Foraging in Replenishing Patches

Abhinav Joshi (20211261)
Archi Gupta (21111014)
Samrudh B Govindaraj (20128409)
Shiven Tripathi (190816)

Problem Statement

Understanding human foraging behaviour in a replenishing patches environment.

<u>Importance</u>

- Learn about sequential decision making processes in the face of uncertainty.
- Understand the role of working memory in sequential decision making.
- Possibly improve existing reinforcement learning algorithms.
- Different from conventional foraging tasks as patches can be revisited.

Sequential Decision making models

RL Agents
- Decay Epsilon Greedy (Bandit Strategies)
- Sarsa
- Q learning
- Eligibility Traces

Pure RL Heuristics
- Distance cost
- Replenishment value function

RL with WM
- Only Forgetting
- With Retention

RL with MVT

Deep RL Agents
- DQN
- A2C
- PPO

Non RL Agents
- MVT

With replenishment

Action selection and estimating rewards

With working memory

Local and global thresholds

Marginal Value Theorem

$$P(exploit)_t = \frac{1}{1 + exp(-[c + \beta(r_{t-1} - T_t)])}$$

The marginal value theorem (MVT) is used to describe the behavior of an optimally foraging individual.

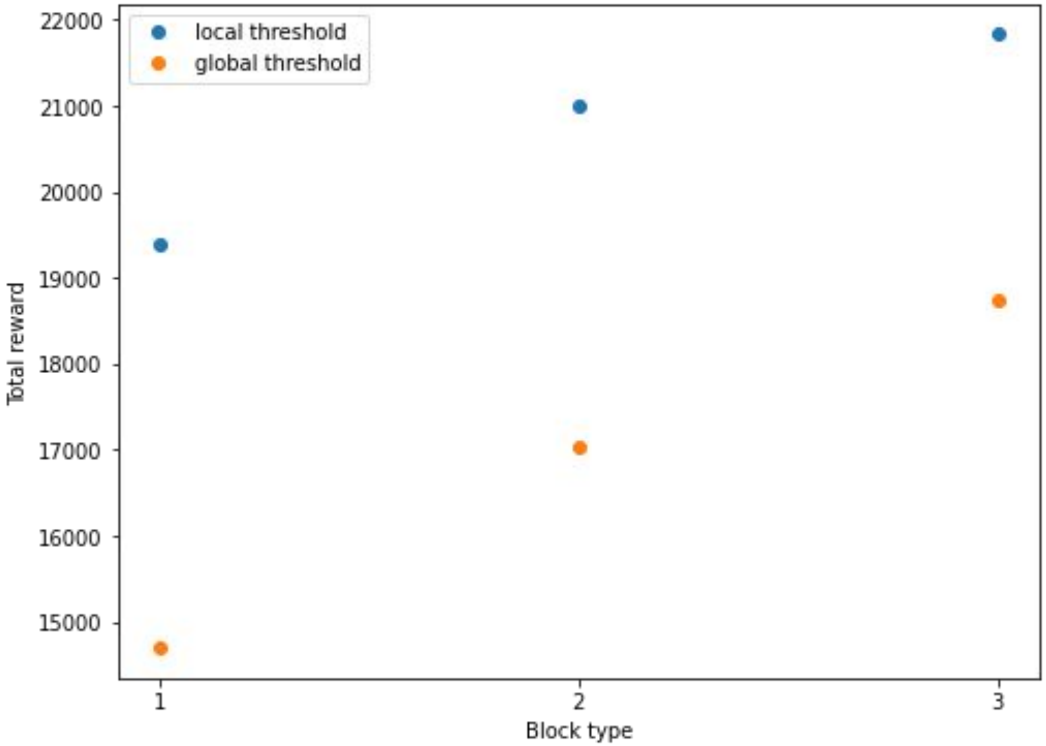
Problems with MVT in the current context:

- Only tells the agent (probabilistically) when and not where to leave.
- Assumes no revisiting, no point of storing previous patches.
- Works for decaying rewards but assumes no replenishment, as patches cannot be revisited.

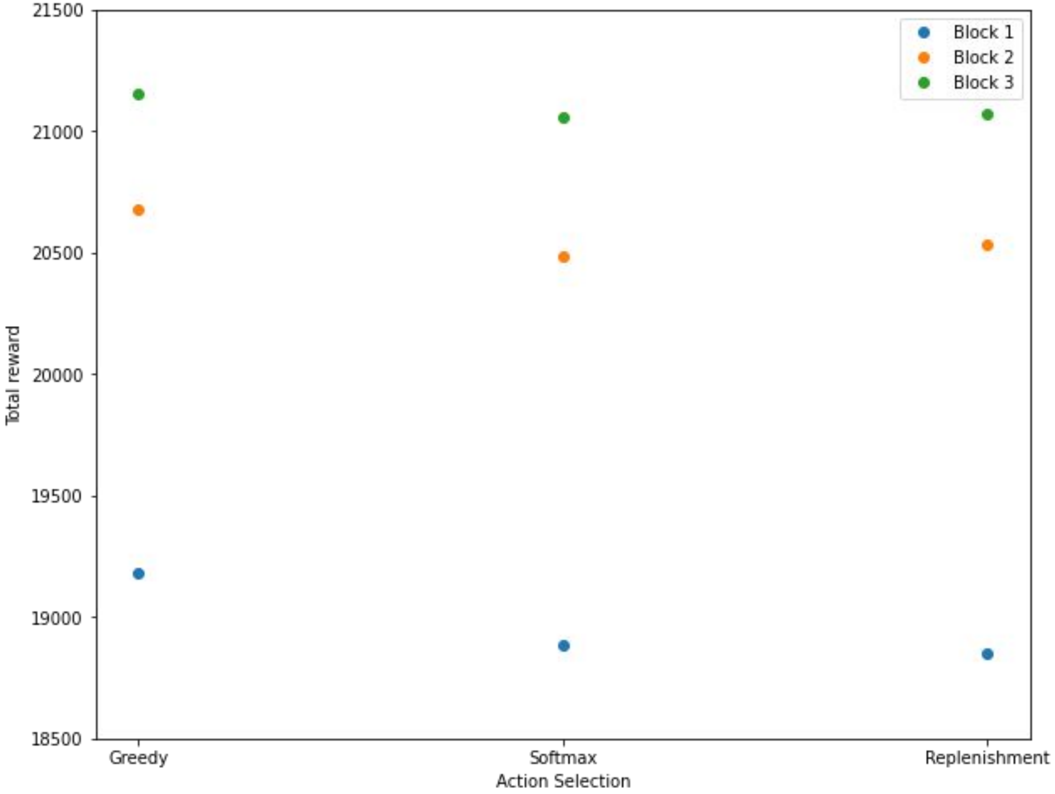
Modified MVT used in the study.

- With local rewards for each patch.
- Replenishment rate.
- Working memory.
- Estimated rewards.
- Action selection strategies.

