Project Report #1

Team Name: Runtime Error
Project Title: Foraging in a Field
Project #: 3
Team Member Names: Roll No
Dibyojyoti Sinha: 180244
Milind Nakul: 180420
Satvik Jain: 180678
Samarth Varma: 180655

1 Introduction

Foraging has been an essential activity for survival in the wild. In our case, the environment contains patches of berries that are non-renewable, the agent loses reward for every movement and dies after a certain amount of time. The goal of our agent is to find an optimum foraging policy that maximizes the reward in a static environment where the resources are limited and there exists a moving cost but no cost when the agent is at rest. The agent can choose to leave the current patch of berries to explore for more patches or stop moving and conserve its rewards collected so far. This setting can be compared with the consumption of rare natural non-renewable resources, using these do generate wealth but also requires an effort to search and extract.

2 Related Work

Kolling and Akam [2017] explores model-based average reward reinforcement learning to provide a common framework for understanding different foraging strategies. The Marginal Value Theorem (Charnov [1976]) predicts the optimal patch leaving strategy when the rewards within the patch decrease monotonically. However, MVT makes sub-optimal decisions when the rewards may be stochastic or when the current reward is not a good estimator of the future rewards. Moreover, MVT does not account for the cost of searching for new patches and the changing trade-off landscape between spending time searching for a new patch and getting smaller but positive rewards by consuming the current patch. The latter makes more sense when the agent is aware that the episode is about to end.
The agent is required to navigate the environment, and thus it is essential to remember the positions of the already explored regions and patches. Ramezani and Lee [2018] explore a memory-based reinforcement learning algorithm to autonomously explore in an unknown environment. One such memory-based algorithm is Episodic Memory DQN (EMDQN) proposed by Lin et al. [2018].

In Volodymyr Mnih [2013], the authors have used the Deep Q Network algorithm to train an agent to play several Atari 2600 games. It uses a Convolutional Neural Network along with the Q-Learning algorithm to train the agent. Hado van Hasselt and Silver [2016] uses the Double Deep Q Learning algorithm to overcome the maximization bias of the traditional Q-Learning algorithm and trained an agent to play Atari games. Ziyu Wang [2016] introduced a new neural network architecture for model-free reinforcement learning. It uses two estimators one for the state value function and one for the state-dependent action advantage function. It helps in generalizing learning across actions. The dueling network automatically produces separate estimates of the state value function and state-advantage function, without any extra supervision. Since the research problem proposed is very similar in nature to the Atari games, it makes sense to look into these works and come up with a similar technique to solve the problem.

3 Problem Statement

As described above, our goal is to find an optimal solution for collecting berries from a field. The field has berries spread in randomly distributed patches. The agent's goal is to collect the maximum amount of juice (which will be its reward) in a given time frame by moving in 8 directions (N, NE, E, SE, S, SW, W, NW). The agent can increase the amount of juice by collecting more berries, and it will lose the juice while moving around in the field (at a constant rate). Depending on the size, there are four kinds of berries in the field. The bigger the berry, the more is the amount of juice the agent will get by collecting it (linear relation between berry size and the amount of juice it contains).

