**CS7641 Unsupervised Learning HW4**

Aleksandr Plokhikh ([aplokhikh3@gatech.edu](mailto:aplokhikh3@gatech.edu))

**Introduction, dataset selection, data preprocessing**

The goal of the assignment is to study solutions for Markov Decision Problems (MDP), via model based algorithms such as Value Iteration (VI), and Policy Iteration (PI), and model free reinforcement learning algorithm.

As a first problem I took Frozen Lake model, since this model works with multidimensional world (in the general case), and have uncertainty and stochasticity in movements. It lays a fundament to robotics, car autopilots operation, etc. All futurologist of the past (until 2022) thought that the next breakfruth in AI will be in robotics, and there were prerequsits for it like work of Boston dynamic, but great leap came from the completely other thing of AI fild, and robotics is lagging behind probably because to teach it we need a real interaction with the physical world, good exploration-exploatation models and balance, and we can not achive it with extencise calculations on GPUs. I used 8x8 Frozen lake model with 64 states which considered to be a small problem. In my work I will mainly use analogy not with frozen lake but with The Lord of The Rings. Map is the map of middle earth and we need to deliver thr ring of power from Shire to Orodruin. I think it is more live, and amusing analogy also it is much easier to explain the concept of discounted reward here – the longer we go to Orodruin the less Middleerath dweller we will save during the war with Saruron, and Stronger the Sauron – the faster we need to destroy the ring and make more riskier movements

The second problem is Black Jack. Of cause the problem have been solved a million times before and on average you can not beat the house, but we probably all saw movie “21” where guys counted cards in Las Vegas and turned result expectancy on their side. Release of othe cards from the deck changes the actual value of the MDP state. We can add realsed cards as a prehistory and make new MDP states based on it. The state space in this case can be as high as trillions instead of the basic 290, which is not really true as a basic model takes all cards with replacement, which does not happen in the real life, modern casinos increase number of decks for increasing their own margine, and the sampling without replecment become close to drawing cards with replacement but it still not the case and we need to ake it into account and use more complex moddels. MDP with trillions states based on prehistory of released cards can be useful for finding the best strategy but it will be vary hard to find one, so best judjment should be applied, and simple counting like in the movie can be helpful and increases number of states to only several thousands, while providing a better strategies than the default one with no knowledge of prehistory. I do not things that guys in the movie actually solved MDP for most popular “hot” states of the deck so it will be interesting to evaluate overall. Plus basic strategies do not push player to the end of the game faster, and treats the world as we have unlimited time (works with gamma=1) but in the real life aand as in case of movie “21” if you count cards, the casino security may come after you, and you may have only few minutes left to finish several games and escape with prize before got bitten by security team. In thi cases we may want o finish games more quickly and sacrifice the persue of optimal strategy which may require reveal of additional cards. For that we can solve MDP with discounted rewards (gamma<1). So black jack problem is more interesting, undiscovered, and hard than it may seem. In the scope of the assignment I will explore only standard case with 290 states which is already considered big according to asiignment note, and leave much harder cases for further investigation.

**Frozen lake value iteration**

A graph of numbers and a diagram

Description automatically generated with medium confidenceA colorful squares with numbers and symbols

Description automatically generatedA graph with numbers and lines

Description automatically generatedA graph with numbers and lines

Description automatically generated

Fig.1 Value iteration on frozen lake problem with no discontinued reward (gamma=1)

