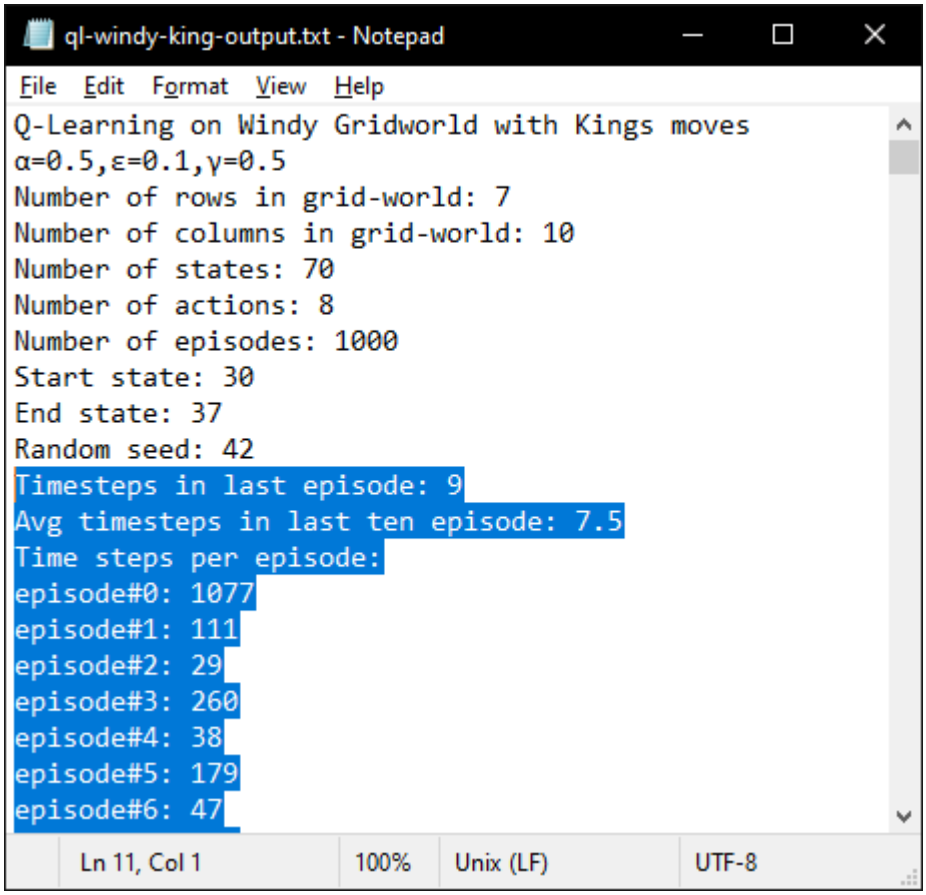
5.4. Example – Q-learning in windy + eight moves gridworld

Command:

```
# python gridworld.py -l ql -w windy-king -p 1000 -pf='ql-windy-king-plot.png' -
of='ql-windy-king-output.txt'
```

This creates two files:

- ql-windy-king-output.txt (included in example-plots directory):



- ql-windy-king-plot.png (included in example-plots directory):