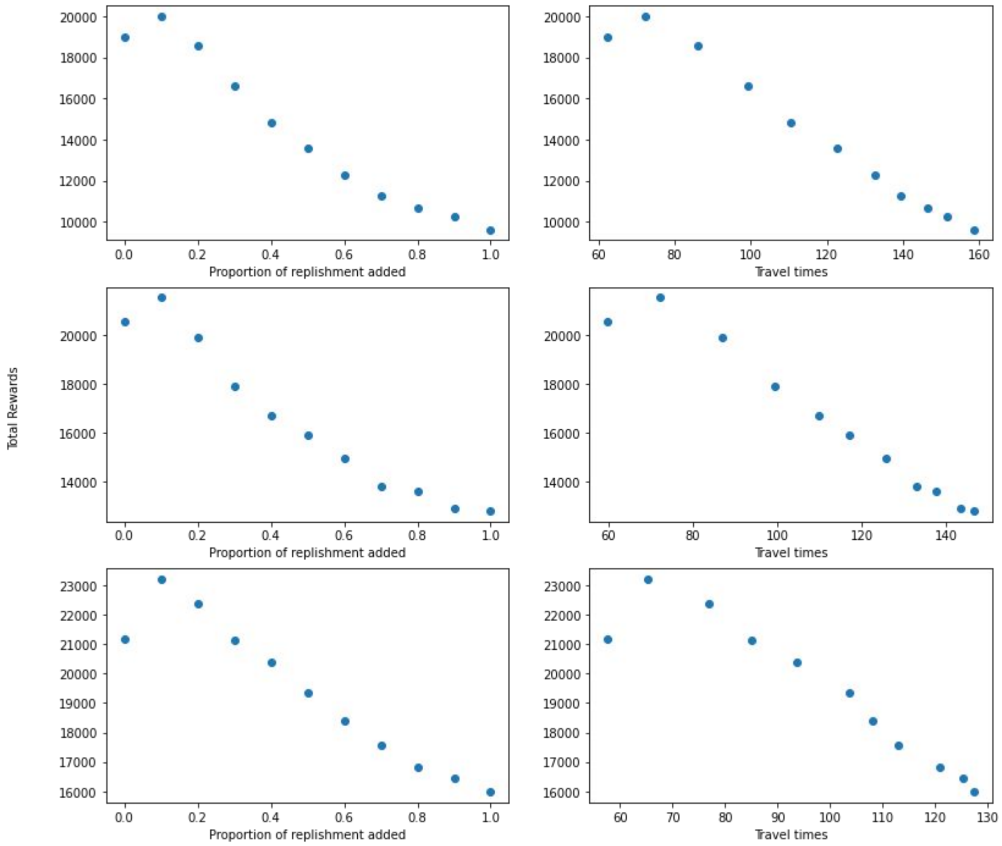
Local vs Global thresholds



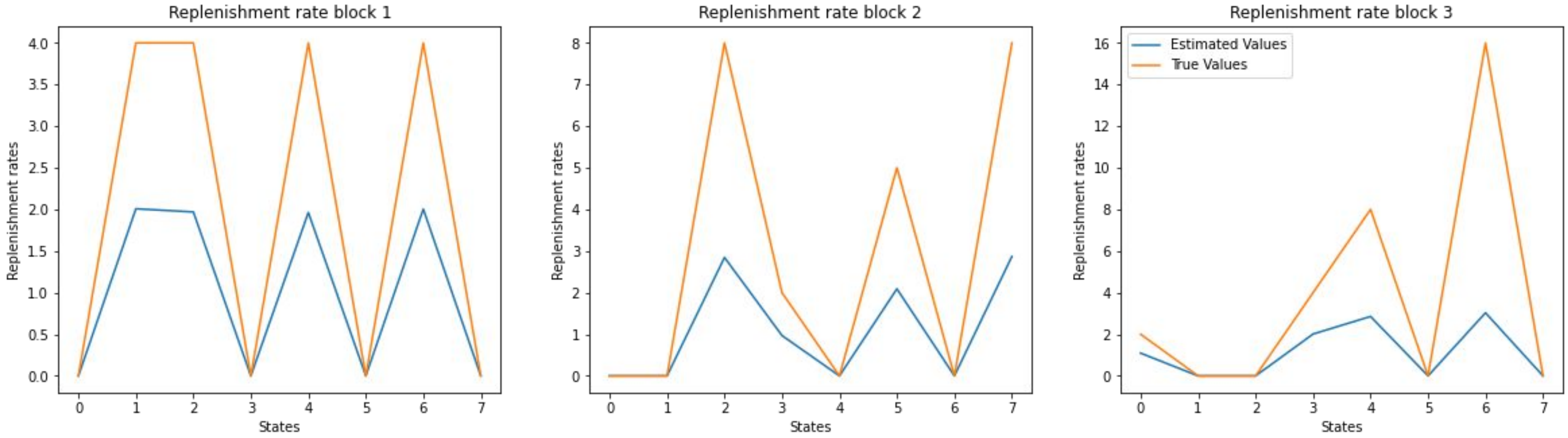
Action selection strategy



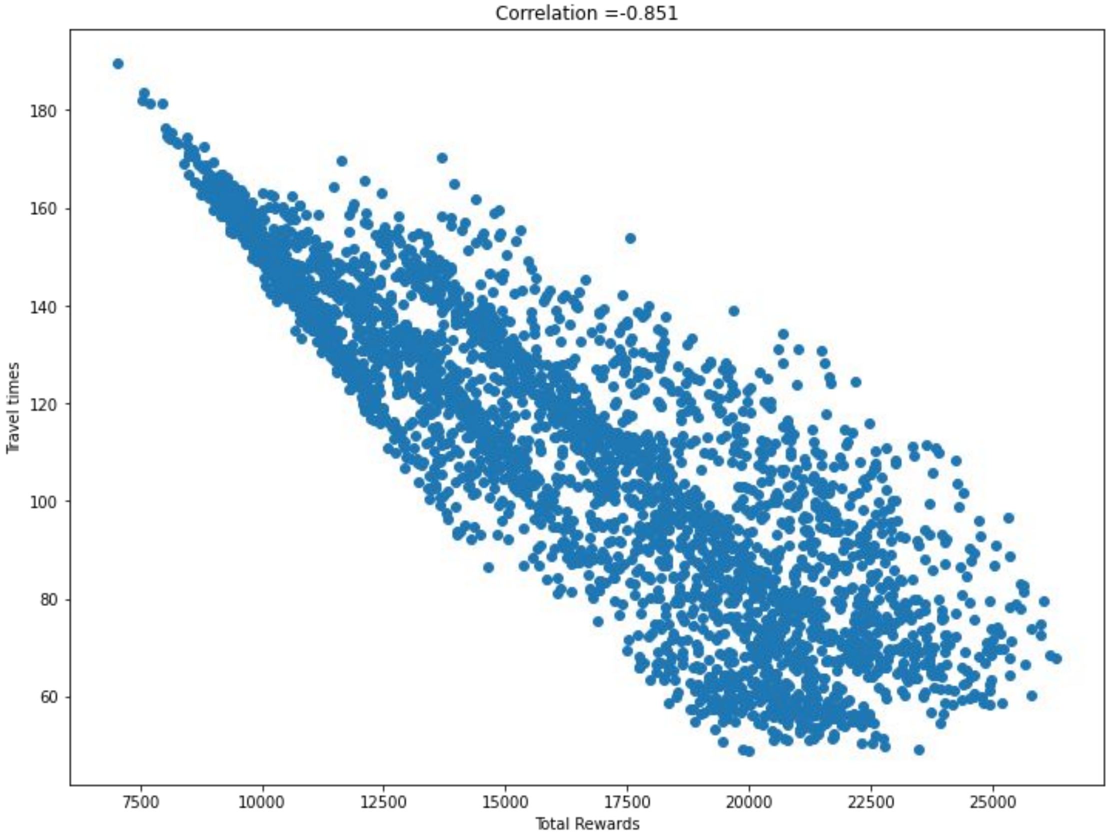
Adding replenishment rate



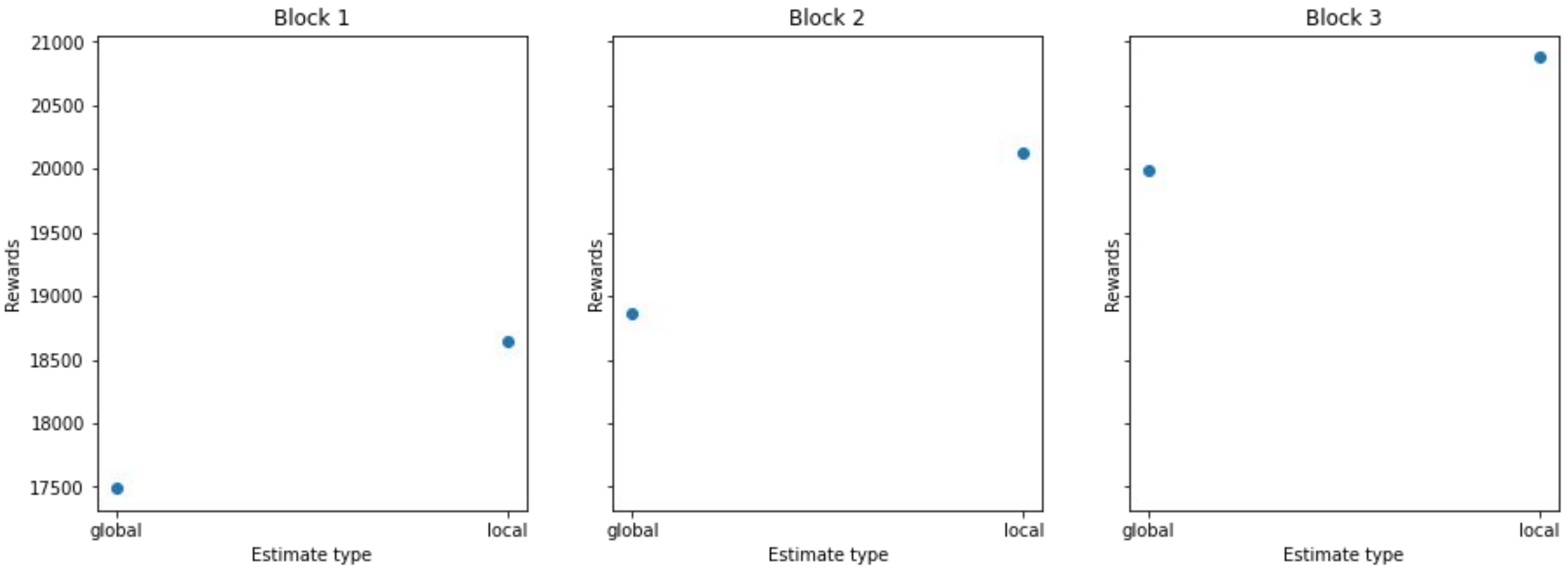
Replenishment rates



Travel time and total reward received

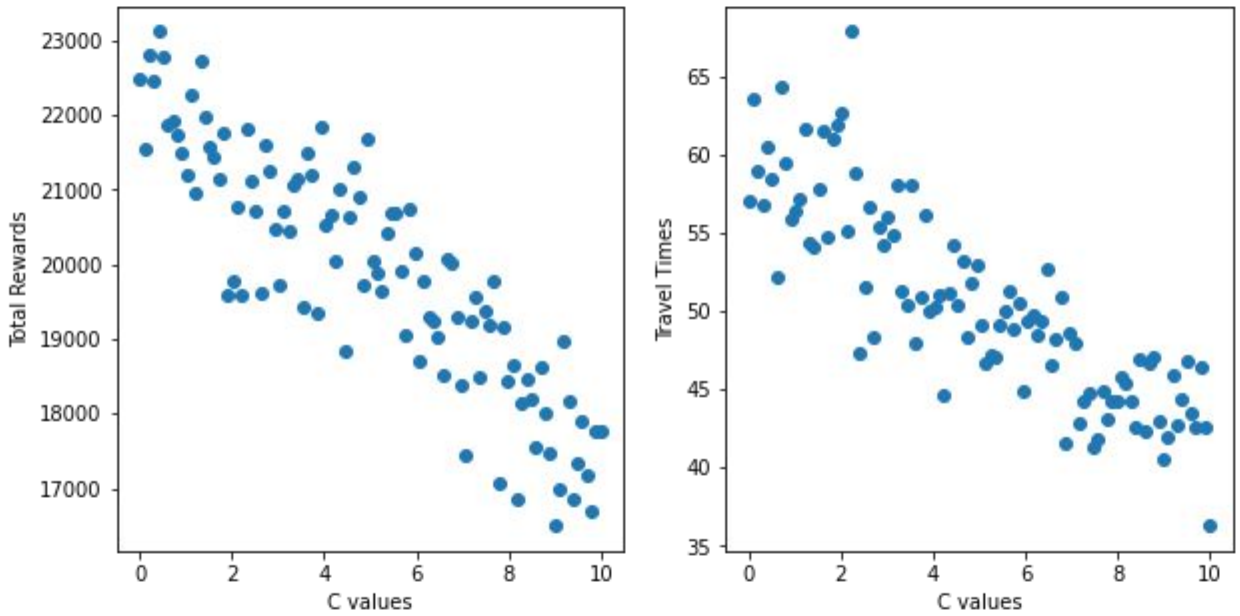


Learning estimated rewards from the environment

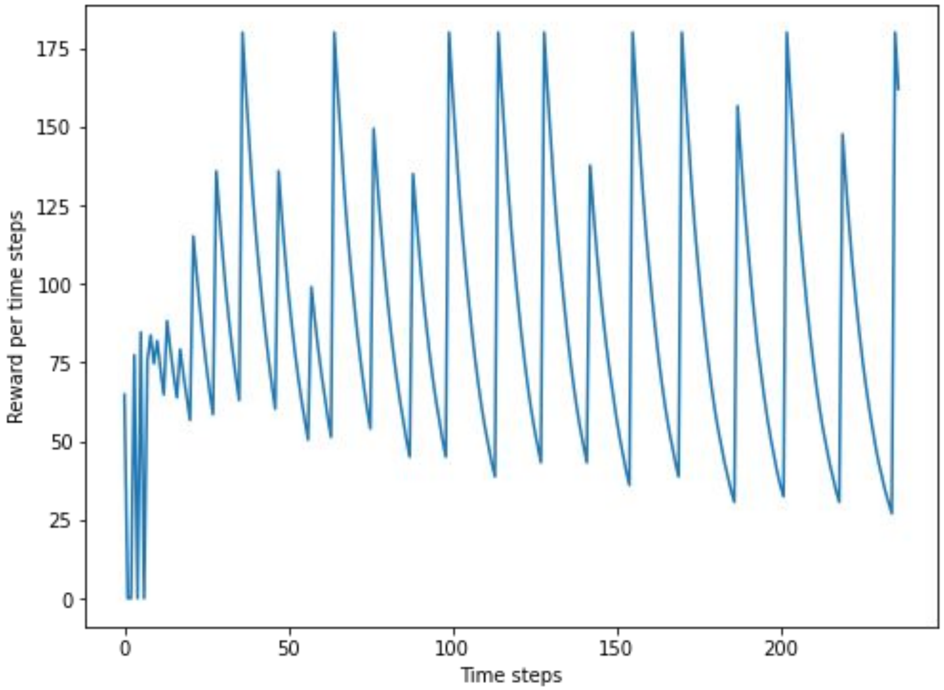


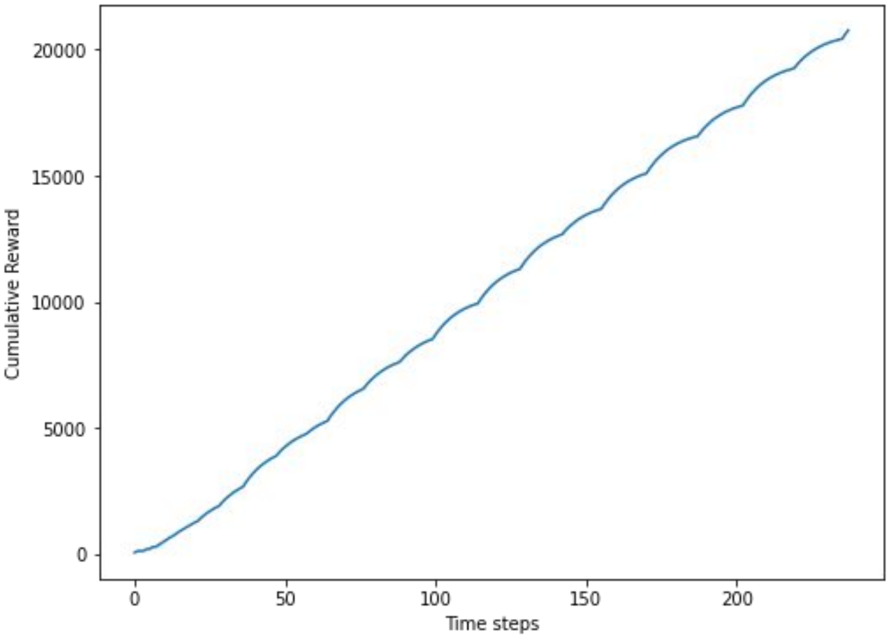
C-values

$$P(exploit)_t = \frac{1}{1 + exp(-[c + \beta(r_{t-1} - T_t)])}$$

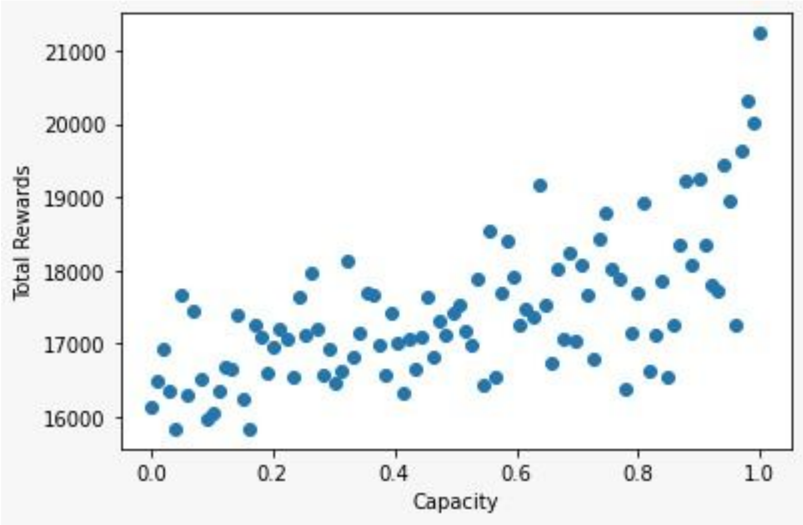


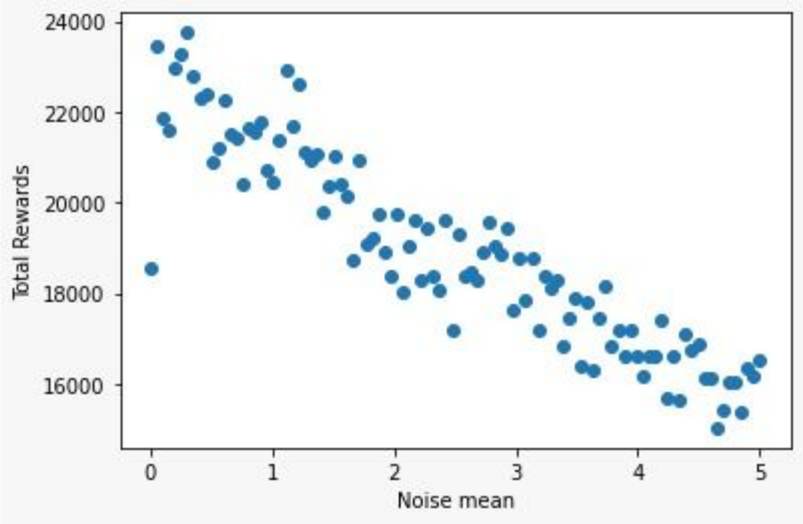
Reward across time



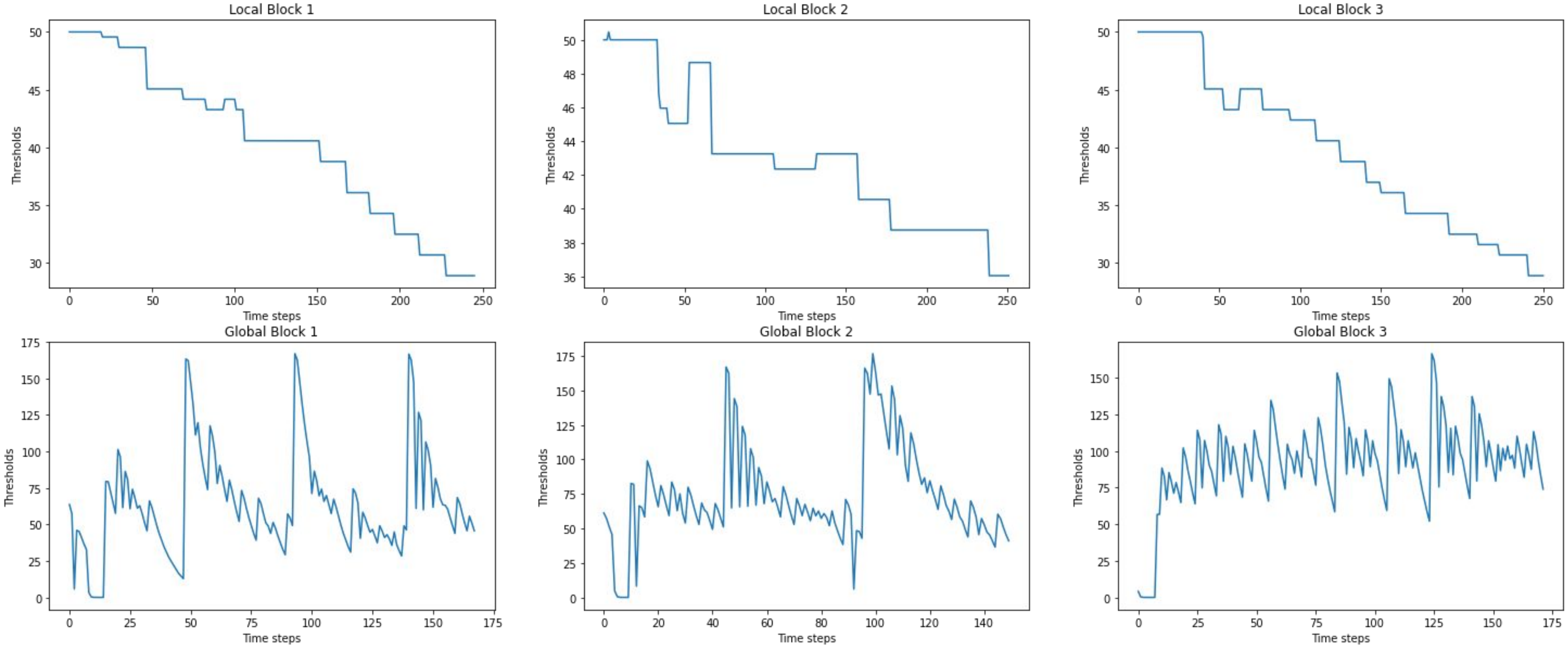


Working memory parameters

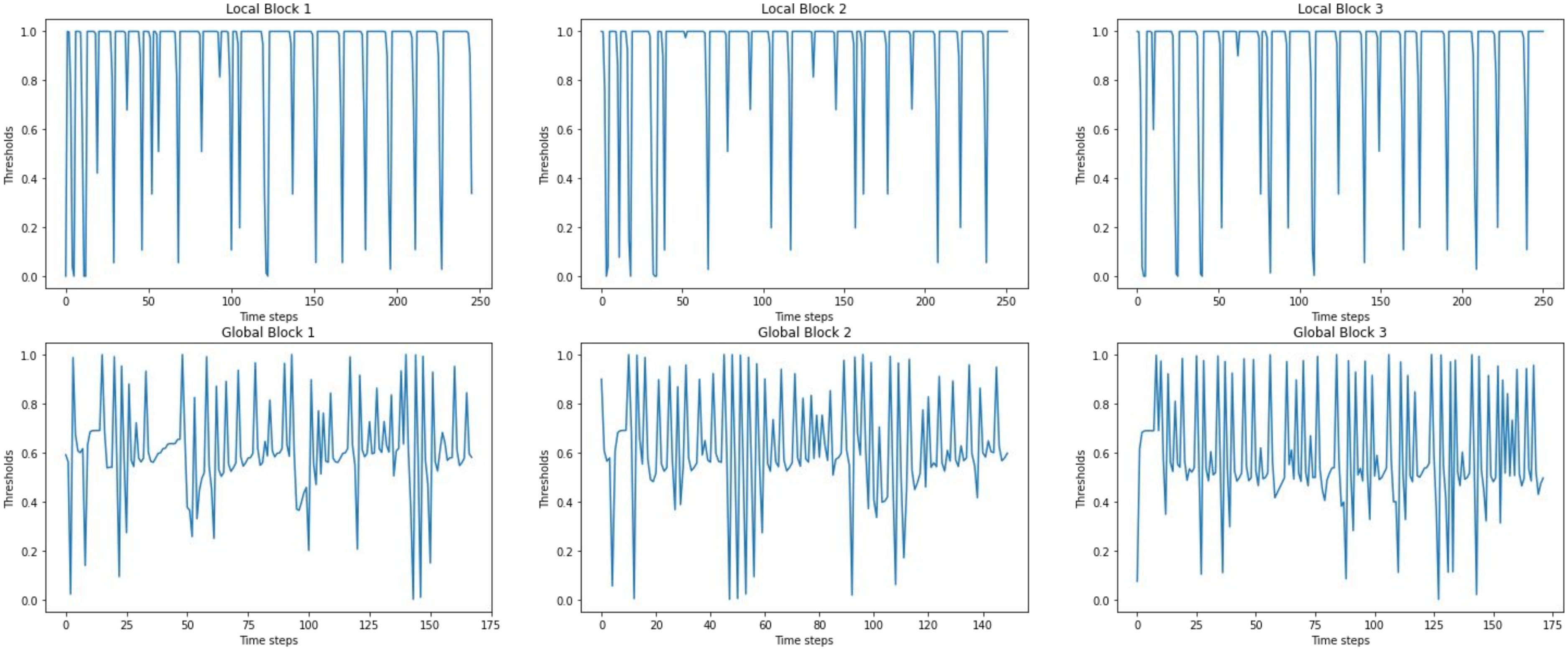


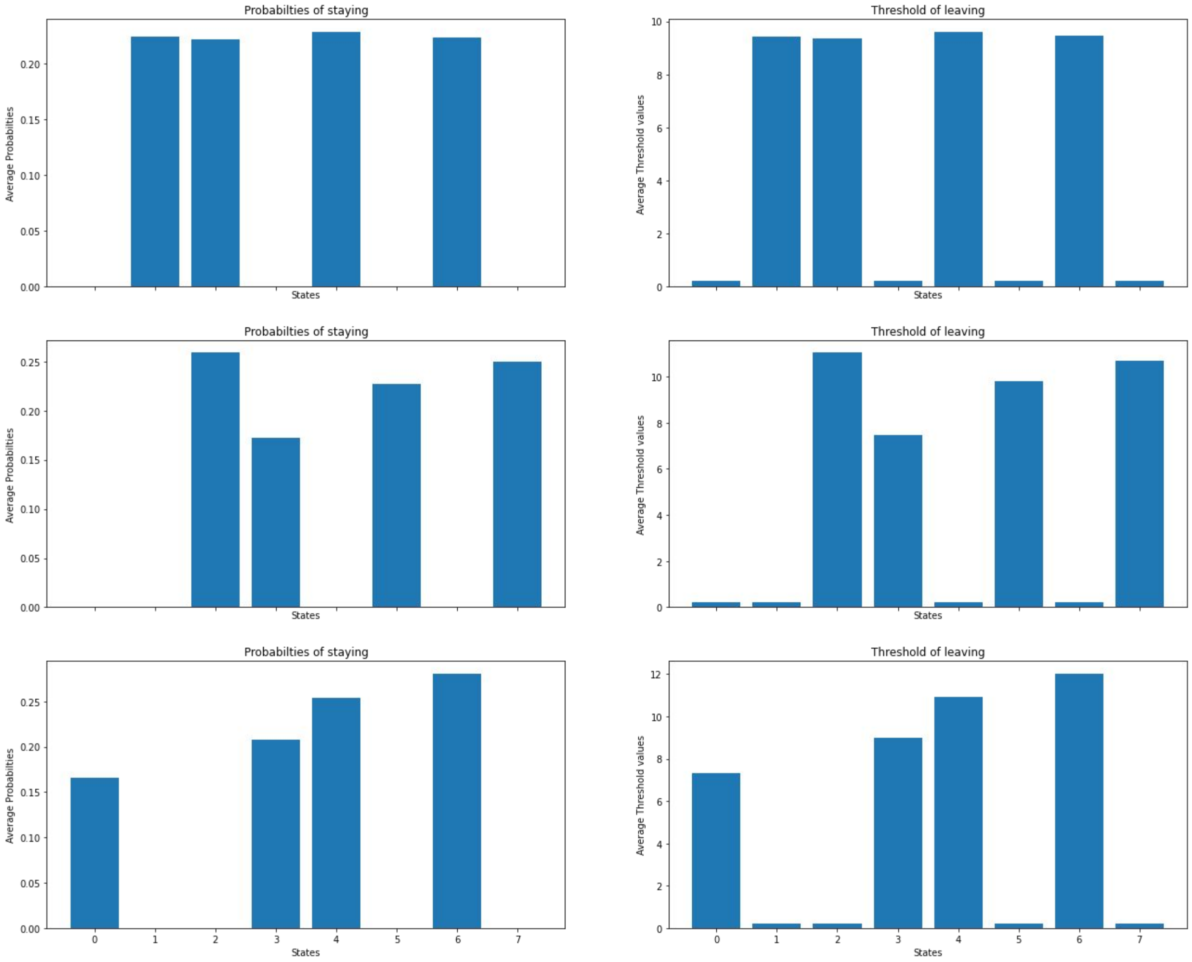


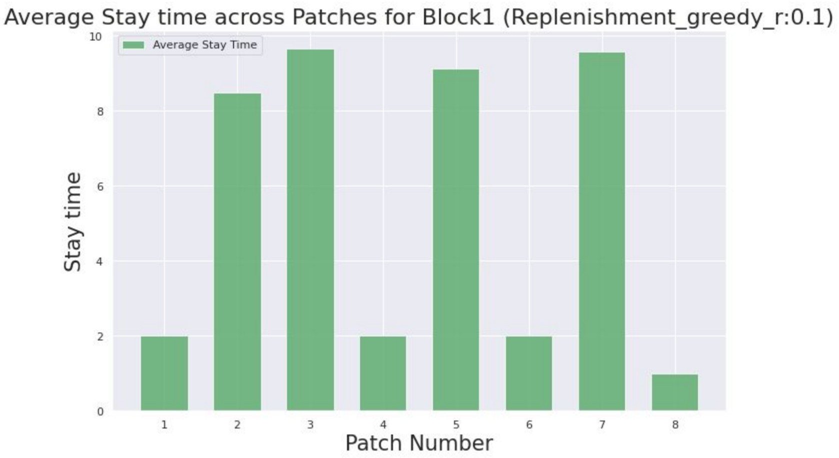
Thresholds of staying over time

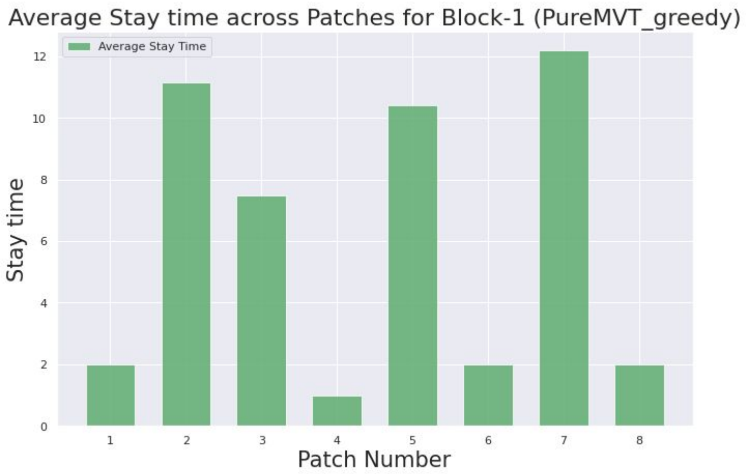


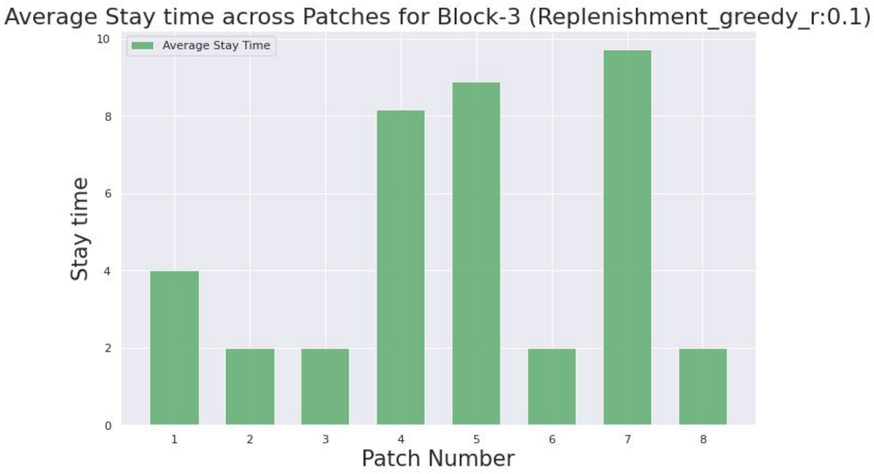
Probabilities of staying over time

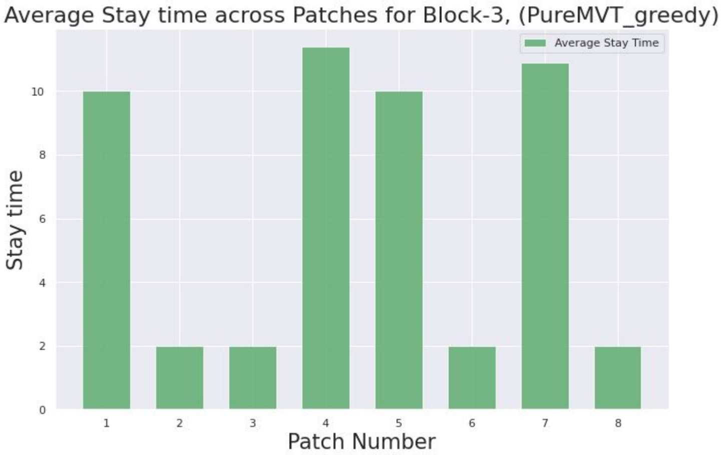



Probabilties of staying
Average Probabilties
0.00
0.05
0.10
0.15
0.20
States
Threshold of leaving
Average Threshold values
0
2
4
6
8
10
States
Probabilties of staying