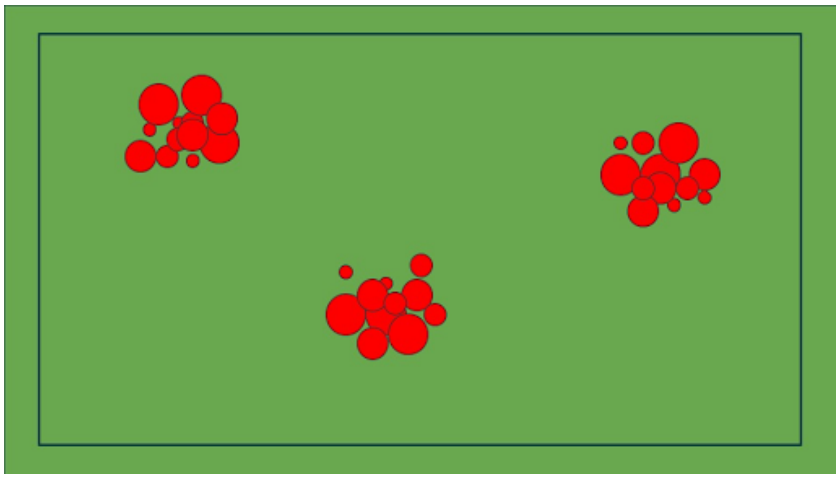


Figure 1: Visualization of the Problem Description

4 Environment Details and Implementation

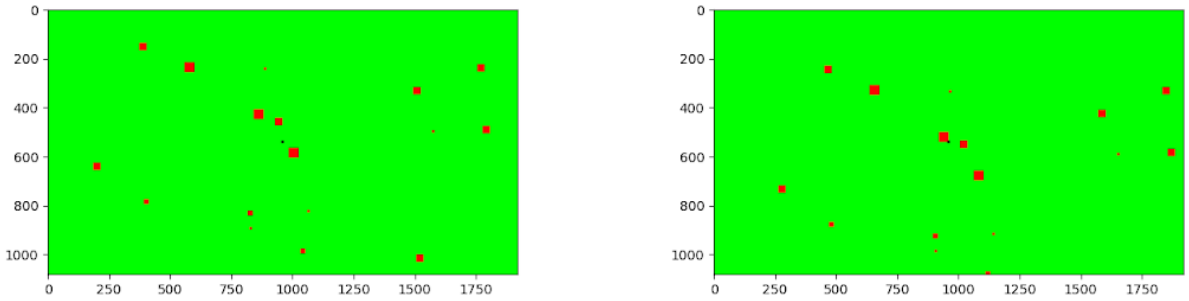
Following are the technical details of the environment:

- Field size: (20000, 20000)
- Patch size: (2600, 2600)
- Berry sizes: (10, 10), (20, 20), (30, 30) and (40, 40)
- Number of patches: 10
- Number of berries: 20 berries of each size in each patch ,i.e. Total 800 berries, 200 berries of each size
- Total time: 5 min = 300 sec
- Agent size: (10, 10)
- Agent speed: 400 pixels/sec
- Max steps: (Agent speed)*(Total time) = 120000
- Action space: (No move, N, NE, E, SE, S, SW, W, NW)
- Observation space size: (1920, 1080)
- Initial reward: 0.5
- Drain rate: $\frac{0.5d}{120*400}$, where 'd' is the distance moved in pixels
- Reward factor: $\frac{s}{1000}$, where 's' is the size of the berry collected

Table 1 describes the MDP of the environment:

State space (S)	Set of all the matrices of size 1080*1920, where all the values of each matrix are either 0 (representing field) or 1 (representing berries)
Initial states (S_θ)	Singleton set (Initial state is fixed)
Action Space (A)	Set of size 9 where action 0 corresponds to no-move and actions 1 to 8 correspond to a move in one of the 8 directions
Transition function (T)	The probability of landing in a particular state (s') when the agent is in state (s) and takes action (a) is either 1 (for one state) or 0 (for all other states)
Reward (R)	$-1 * drain\_rate + \sum_{collectedberries} berry\_size * reward\_factor$
Horizon (H)	Finite Horizon (120000) - Agent is aware that the task will terminate in finite time steps

Table 1: MDP for the environment



(a) (b)

Figure 2: Rendered Images of the Environment

The above images show the implemented environment for two different locations. The berries and the agent are both implemented as squares. An animation of the open-ai gym environment can be found here.