At condition of no discontinued reward (gamma=1) the state value of the start point is 1 which is equal to the reward of the end which says that if we do not have discontinued reward and allowed to play as long as we want we are guarantied to finish the game at the goal, and will not fall in the hole. The holes have state value of 0 which is obvios since game do not continue if we reach this states. Since on the whole map only goal state has non zero immediate reward than positive state values gradually spread across the map with every step of VI, until saturation with logarithm like manner which is seen on the plot of median state value vs number of iterations. Algorithm takes 569ms and 1425 iteration to converge with the standard epsilon of 1e-10. Mean state values do not change much after already 300 iterations. With tightening/loosing epsilon requirents the convergence time will exponentially increase/decrease respectively which clearly visible form the mean state value graph which clearly has logarithmical behavior. No epsilon skew is interesting to make here the behavior is obvious. What interesting to see is when the policy convergence to the optimal one because it is always nice to know the precise expectancy reward at the state but it may not worth extra calculation cost if further improvement of the state value to the true one does not change policy which may be already optimal, and this extra calculations in this case does not improve our performance in the game since we use the same policy anyway, and will cause just a waste of resources on computation, and the plot with policy difference shows that the policy converges to the optimal one in only 365 iterations, and we need just 156ms for it, which is significantly less than the number for state values convergency.

On the policy map proposed movement to the left denoted as 0, down-1, right-2, up-3. If we look closely at the bottom left corner of the map we see that the algorithm propose to avoid holes at all cost and do on the right side trying navigate agent via road with holes to the goal with short path (does not go in the Mines of Moria in Khazad-dum, due to the presence of Balrog there). Instead it poposes to “hit” the wall and assume that due to stochasticity of the world we will eventually reach the top of ther map and will go around all holes to the goal, sticking to the boarder of the map. On the right side of the map the agent also propose to “hit” the wall and rely on the stochastisity that should eventually lead us to the goal – Orodruin to destroy the ring as in case if we have unlimited time to do so. So the agent acts like the initial plan of Gandalf – just go around -avoid pitfalls, and we guarantied to destroy the ring of power and save everyone if Sauron did not start the war yet.

A graph of numbers and a diagram

Description automatically generated with medium confidenceA colorful chart with numbers and symbols

Description automatically generated with medium confidenceA graph of numbers and symbols

Description automatically generated with medium confidenceA graph of numbers and a bar chart

Description automatically generated with medium confidence

A colorful squares with numbers and symbols

Description automatically generatedA colorful squares and numbers

Description automatically generatedA colorful squares with numbers

Description automatically generated with medium confidenceA colorful squares with numbers

Description automatically generated

Fig.2 Value iteration on frozen lake problem with with discontinued reward (gamma=1, 0.99, 0.9, 0.3)

Complettely other things starts to happen if Sauron started the war and we can not save everyone. Evry step Sauron kills some present of the middle earth dwellers. Not everyone can be saved. If gamma=0.99 we kill 1% f middle earth population every step, and state value of all states dramatically drops down. The further state from the finish – the higher the drop. If we start at shire only 41% of all dwellers can be saved – it is completely different game, and our strategy changes as well. Now on the left border of the map on the top tilses which are pretty away from holes we do not hit the wall but go up in order to go around faster, here we take the rist to come to the holes closer to save more lives, but this change does not make to much sence since we should not apper at this tiles if we stick with the optimal strategy. The most minigfull changes is happening on the right top corner -with the new gamma=0.99 we stop hitting the wall in the states which are far away from holes (not very risky states) and go along the wall rather than hitting it in order to save time. If we decrease our gamma even further to 0.9 we can only save about 1% of middle earth population, the changes to the policy map become more prominent, and remain the same character, now more tiles on the top right and left corners change their direction from hitting the wall to go along it. The states which were considered to be risky to change at gamma=0.99 is not risky that much anymore considering higher discont, and now subject to change. On the mean we reach goal faster but have more chances to die in the hole. Gamma=0.3 is a very special case here because discount rate of 0.3 is less than the probability of successful movement. Here there should not be any greater risk to the reward than losing time, and indeed with such discount rate the agent propose to go right away in the “Mines of Moria” dash frouth the middle ground of the map with all holes and finish the game earlier- try to destroy the ring as soon as possible regardless of all pifalls there. The agent navigates us between all the holes.