Average Probabilties
0.00
0.05
0.10
0.15
0.20
0.25
States
Threshold of leaving
Average Threshold values
0
2
4
6
8
10
States
Probabilties of staying
Average Probabilties
0.00
0.05
0.10
0.15
0.20
0.25
0
1
2
3
4
5
6
7
States
Threshold of leaving
Average Threshold values
0
2
4
6
8
10
12
0
1
2
3
4
5
6
7
States

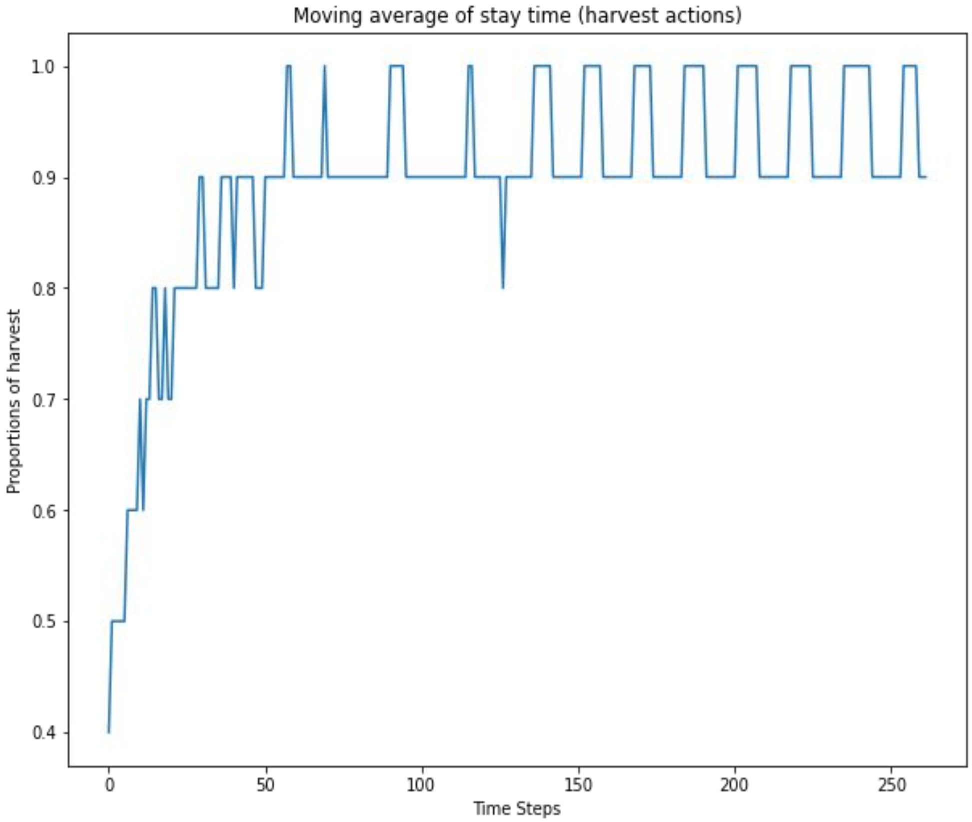

Average Stay time across Patches for Block1 (Replenishment_greedy_r:0.1)
Average Stay Time
Stay time
Patch Number
0
2
4
6
8
10
1
2
3
4
5
6
7
8


Average Stay time across Patches for Block-1 (PureMVT_greedy)
Average Stay Time
Stay time
Patch Number
0
2
4
6
8
10
12
1
2
3
4
5
6
7
8


Average Stay time across Patches for Block-3 (Replenishment_greedy_r:0.1)
Average Stay Time
Stay time
Patch Number
0
2
4
6
8
10
1
2
3
4
5
6
7
8


Average Stay time across Patches for Block-3, (PureMVT_greedy)
Average Stay Time
Stay time
Patch Number
0
2
4
6
8
10
1
2
3
4
5
6
7
8

Number of harvest actions over time



RL Methods

Multi Armed Bandit

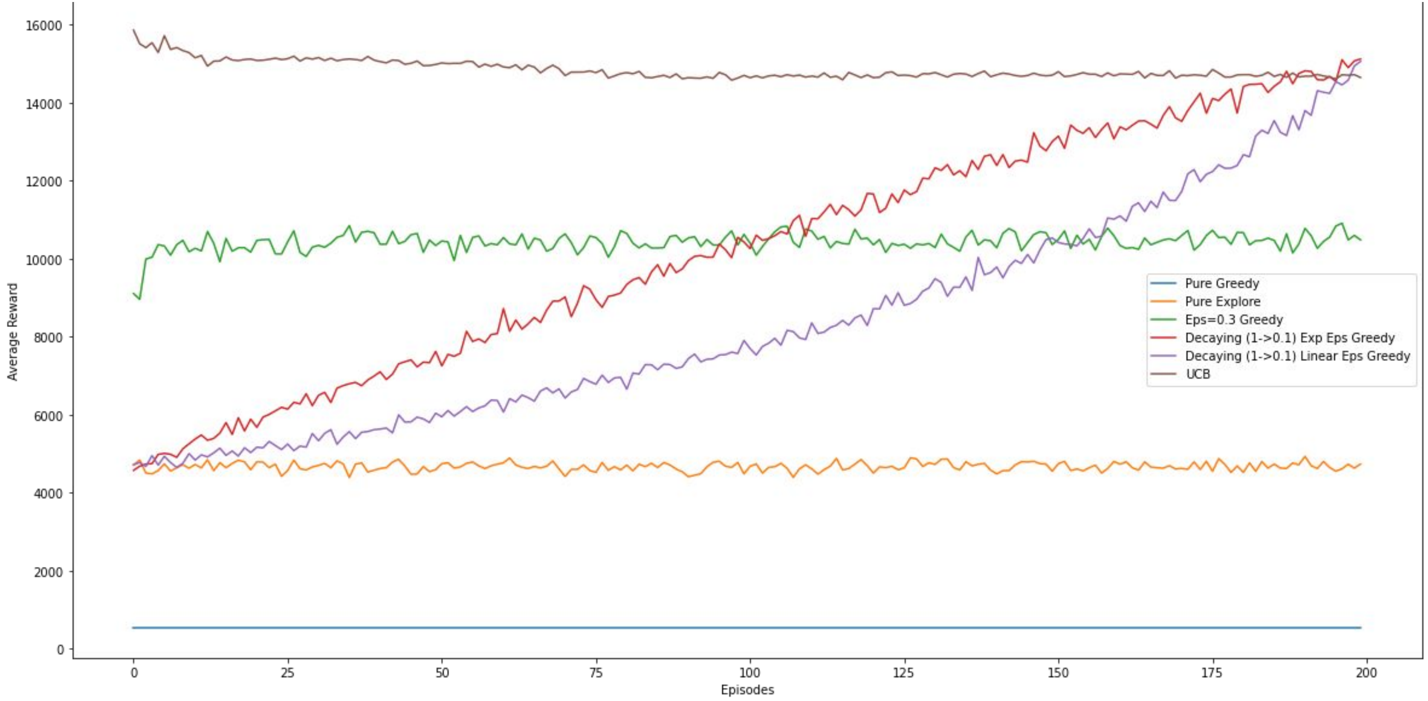
Assumptions:

- Problem can be approximated to a multi armed bandits situation
- Choice with agent to pick a patch to go to, and then it commits to harvesting
- Each episode is a sequence of bandit decisions, until time runs out