5 Future Directions: Prospective Solutions

After the implementation of the environment, the immediate next step is to implement a baseline agent using the algorithms such as Deep Q Networks(Volodymyr Mnih [2013]), Double Deep Q Networks(Hado van Hasselt and Silver [2016]) , and the Dueling Deep Q Networks(Ziyu Wang [2016]) in order get to an optimal solution for the problem. We then compare it with solutions from actual human subjects that are available from the Neuroscience experiments. We plan on using a Convolutional Neural Network architecture since we have a continuous observation space. However one of the feedbacks that we have received right now is that in our task, using a Convolutional Neural Network directly would be computationally infeasible. Therefore we plan to reduce the training time for the methods proposed by compressing our environment and the corresponding observation space to a satisfying level. We will have to reduce the size of the berries as well so that the relative reward of the berries of the different sizes is the same. We could also try a method called frame skipping(Volodymyr Mnih [2013]) in which a new action is chosen only after k steps and in those k steps the same action is repeated . This may help in reducing the

computational cost. The choice of k would depend on the nature of the problem we are trying to solve and hence be decided while training the agent.

We also plan on using a working memory for our agent. This would work such that the agent can remember the parts of the field that have already been explored and hence for further steps avoid those areas of the field. This would help us in implementing a memory-based reinforcement learning agent. We could try and make the memory volatile such that the agent forgets which parts of the field have been explored. This would mimic the behavior of the human test subjects and therefore help in the comparison of the optimal policy of the reinforcement learning agent and the policy of the humans.

An issue with compressing the observation space is that we are limited by the minimum berry size. Thus we will also explore the possibility of further reducing the input space by identifying and inputting the directions to the visible berries and the corresponding distances in form of a matrix to the agent. Thus the input features can be the following tuple: *[is-berry, direction, distance, berry-size]*. Where *is-berry* takes the value as 1 if the tuple contains information about a berry and 0 otherwise. *direction* is a unit vector towards the berry's center and *distance* is the corresponding Euclidean distance.

To maintain a fixed input size we have two approaches, We can divide the observation space radially into N segments with the agent in the center. This will give us an observation space in form of an array of size (N, 6). Each of these segments will correspond to an entry in the input array. Those segments without berries can contain a placeholder value with *is-berry* set to zero. Alternatively, We can append the information of all the visible berries in an array of large enough size and fill the remaining space with placeholders with *is-berry* set to zero.

We may also explore the case when the entries in the above-mentioned reduced observation space are ordered by the directions in the order *N, NE, E, SE, S, SW, W, NW*. This might help in training since rows of the input array would represent fixed directions. Also, such an ordering may allow us to hierarchically compare between berries and groups of berries.

6 Member Contributions

Below are the member contributions till now for the project.

- Environment - Satvik Jain
- Rendering and Environment Testing - Samarth Varma and Satvik Jain
- Research Work - Milind Nakul
- Preparation of Slides - Milind Nakul
- Report - Everyone

References

Eric L. Charnov. Optimal foraging, the marginal value theorem. *Theoretical Population Biology*, 9(2):129–136, 1976. ISSN 0040-5809. doi: https://doi.org/10.1016/0040-5809(76)90040-X. URL https://www.sciencedirect.com/science/article/pii/004058097690040X.

Arthur Guez Hado van Hasselt and David Silver. Deep reinforcement learning with double q-learning. *Proceedings of the AAAI Conference on Artificial Intelligence, 30(1)*, 2016.

Nils Kolling and Thomas Akam. (reinforcement?) learning to forage optimally. *Current Opinion in Neurobiology*, 46:162–169, 2017. ISSN 0959-4388. doi: https://doi.org/10.1016/j.conb.2017.08.008. URL https://www.sciencedirect.com/science/article/pii/S0959438817301393. Computational Neuroscience.

Zichuan Lin, Tianqi Zhao, Guangwen Yang, and Lintao Zhang. Episodic memory deep q-networks. *arXiv preprint arXiv:1805.07603*, 2018.

Amir Ramezani and Deokjin Lee. Memory-based reinforcement learning algorithm for autonomous exploration in unknown environment. *International Journal of Advanced Robotic Systems*, 15:172988141877584, 05 2018. doi: 10.1177/1729881418775849.

David Silver Alex Graves Ioannis Antonoglou Daan Wierstra Martin Riedmiller Volodymyr Mnih, Koray Kavukcuoglu. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Matteo Hessel Hado van Hasselt Marc Lanctot Nando de Freitas Ziyu Wang, Tom Schaul. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*, 2016.