Since Gamma greatly reduces further reward and state values it is no surprise that state values converges to stable values in low amount of time, and iterations at given epsilon convergence criteria (see Fig 3). What less obvious is why policy also converges faster but if we look deeper with discounted reward we just need to know the values of states in the shorter vicinity to build a right policy do to the discontinued reward of later states, and since VI gradually spread the rewards through the map step by step from the states with immediate reward and game stop tiles to the rest of states. Here we need less steps for good enough state values which provide a stable policy. Then we decrease gamma from 1 to 0.99, to 0.9 to 0.3 the amount of required iterations for state values convergence reduces in the row 1426, 662, 158, 17 respectively, while the number of iterations needed for policy convergence reduces in the row 365, 127, 48,16. Interesting to see that for gamma 0.9-1 we need 3-5 times more iterations for state value convergence when policy convergence which does make more metter than state convergence while for 0.3 policy and state convergence happening almost at the same time probably because a very low mean state valkuse at this case since high discount in the reward and epsilon/mean state value is high here, and the relative convergence margin is orders magnitude higher than at gamma 1-0.9

A graph with a line and a point

Description automatically generated with medium confidenceA graph with a line and a dotted line

Description automatically generatedA graph with a line and a point

Description automatically generated with medium confidenceA graph with a line and a point

Description automatically generated with medium confidence

Fig.3 VI state values, and policy convergence on frozen lake problem with with discontinued reward (gamma=1, 0.99, 0.9, 0.3)

**Frozen lake Policy iteration**

PI – is another model based method to solve MDP. Each iteration of the algorithm is much more costly than in case of VI since each time we need to solve system of linear equations, but in this case true state values propagate though the state space much quickly via randomly preselected policy at each step. This leads to faster convergency than VI in terms of iterations but real time cost may vary. Initial randomness in generation the first policy map may greatly affect the convergence speed, so I modified the algorithm to parse random seed to it.

A graph of colored squares

Description automatically generatedA colorful squares with numbers and symbols

Description automatically generatedA graph with a line

Description automatically generatedA graph with a blue line

Description automatically generated

Fig.4 Policy iteration on frozen lake problem with no discontinued reward (gamma=1)

PI converges to exactly the same policy as VI at gamma=1, and state values are exactly the same with accuracy of epsilon, and we can not expect any other behavior as both algorithms are model based and do converge to the true states and optimal policies, no surprise here. What interesting to see is how fast it happening compared to VI, and indeed PI needs only 9 iterations for state values convergency, and 7 iterations for convergence to the stable policy (with VI we needed 1425, and 365 iterations respectively). Unfortunately it did not transferred to faster run time but on contrary it took more time run the algorithm -2.36s vs 1.28 for VI. The shape of the “learning curves” are very similar to the one in case of VI. The shape of the mean value curve is fast rising and remind logarithmical curve but not as close as in case of VI where we have a specific sate values update algorithm. PI is different here and many things depend on the initial policy which drives state values updates on first iteration of the algorithm.

State values at different discount rate almost exactly match the once from VI ran. But the policy map at gamma=0.3 is very slightly different. The only tile which is different is ironicaly the start state and due to it he difference is very crucial. VI proposes to go right which is sane movement at such discount rate but PI proposes to go down, and it will lead to harder time navigating between the holes. Such artifact is occurred because the epsilon of 1e-10 here is not infinitely small but is actually greater that the value of the start state which is 9.7e-12, and it lead to not really converged map, and it is surprising that with such ratio of epsilon to the state value all other polices are the same! Extra error counld also erose from the solving of the system of linear equations in PI, since the dtype of the map is only 64bit and we already have values of 9.7e-10 which leads to fast accumulation of the error in calculation which lead to mistake in the policy map. So at aggressive discount rate we need to use much lower epsilon, and igher bit values.