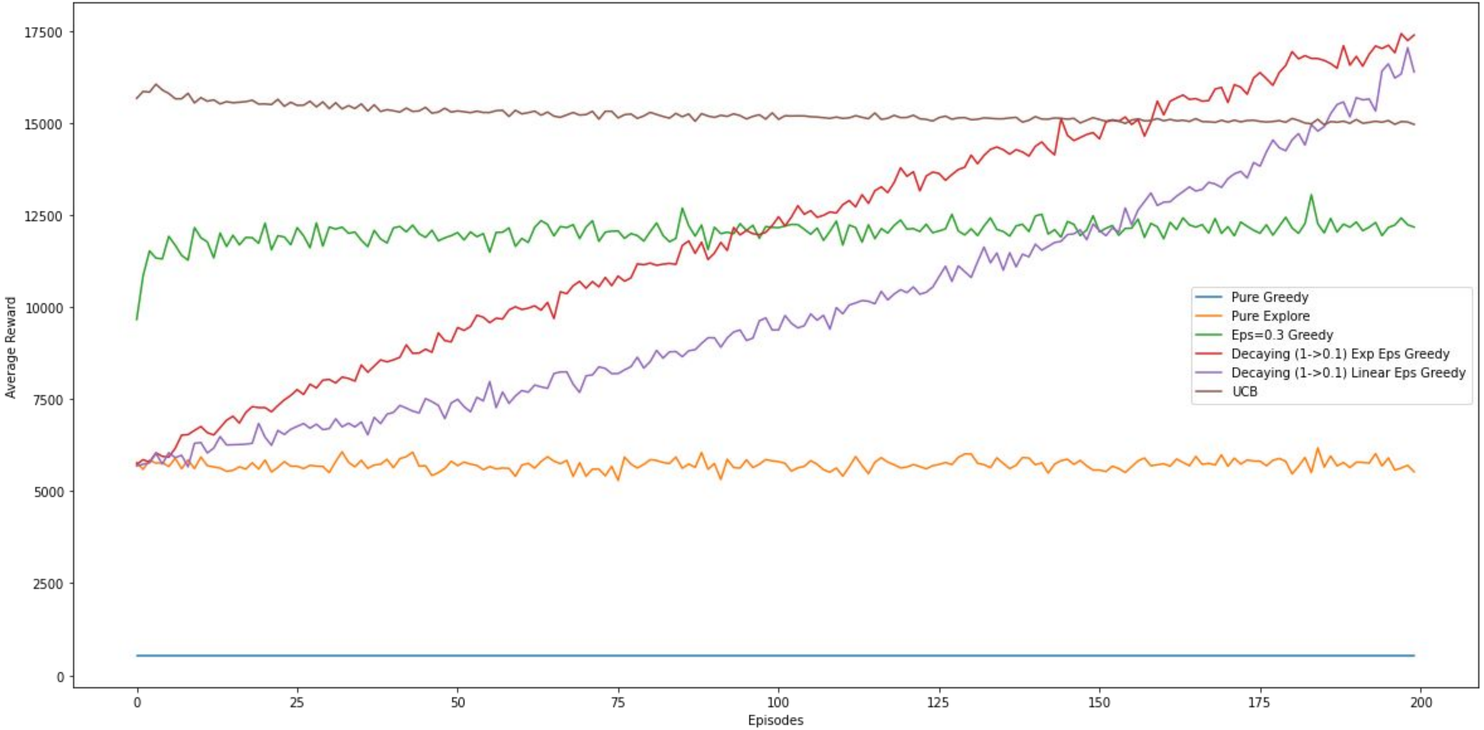
Under these simplifying constraints, following strategies were tested:

- Pure Greedy
- Pure Exploratory
- Fixed Epsilon Greedy
- Decaying Epsilon Greedy
- Uncertainty Confidence Bound (UCB)

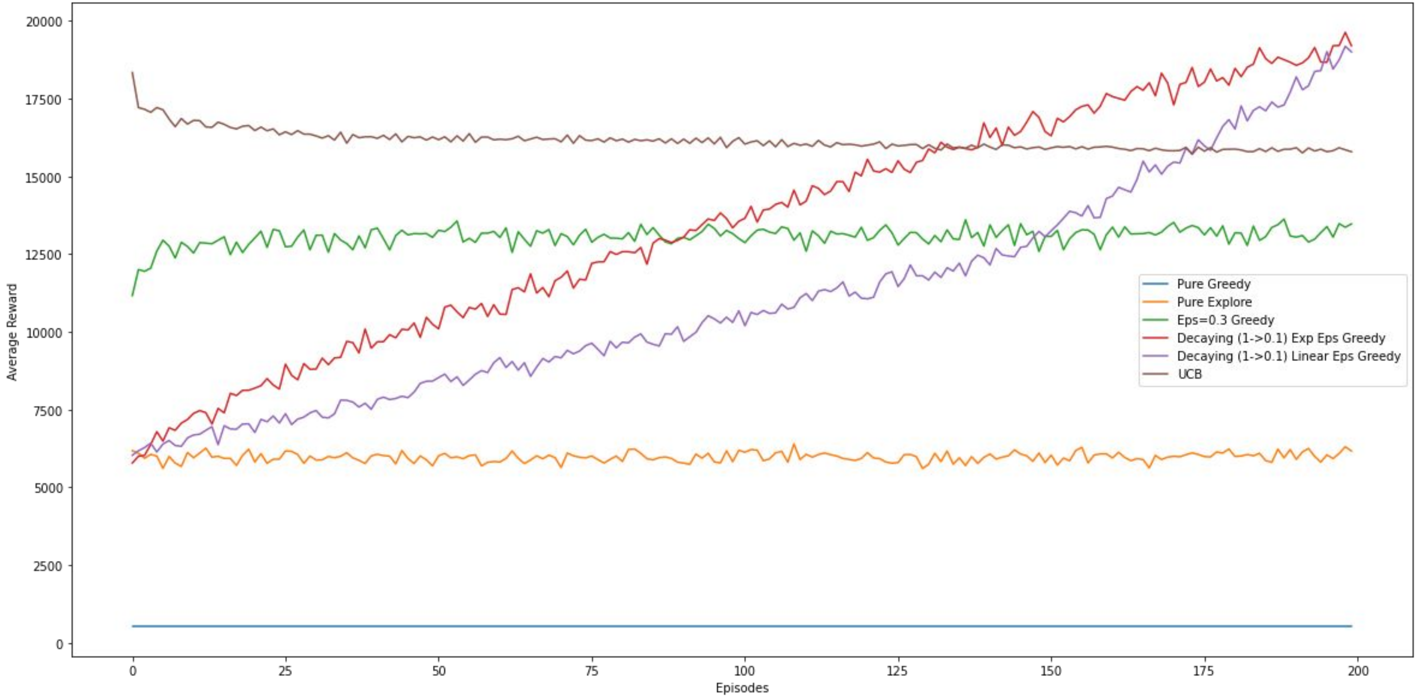
Average Rewards for Agents in MAB across Episodes Block 1 (replenish rate = [0, 4, 4, 0, 4, 0, 4, 0]) avg. over 25 same training runs



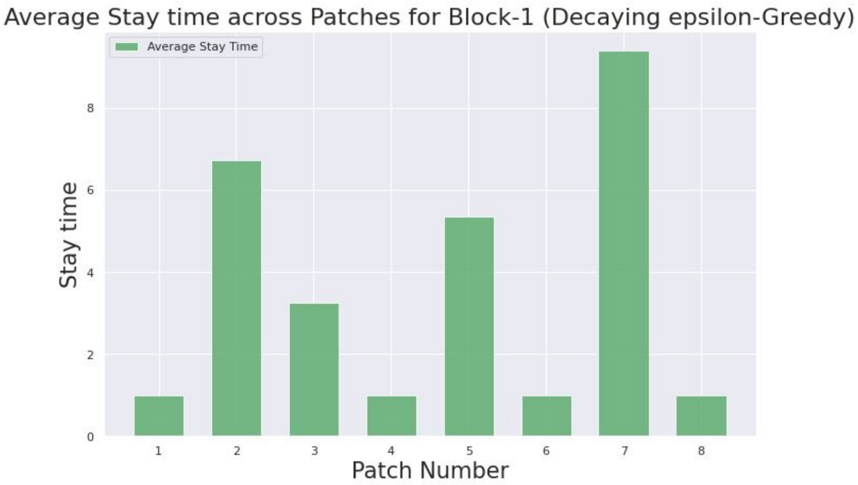
Block 2 (replenish rate = [0, 0, 8, 2, 0, 5, 0, 8]) avg. over 25 same training runs

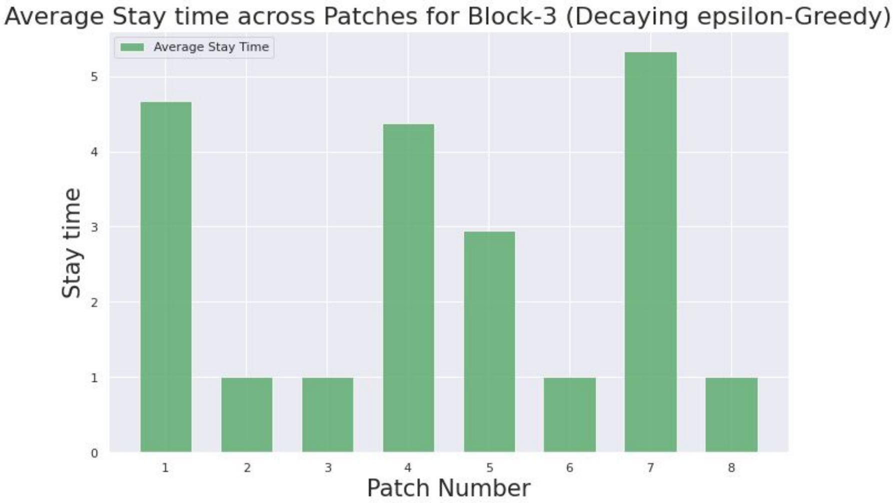


Block 3 (replenish rate = [2, 0, 0, 4, 8, 0, 16, 0]) avg. over 25 same training runs



Average Stay Time for Agent following Decaying Epsilon Greedy Strategy

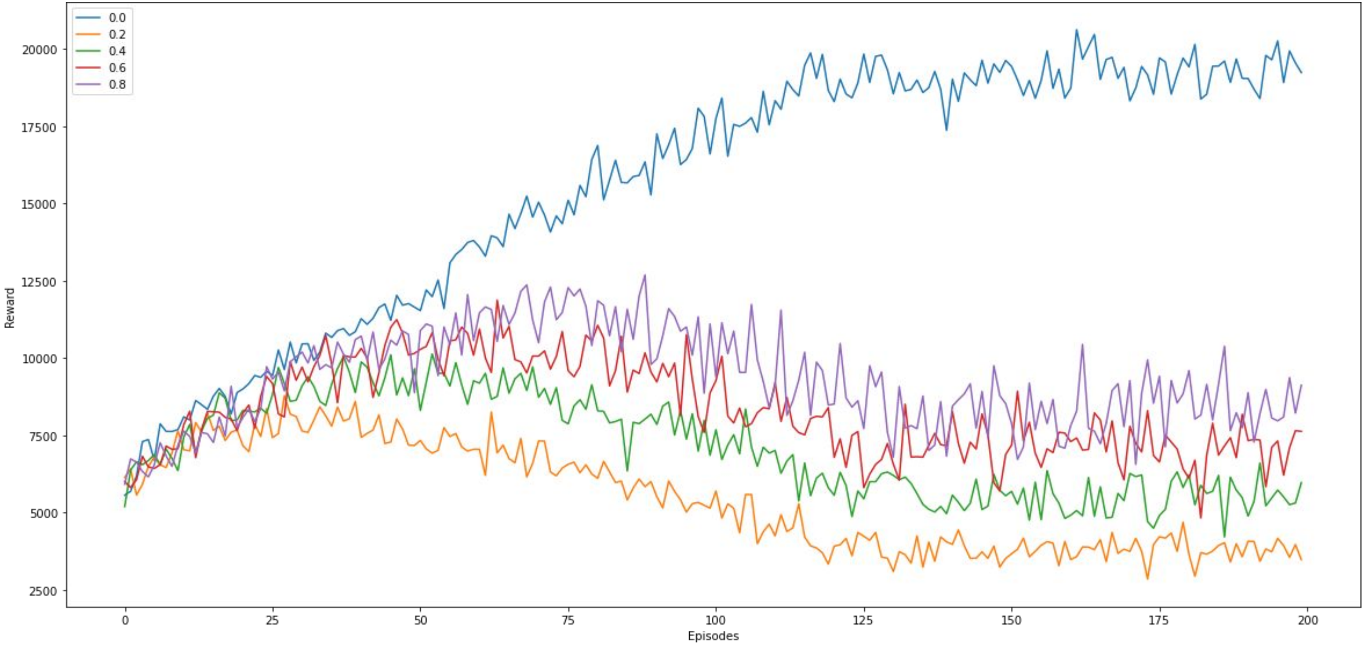




Forgetfulness Model in Decaying Epsilon Greedy Strategy Block 3

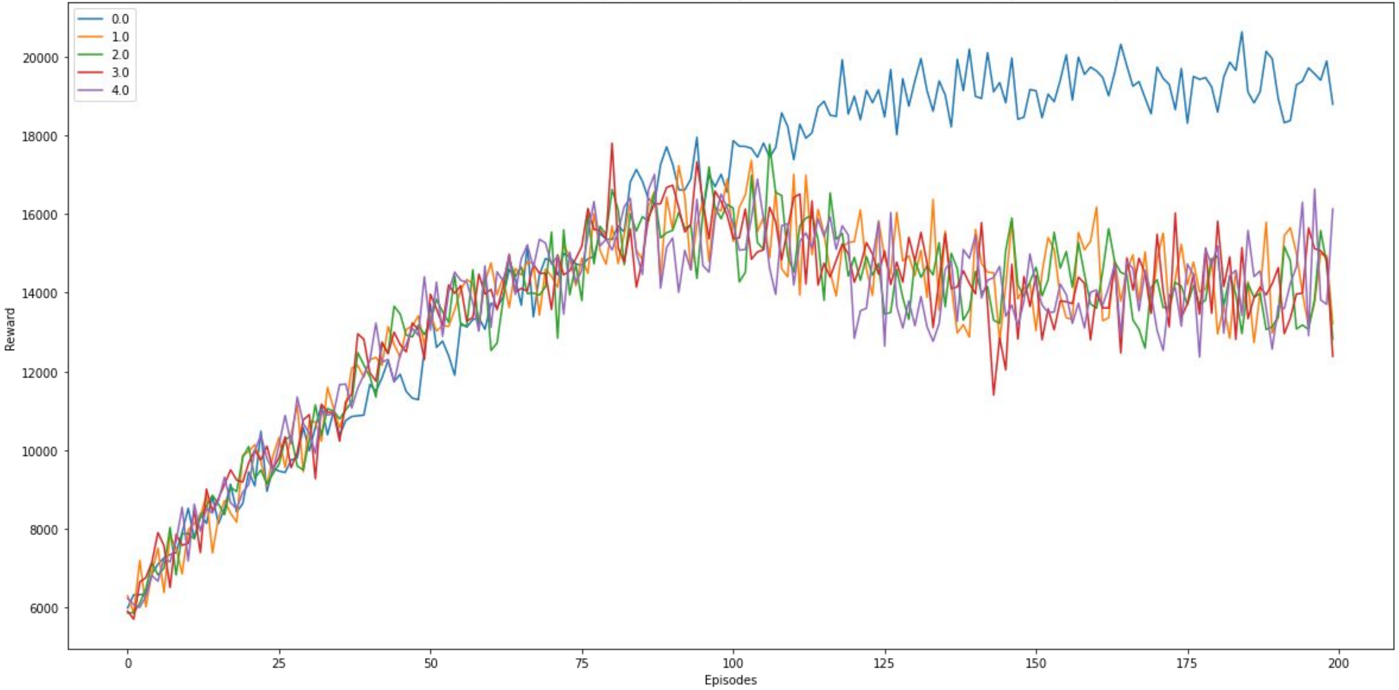
Q *= (1 - forgetfulness)

Q[s] /= (1 - forgetfulness)



Distance Cost in Decaying Epsilon Greedy Strategy Block 3

```
a = np.argmax(Q - distanceWeight * Distances[s])
```



MDP Methods

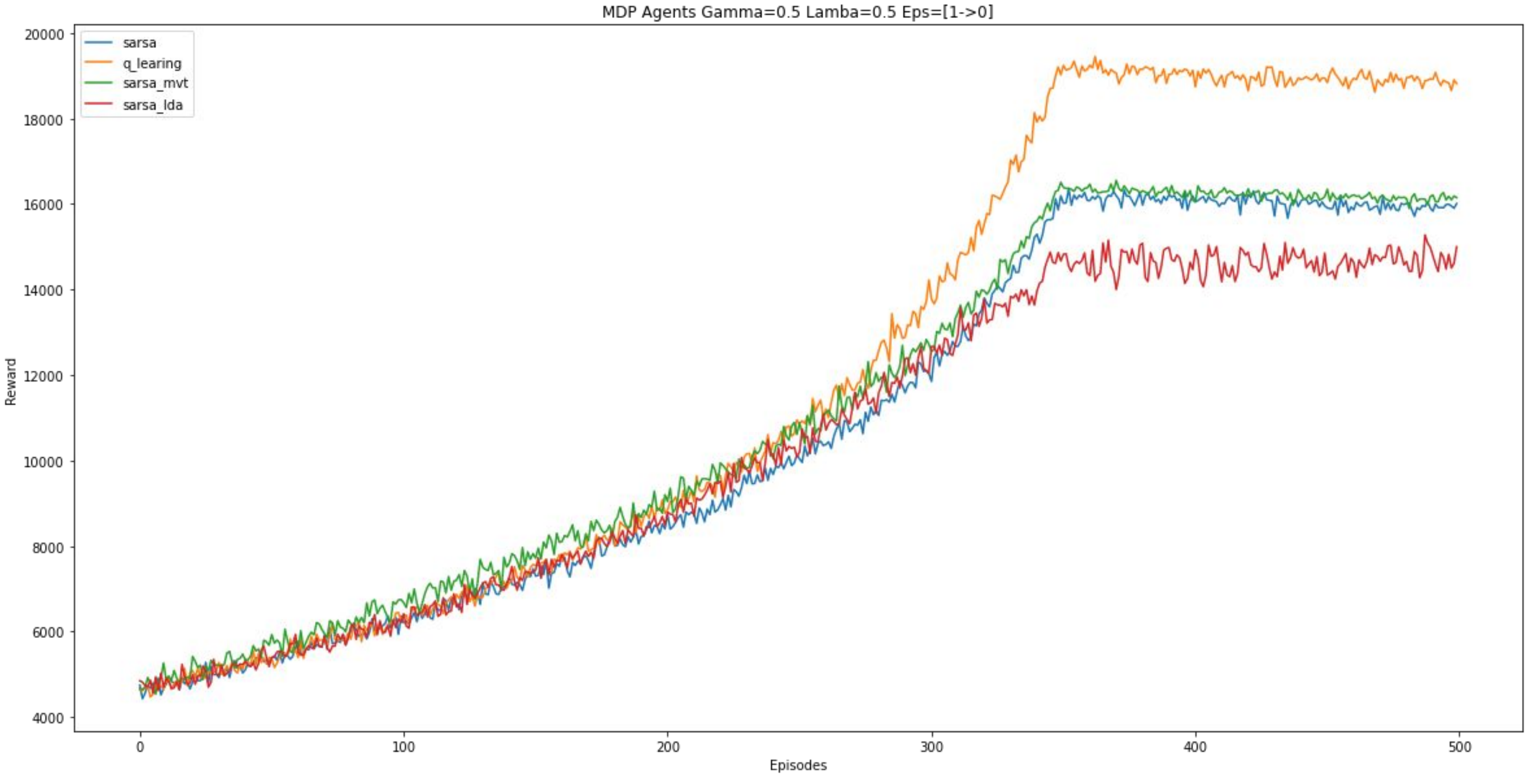
Assumptions:

- Problem can be approximated to a markov decision process
- Choice with agent to pick a patch to go to, and then it commits to harvesting

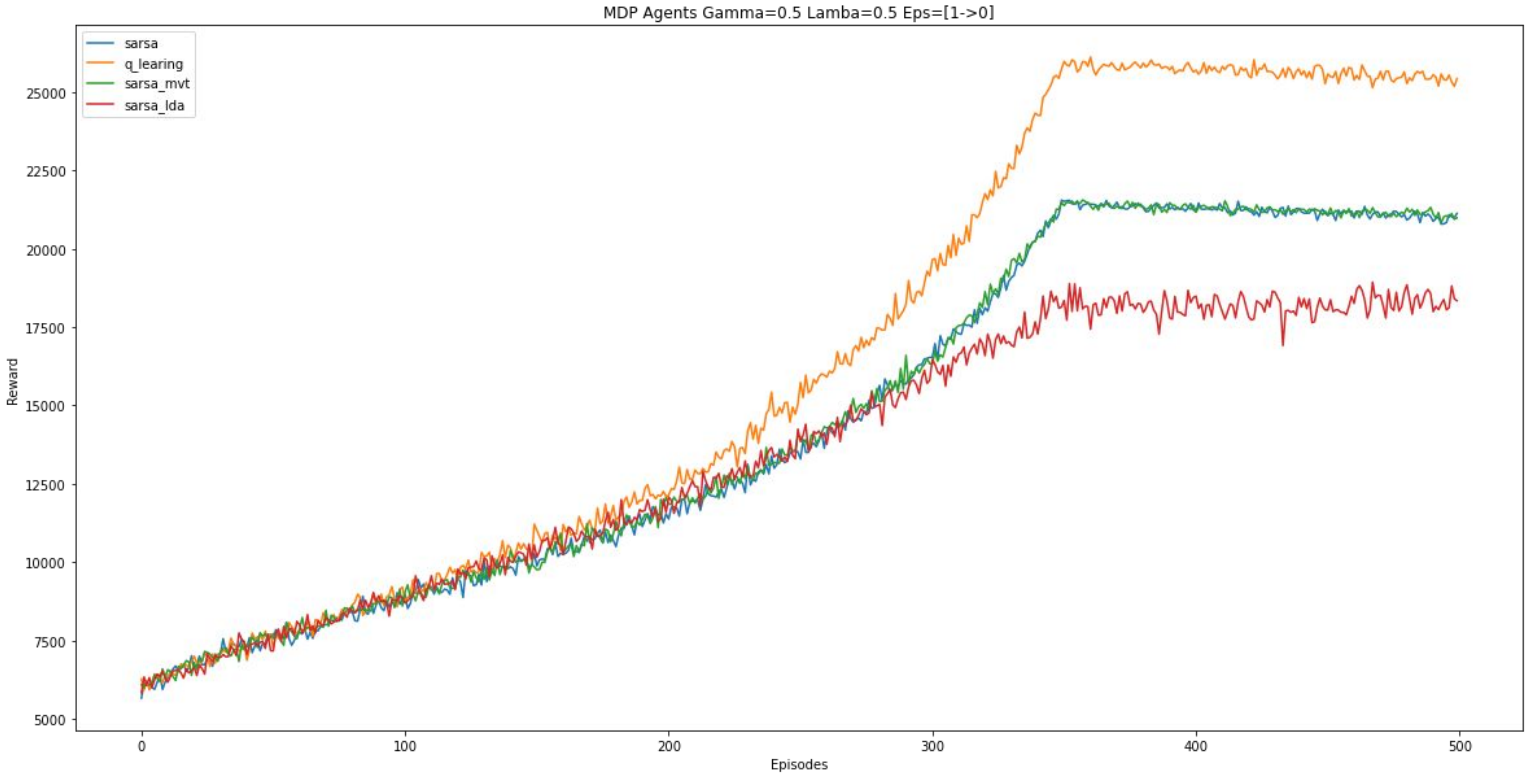
Under these simplifying constraints, following algorithm were tested:

- SARSA
- Q Learning
- SARSA - Lambda
- SARSA - WM
- SARSA Forgetful

MDP Methods - Sarsa, Q Learning, Sarsa with MVT Block-1



MDP Methods - Sarsa, Q Learning, Sarsa with MVT Block-3



Human Forgetful Behaviour Modelling

Aim - Dynamic integration of RL and WM processes observed in human behaviour to capture *behavioural variance*

Forgetfulness - After each value update step, we decay the values towards their initial values

$$\boldsymbol{Q(s,a) \leftarrow Q(s,a) + \varepsilon \times (Q_0 - Q(s,a))}$$

Forget Decay - As we play repeatedly, we become better and gradually we forget less

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3390186/

Human Forgetful Behaviour Modelling

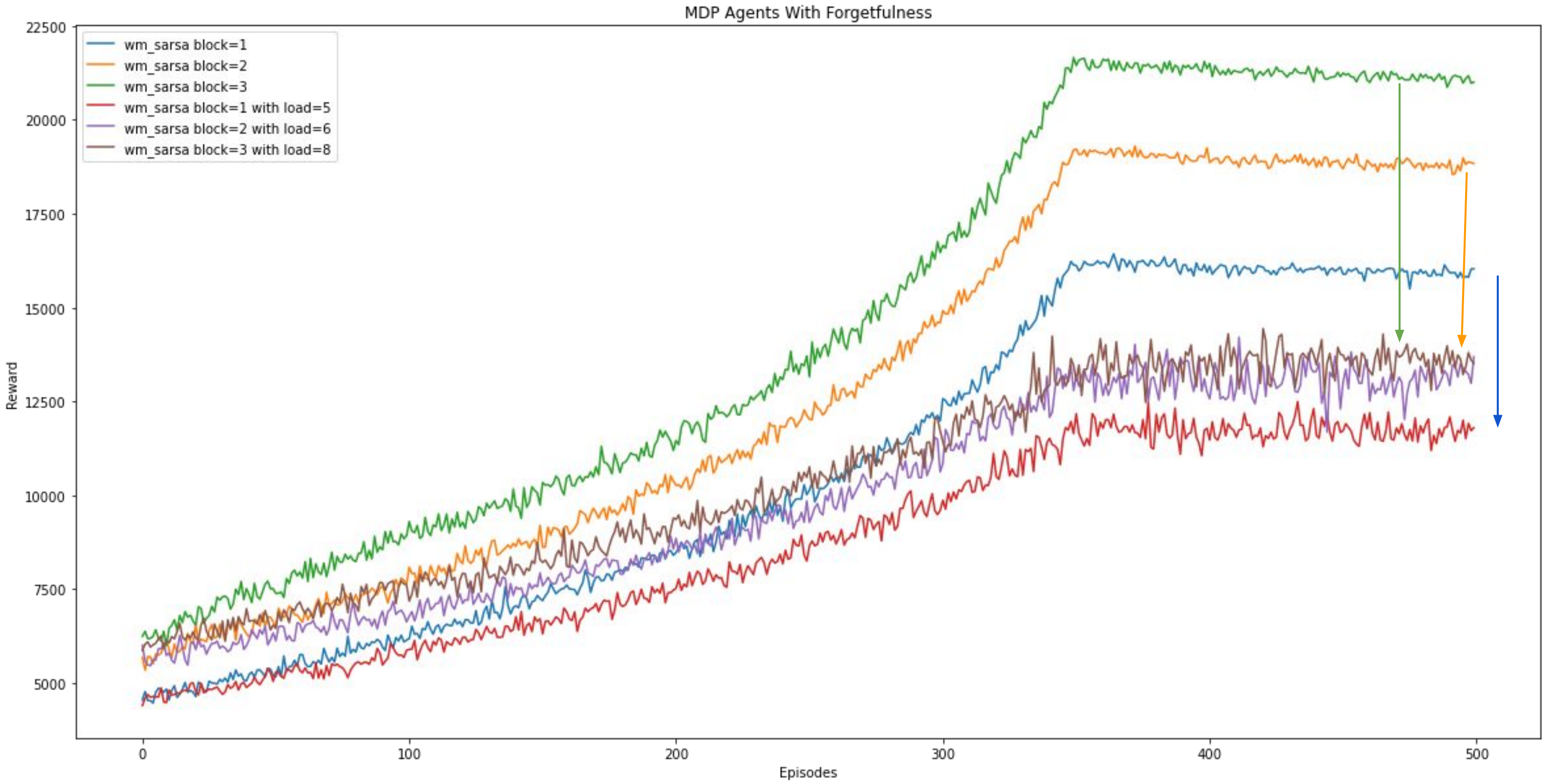
Working Memory Model - Using two value functions $\boldsymbol{Q_{RL}}$ (pure RL) and $\boldsymbol{Q_{WM}}$ (with forgetting) and assigning a weighted probability for action selection.

Memory Capacity *c* - Determines the probability that action selection is governed by the RL or WM component (generally an int 7 +- 2)

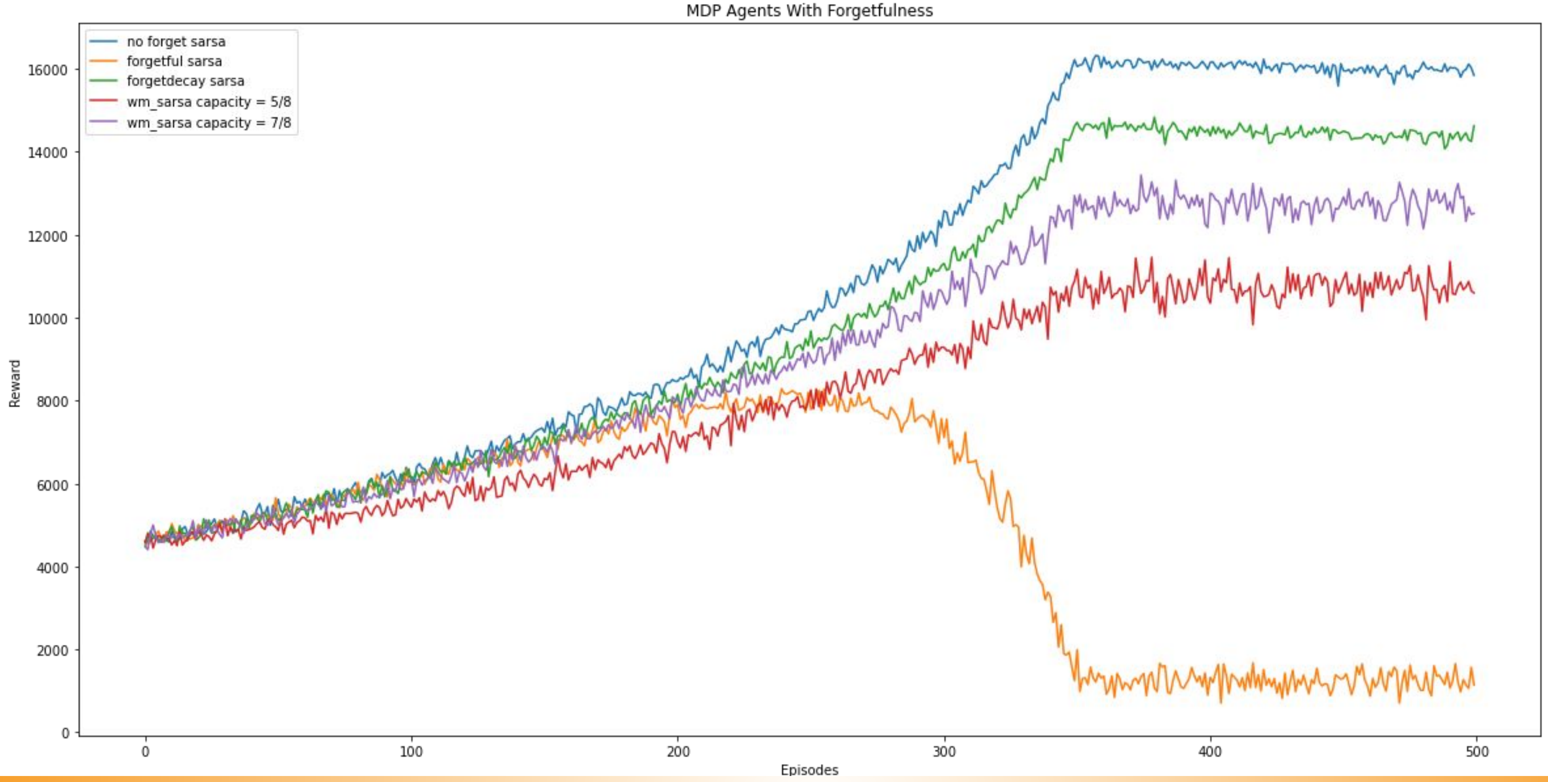
$$
\begin{aligned}
&Q_{RL} \text{ if } \frac{c}{n_s} > 0.5 \\
&Q_{WM} \text{ otherwise}
\end{aligned}
$$