A graph of colored squares

Description automatically generatedA colorful chart with numbers and symbols

Description automatically generated with medium confidenceA graph of numbers and symbols

Description automatically generated with medium confidenceA graph of numbers and a bar chart

Description automatically generated with medium confidence

A colorful squares with numbers and symbols

Description automatically generatedA colorful squares with numbers and symbols

Description automatically generated with medium confidenceA colorful squares with numbers and symbols

Description automatically generated with medium confidenceA colorful squares with numbers and symbols

Description automatically generated

Fig.5 Policy iteration on frozen lake problem with with discontinued reward (gamma=1, 0.99, 0.9, 0.3)

It is also interesting to see is how much time and how many iterations of the algorithm do we need for state values and policy convergence. Runtime did not give us any surprises – the behaviour the same as in case of VI. The run time almoist exponentialy decase with decrease of gamma, but iterations plot gave us a peculiar artifact the run with gamma=0.99 requred the leas amount iterations for state values, and policy convergence this observation almost surtenly rose rose from the random nature of the initial map.

A graph with blue lines and numbers

Description automatically generatedA graph with a line and a point

Description automatically generatedA graph with blue lines and numbers

Description automatically generatedA graph with a line and a dotted line

Description automatically generated

Fig.6 PI state values, and policy convergence on frozen lake problem with with discontinued reward (gamma=1, 0.99, 0.9, 0.3)

Frozen lake Policy iteration

For the RL part of the assignment I used Q learning algorithm with fixed epsilon, and alpha without decay for better comparison during varying thehyper parameters.

A graph of numbers and letters

Description automatically generated with medium confidenceA graph of numbers and letters

Description automatically generated with medium confidence

Fig.7 Q learning state value, and policy convergence

One of the main aspects of reinforsment learning and Q learning in particular is exploration -exploatation dilemma -for quickly learning the best policy we have to make a lot of random actions, but choosing many random actions preveints us from taking benefit from learned strategy. Q learning perfecting the value state and policy not only when we chose random action but also when we follow the existion policy which helps polish and finetune it, and it is interesting to see wheat is the fastest way to learn the best policy- do only explorationaand make random choises everytime or find some sweet spot between random choises and fine polishing of the existing data. In Q learning epsilon is part of random choises in the decision making process. At at gamma=1 epsilon=1 the mean sate value starts to rise right away the positive state values starts to spread out frm the finish state. The rise pace is moderate and tapers out at the values around the ones we saw with model-based VI, and PI algorithms. The rise is logarithimic-like but not as sharp as in case of VI. The policy is also starts to perfect it-self during learning process almost immidietely, and probably the best than all other rund on the late stages of learning, but never matches the ideal policy obtained by VI, and PI algorithms, while at the late stages the number of altered states is low (2-12) it never reaches the ideal policy and the policy is very unstable as in all other cases.

A graph of numbers and a diagram

Description automatically generated with medium confidenceA graph of numbers and colors

Description automatically generated with medium confidenceA graph of colored squares

Description automatically generatedA graph of colored squares

Description automatically generated

A colorful squares with numbers and symbols

Description automatically generatedA colorful squares with numbers and symbols

Description automatically generated with medium confidenceA colorful squares with numbers and symbols

Description automatically generated with medium confidenceA graph of a puzzle

Description automatically generated with medium confidence

Fig. 8 Value states and policy map comparison between VI and Q learning with different epsilon

When we reduce epsilon to 0.5 learning behaviour changes dramatically – mean state values does not change until 70000 iterations mark. I think it happens due the fact that on the map there is only 1 state with non zero positive immediate reward – the goal state and it is tha last state in the numpy array, but in the q learning algorithm then we do a sane move we chose the one with the highest Q value, but when all qvalues are 0 we do a tie break an here we chose the state which is closer to the beginning of the numpy array, and it significantely delays the initial learning (steep learning curve), but once the map picked up some nonzero-state values, the “sane” pasr of Q-learning quickly starts to spread positive states value across the map and for some time from 75000 to 150000 iterations this agent outperform the completely randon Q learner with epsilon=1 in tersms of mean state value, after 150000 their performance match, the same story happens with policy perfection – there is sdelay with polyci update up to 70000 iterations, and after this freshfold policy updates to more reasonable one with altered states similar to the learner with epsilon=1. When we sightly decrease epsilon to 0.45 it moves the “learning threshold” significantly further to 120000 iterations mark, at epsilon=0.4 the threshold is already close to 50000 and goes beyond the scope of reasonable observation