n_s = Task specific memory load

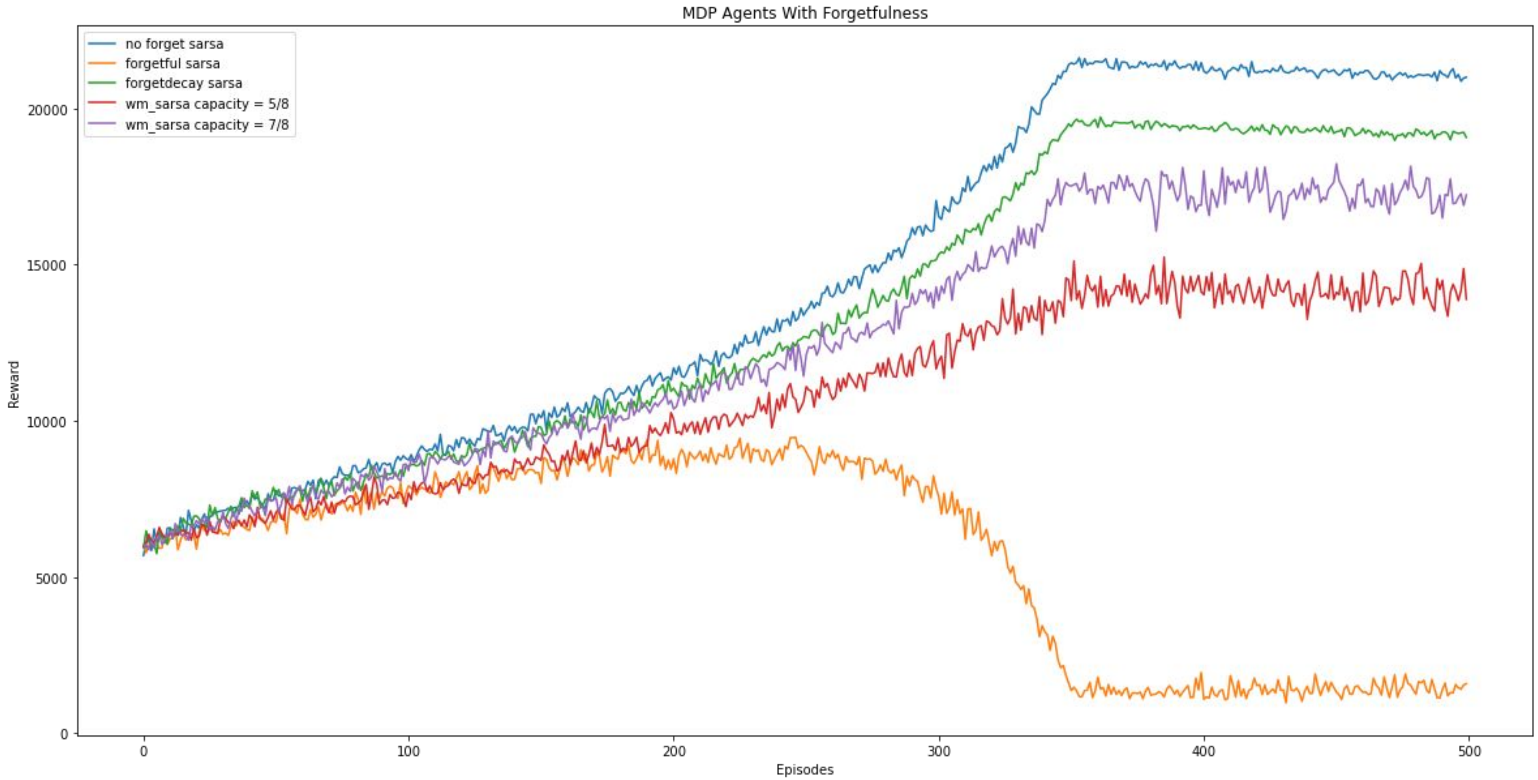
Working Memory Model (capacity = 4) for all blocks

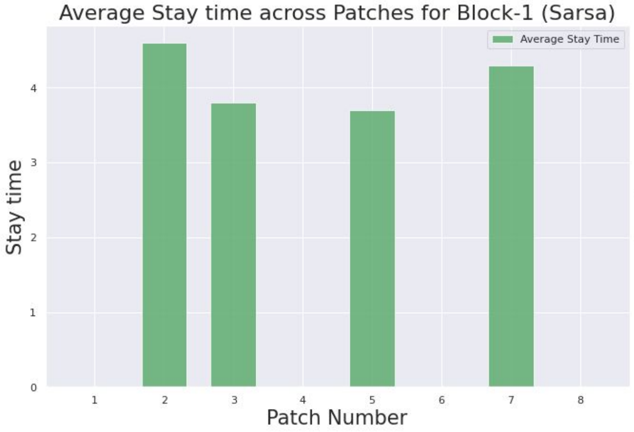


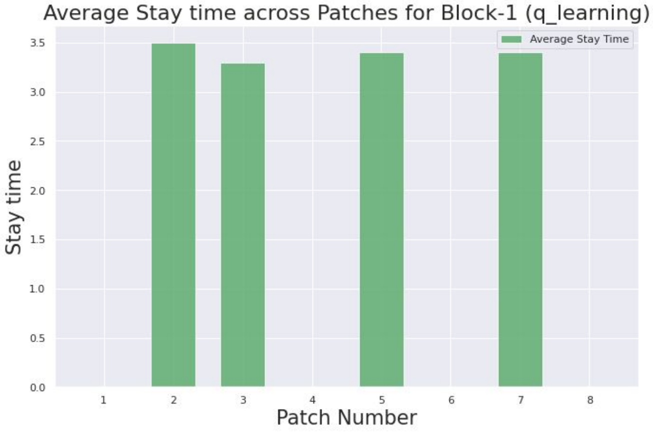
Including Forgetful Behaviour Block-1

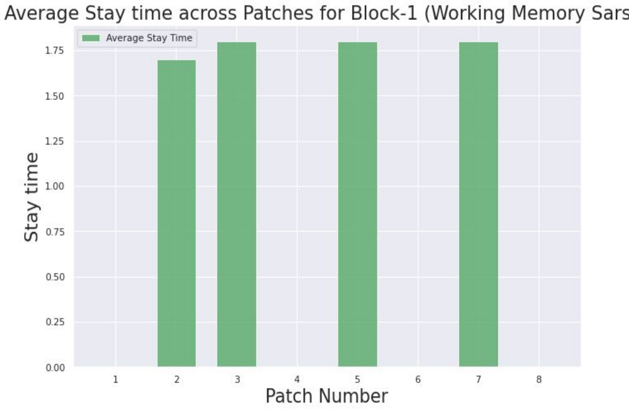


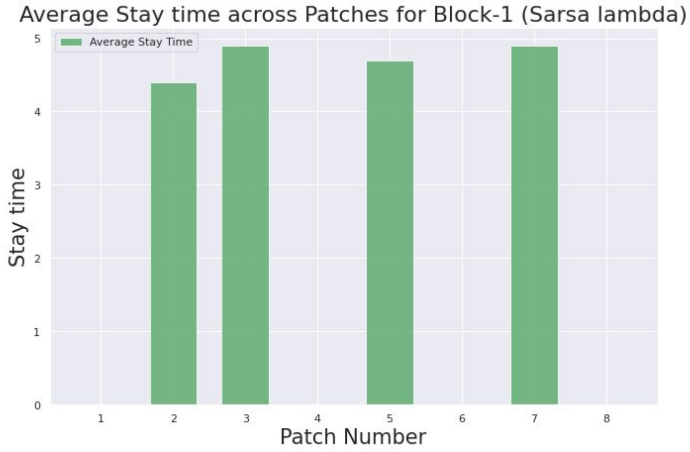
Including Forgetful Behaviour Block-3

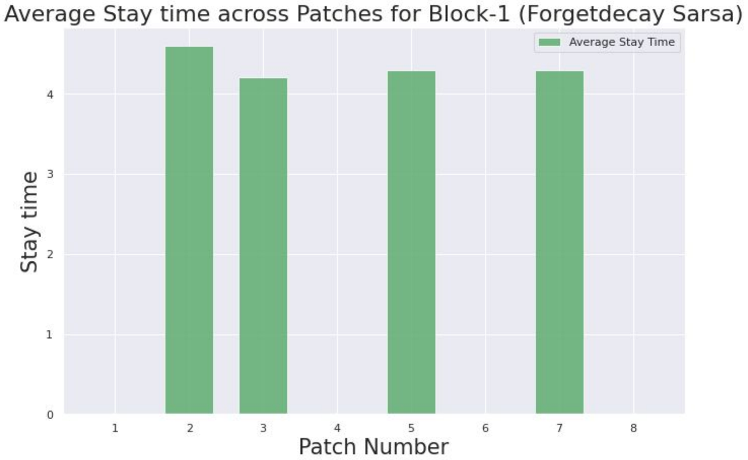


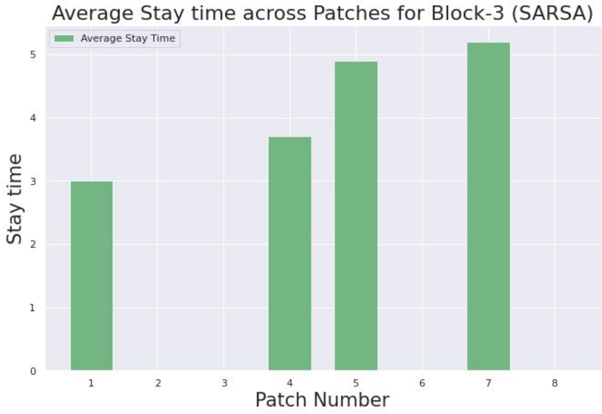

Average Stay time across Patches for Block-1 (Sarsa)
Average Stay Time
Stay time
Patch Number
0
1
2
3
4
1
2
3
4
5
6
7
8


Average Stay time across Patches for Block-1 (q_learning)
Average Stay Time
Stay time
Patch Number
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
1
2
3
4
5
6
7
8

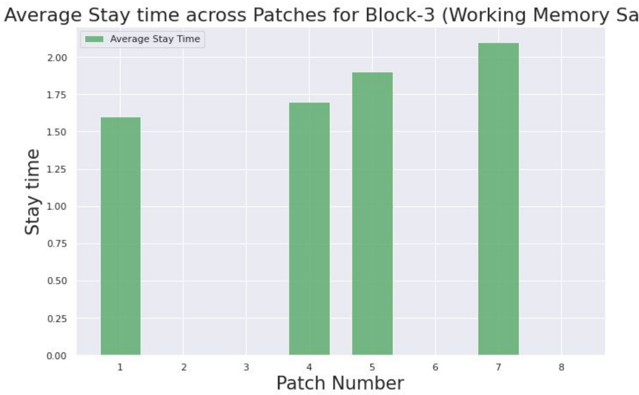

Average Stay time across Patches for Block-1 (Working Memory Sars
Average Stay Time
Stay time
Patch Number
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
1
2
3
4
5
6
7
8

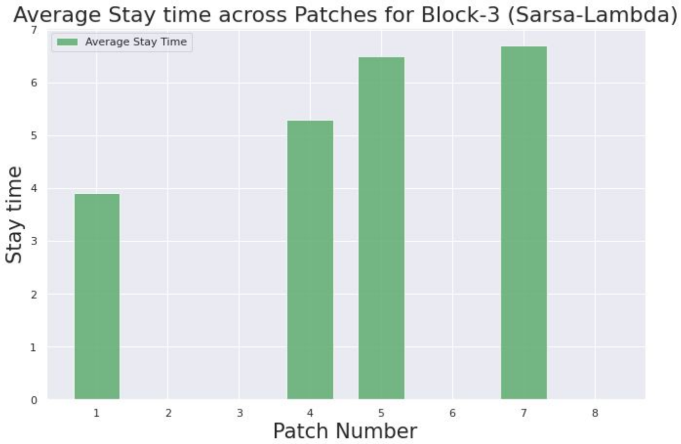

Average Stay time across Patches for Block-1 (Sarsa lambda)
Average Stay Time
Stay time
Patch Number
0
1
2
3
4
5
1
2
3
4
5
6
7
8

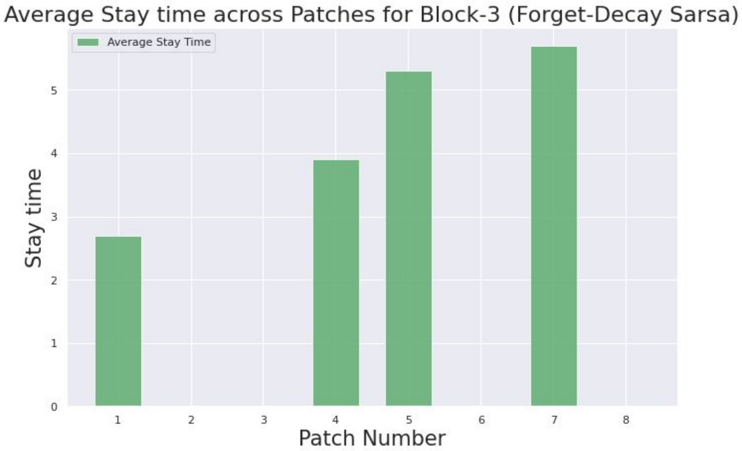

Average Stay time across Patches for Block-1 (Forgetdecay Sarsa)
Average Stay Time
Stay time
Patch Number
0
1
2
3
4
1
2
3
4
5
6
7
8


Average Stay time across Patches for Block-3 (SARSA)
Average Stay Time
Stay time
0
1
2
3
4
5
Patch Number
1
2
3
4
5
6
7
8


Average Stay time across Patches for Block-3 (Q-Learning)
Average Stay Time
Stay time
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Patch Number
1
2
3
4
5
6
7
8


Average Stay time across Patches for Block-3 (Working Memory Sa
Average Stay Time
Stay time
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Patch Number
1
2
3
4
5
6
7
8


Average Stay time across Patches for Block-3 (Sarsa-Lambda)
Average Stay Time
Stay time
0
1
2
3
4
5
6
7
Patch Number
1
2
3
4
5
6
7
8


Average Stay time across Patches for Block-3 (Forget-Decay Sarsa)
Average Stay Time
Stay time
0
1
2
3
4
5
Patch Number
1
2
3
4
5
6
7
8

Deep RL Methods

Assumptions:

- Problem is treated as a Markov Decision Process across individual blocks
- Choice with agent to pick a patch to go to, and then it commits to harvesting
- Multi Layer Perceptrons are used as function approximators for policy, state values (as per the algorithm)

Under these simplifying constraints, following algorithm were tested:

- DQN
- A2C
- PPO

DQN (Deep Q Network)

1. Multi Layer Perceptron with one hidden layer (64 neurons) used for learning state values
2. In value-based model-free reinforcement learning methods, the action value function is represented using a function approximator, such as a neural network.
3. Let $Q(s, a; \theta)$ be an approximate action-value function with parameters θ.
4. Q-learning, aims to directly approximate the optimal action value function: $Q*(s, a) \approx Q(s, a; \theta)$

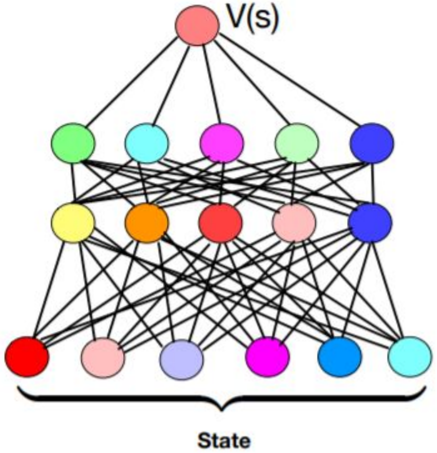
A2C (Advantage Actor Critic)

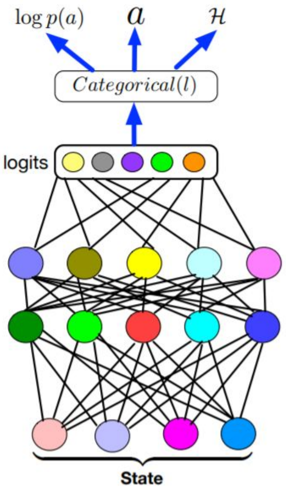
1. Multi Layer Perceptron with one hidden layer (64 neurons) used for learning state values and computing actions
2. Weight sharing between Policy and Value Networks
3. The actor critic algorithm consists of two networks (the actor and the critic)
4. Advantage Function calculates the agent's TD Error or Prediction Error.
5. The actor network chooses an action at each time step and the critic network evaluates the Q-value of a given input state.
6. As the critic network learns which states are better or worse, the actor uses this information to teach the agent to seek out good states and avoid bad states.

PPO (Proximal Policy Optimization)

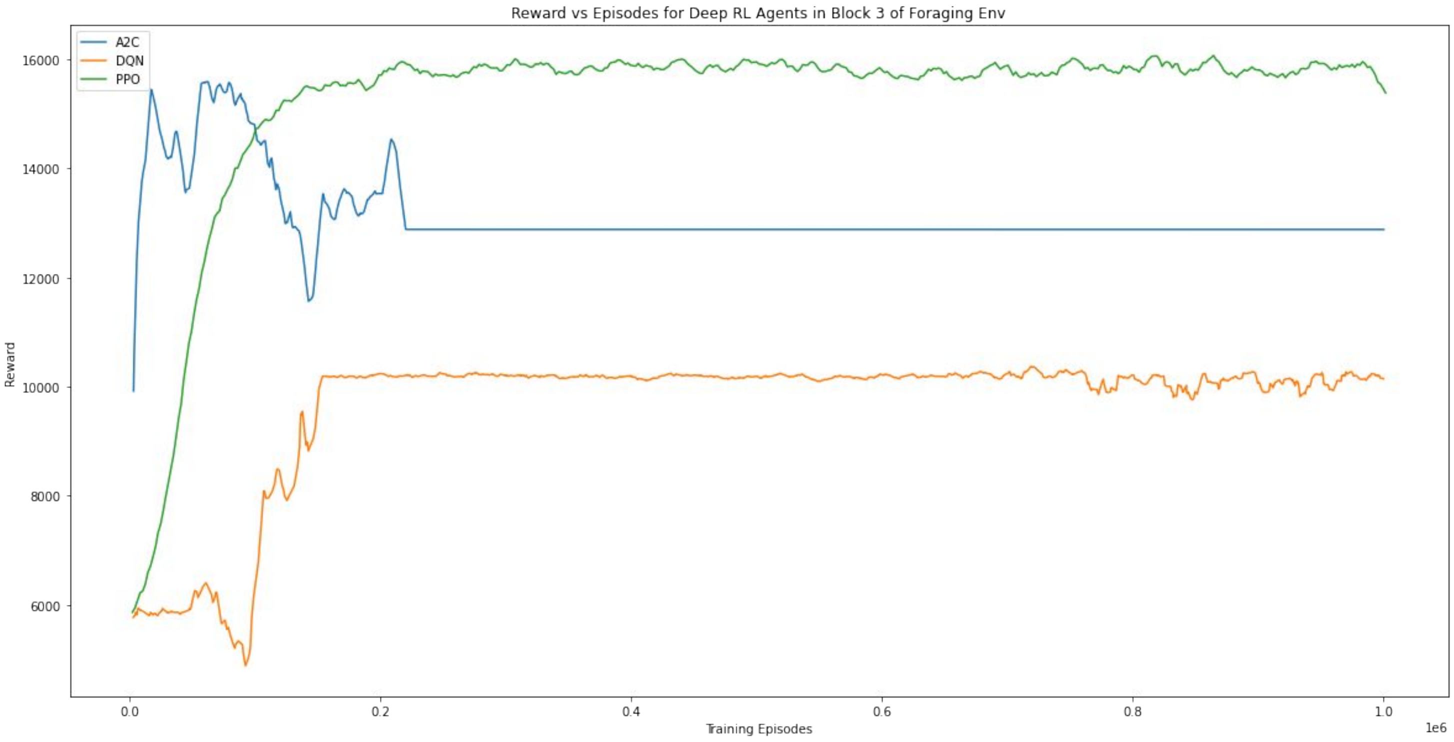
1. Introduces clipped surrogate objective over previous Actor Critic Methods

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right]$$

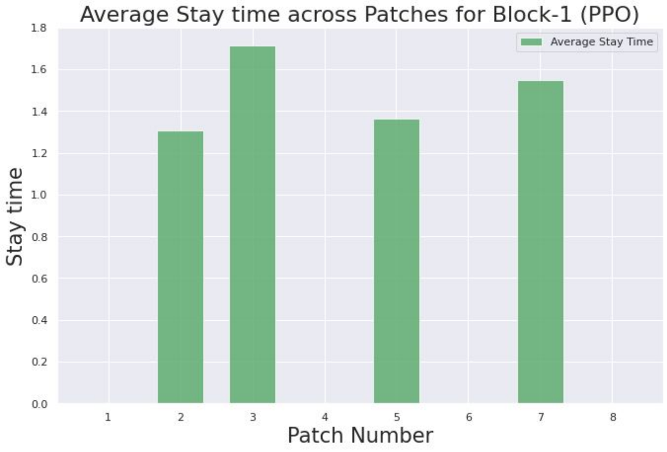


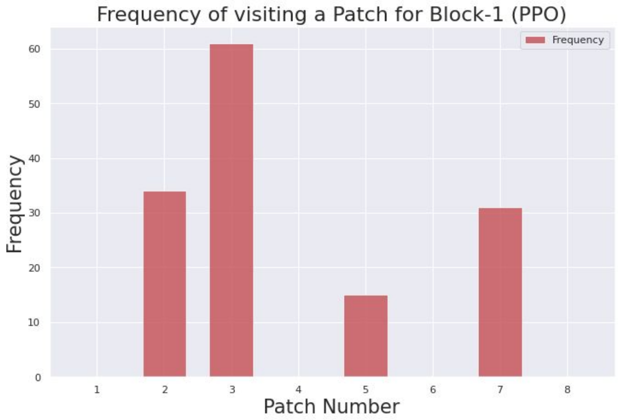


Rewards across Episodes for Deep RL Agents

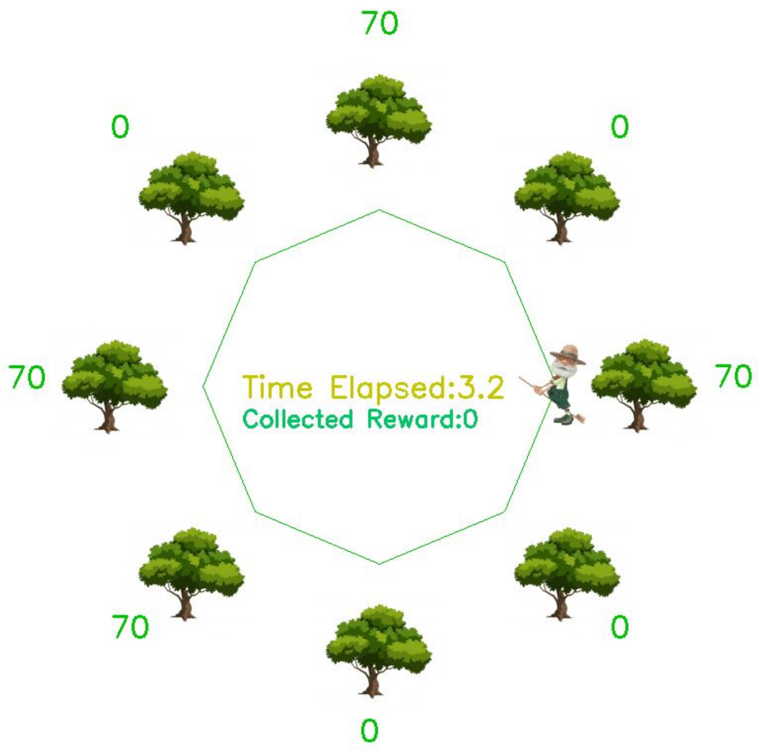


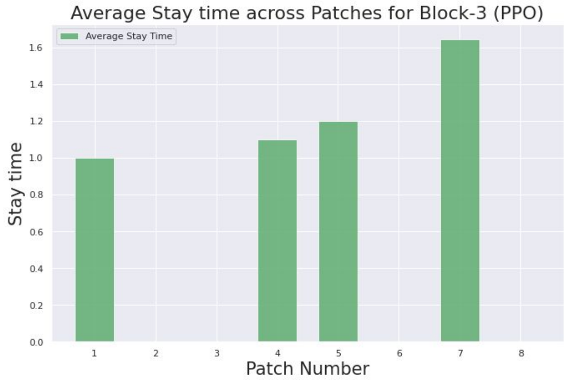
Learnt Policy Simulation for PPO (Block-1)

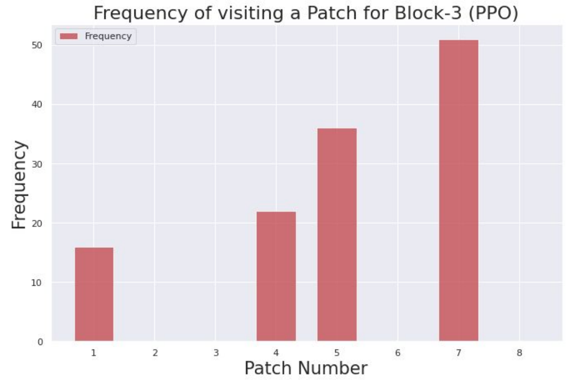




Learnt Policy Simulation for PPO (Block-3)







Discussion and Conclusions

Rewards Obtained by Trained Agents

Method	**Reward in Block 1**	**Reward in Block 3**
MVT (with replenishment)	**19983.5**	23215.8
Multi Armed Bandit (Decaying Epsilon Greedy)	17199	22102.2
SARSA	15842	20999
Q Learning	18814.7	**25732**
Deep RL (PPO)	11610.9	16104

Quantitative Evaluation Block-1

Agent	Convergence reward	Avg Harvest Time	Avg Travel Time
sarsa	15842	196.35	103.35
q_learning	18814	183	117
Sarsa lambda	14997	202	98.
Forgetful sarsa	1143	270.8	29.2
forgetdecay sarsa	14624	200.45	99.5
WM_sarsa	10594	140.765	159.235
MVT	**19983.5**	227.81	72.19

Quantitative Evaluation Block-3

Agent	Convergence reward	Avg Harvest Time	Avg Travel Time
sarsa	20999	205.9	94.1
q_learning	**25732**	176.35	123.65
Sarsa lambda	18339	218.3	81.69
Forgetful sarsa	1591	282.9	15.05
forgetdecay sarsa	19071	207.55	92.45
WM_sarsa (⅝)	13892.355	146.06	153.94
MVT	23215.8	234.83	65.16

Average Staying time Block-1

Agent	0	1 [r=4]	2 [r=4]	3	4 [r=4]	5	6 [r=4]	7
sarsa	0	4.6	3.8	0	3.7	0	4.3	0
q_learning	0	3.5	3.3	0	3.4	0	3.4	0
Sarsa lambda	0	4.4	4.9	0	4.7	0	4.9	0
Forgetful sarsa	0	inf	0	0	0	0	0	0
forgetdecay sarsa	0	4.6	4.2	0	4.3	0	4.3	0
WM_sarsa	0	1.7	1.8	0	1.8	0	1.8	0

Average Staying time Block-3

Agent	0 [r=2]	1	2	3 [r=4]	4 [r=8]	5	6 [r=16]	7
sarsa	3.0	0	0	3.7	4.9	0	5.2	0
q_learning	1.8	0	0	2.7	3.2	0	3.7	0
Sarsa lambda	3.9	0	0	5.3	6.5	0	6.7	0
Forgetful sarsa	inf	0	0	0	0	0	0	0
forgetdecay sarsa	2.7	0	0	3.9	5.3	0	5.7	0
WM_sarsa (5/8)	1.6	0	0	1.7	1.9	0	2.1	0

Human Behaviour Modelling and Explanation

Assumption: The mind & brain are information processing systems.

Memory: Impact of working memory capacity and forgetting on foraging.
Action selection strategy: How actions are selected by human subjects.
Global vs local rewards: Threshold for leaving is based on global or local value.
Reward sensitivity: How sensitive human decisions are to changes in reward.
Initial reward estimation strategy: How initial rewards are estimated.
Short term memory: Eligibility trace in Sarsa lambda.
Learning rate: How fast humans learn from the environment.
Temporal Discounting: How far into the future human bases their decisions on.

Challenges and Limitations

- Identifying which specific events are stored in memory and which are not.
- Adding noise temporally can bias the estimated values.
- Multi Armed Bandits: environment is not strictly a bandit problem as past actions influence decision making.
- Deep RL: black box policy and value networks provide little explainability.

Individual Contributions

Member	Contributions
Abhinav Joshi (20211261)	Gym Environment, Rendering, Literature review, Presentation
Archi Gupta (21111014)	Working Memory, MDP Methods, Literature review, Presentation
Samrudh B Govindaraj (20128409)	Literature Review, MVT and augmentations, Presentation
Shiven Tripathi (190816)	Environment Testing, MAB Methods, Deep RL Methods, Presentation

References

Goldstone, R.L., Ashpole, B.C. Human foraging behavior in a virtual environment. *Psychonomic Bulletin & Review* 11, 508–514 (2004). https://doi.org/10.3758/BF03196603

Hall-McMaster, S., Luyckx, F. Revisiting foraging approaches in neuroscience. *Cogn Affect Behav Neurosci* 19, 225–230 (2019). https://doi.org/10.3758/s13415-018-00682-z

Constantino SM, Daw ND. Learning the opportunity cost of time in a patch-foraging task. Cogn Affect Behav Neurosci. 2015 Dec;15(4):837-53. doi: 10.3758/s13415-015-0350-y. PMID: 25917000; PMCID: PMC4624618.

Matt L. Miller, Kevin M. Ringelman, John M. Eadie, Jeffrey C. Schank, Time to fly: A comparison of marginal value theorem approximations in an agent-based model of foraging waterfowl, Ecological Modelling, Volume 351, 2017, Pages 77-86,

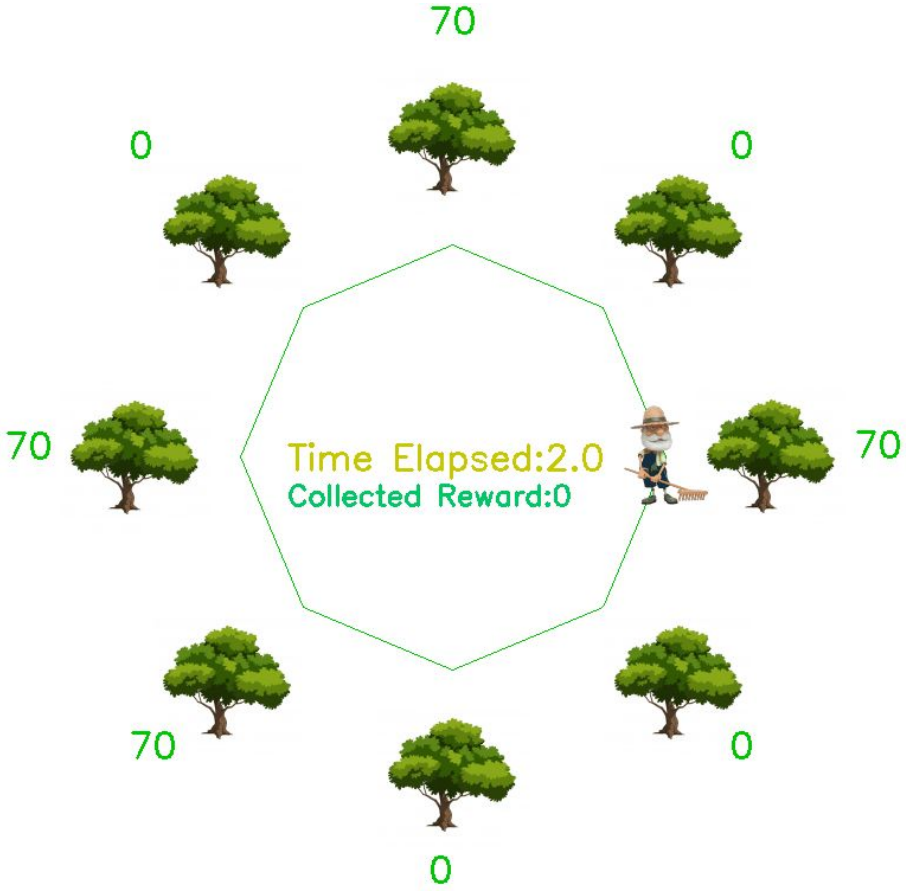
Volodymyr Mnih and Koray Kavukcuoglu and David Silver and Alex Graves and Ioannis Antonoglou and Daan Wierstra and Martin Riedmiller, Playing Atari with Deep Reinforcement Learning, 2013, arxiv

Volodymyr Mnih and Adrià Puigdomènech Badia and Mehdi Mirza and Alex Graves and Timothy P. Lillicrap and Tim Harley and David Silver and Koray Kavukcuoglu, Asynchronous Methods for Deep Reinforcement Learning, 2016, arxiv

Hall-McMaster, S., Dayan, P. & Schuck, N.W., 2021. Control over patch encounters changes foraging behaviour.

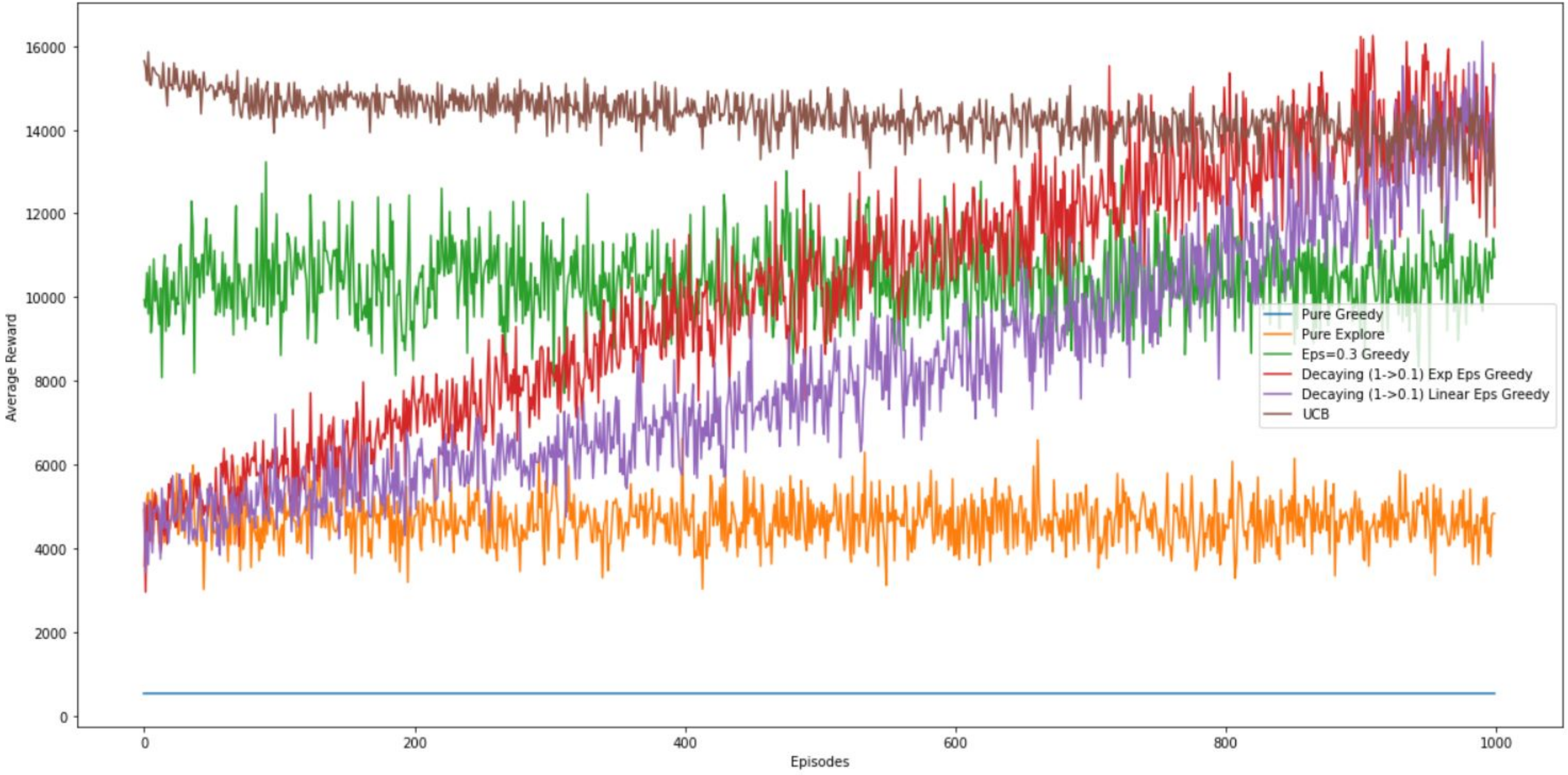
John Schulman and Filip Wolski and Prafulla Dhariwal and Alec Radford and Oleg Klimov, Proximal Policy Optimization Algorithms, 2017, arxiv

Questions

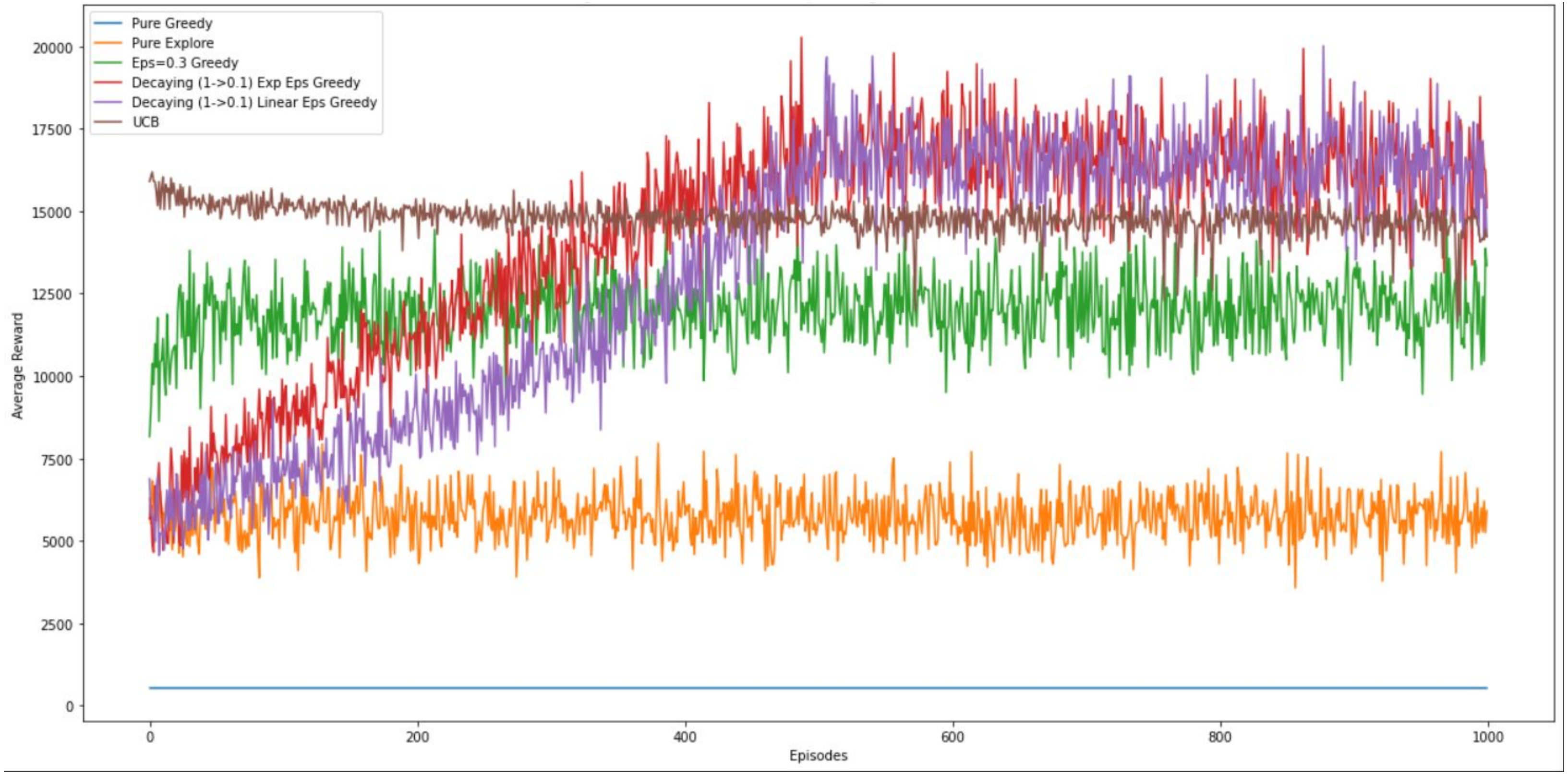


Extra Plots

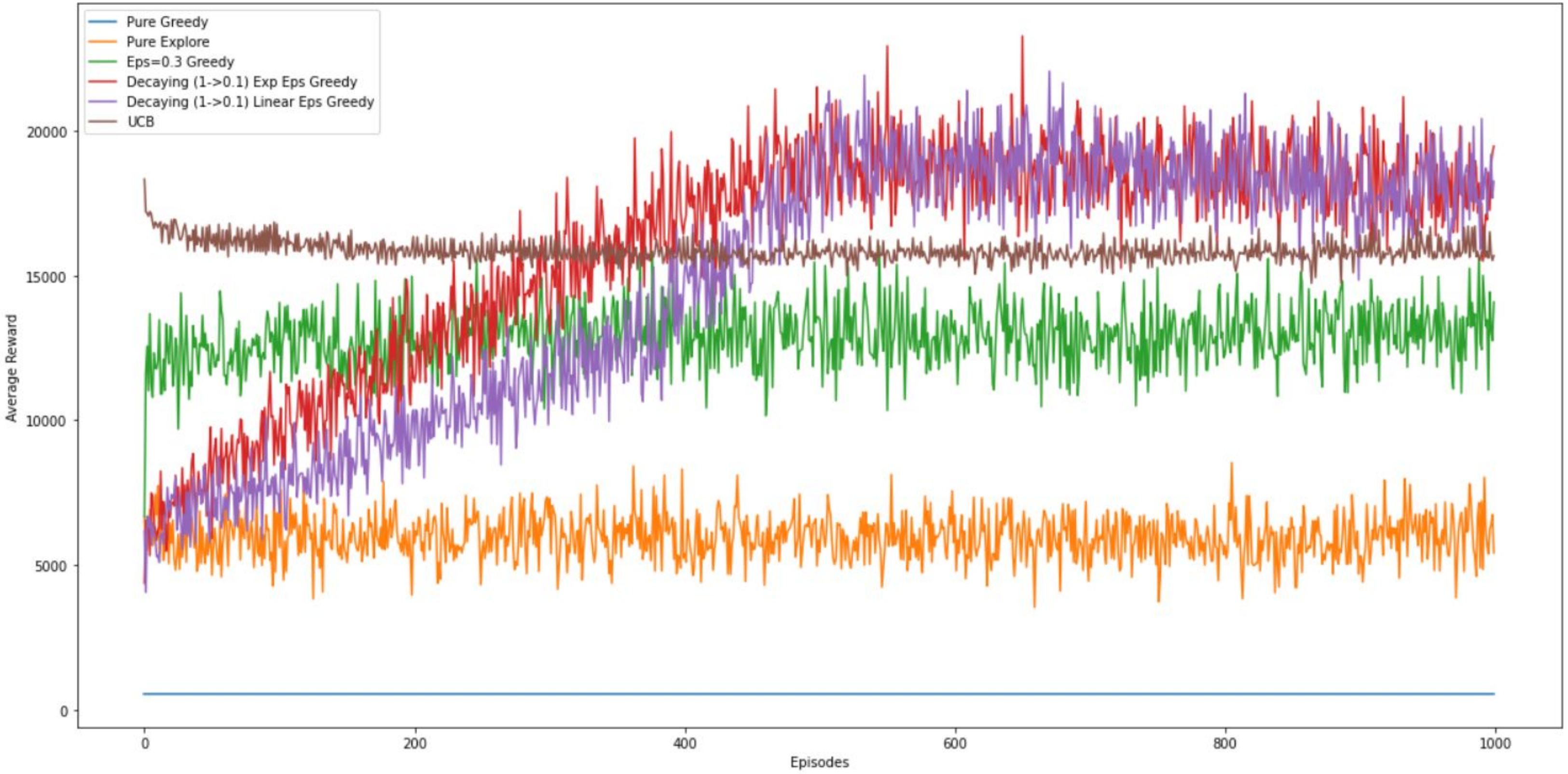
MAB Agent Rewards (block 1, without averaging)



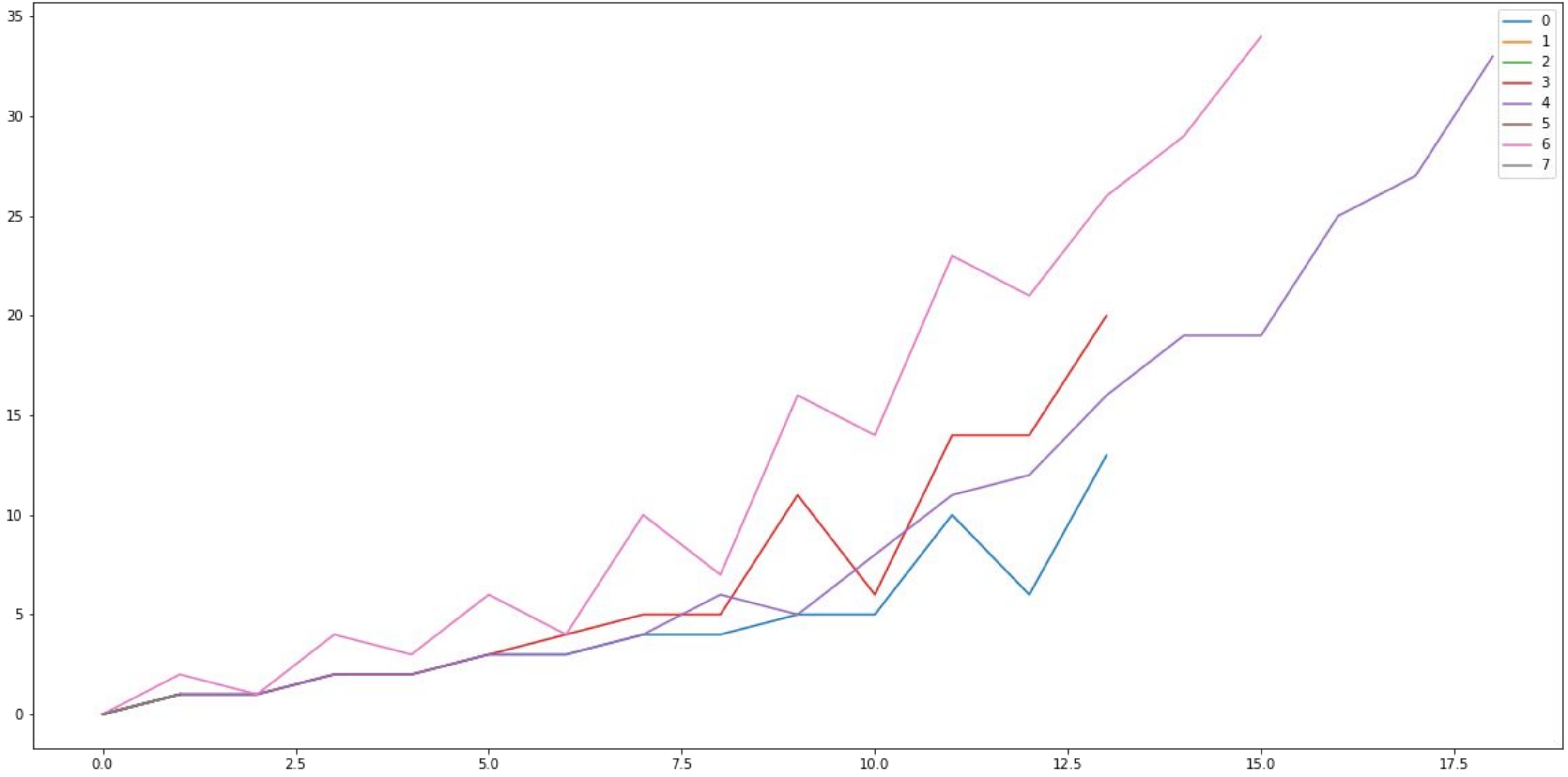
MAB Agent Rewards (block 2, without averaging)



MAB Agent Rewards (block 3, without averaging)



Q-Learning patch stay across time (Block-3)



WM patch stay across time (Block-3)