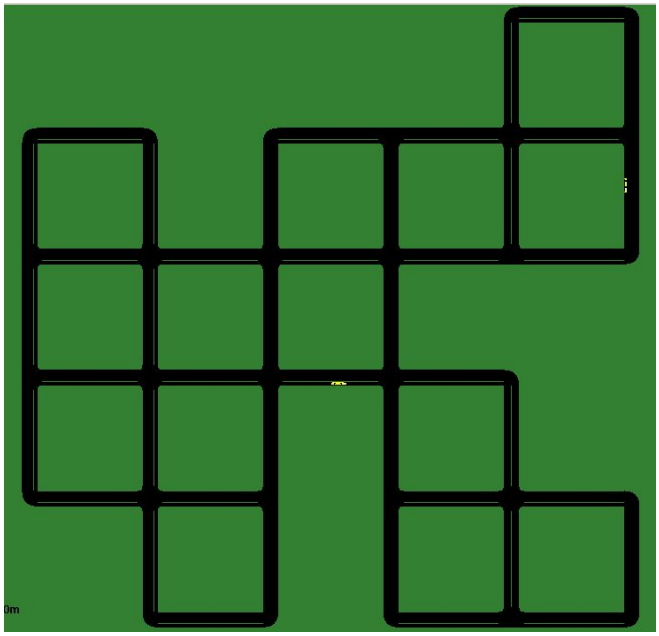
Deep Reinforcement learning for patrolling

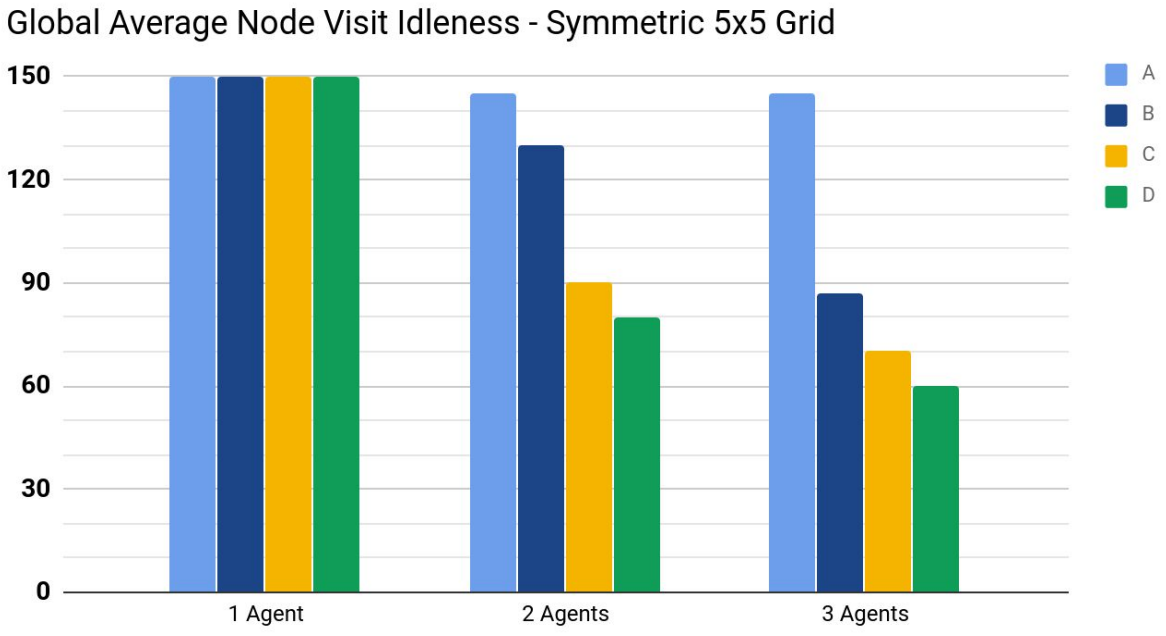
-Meghdeep Jana

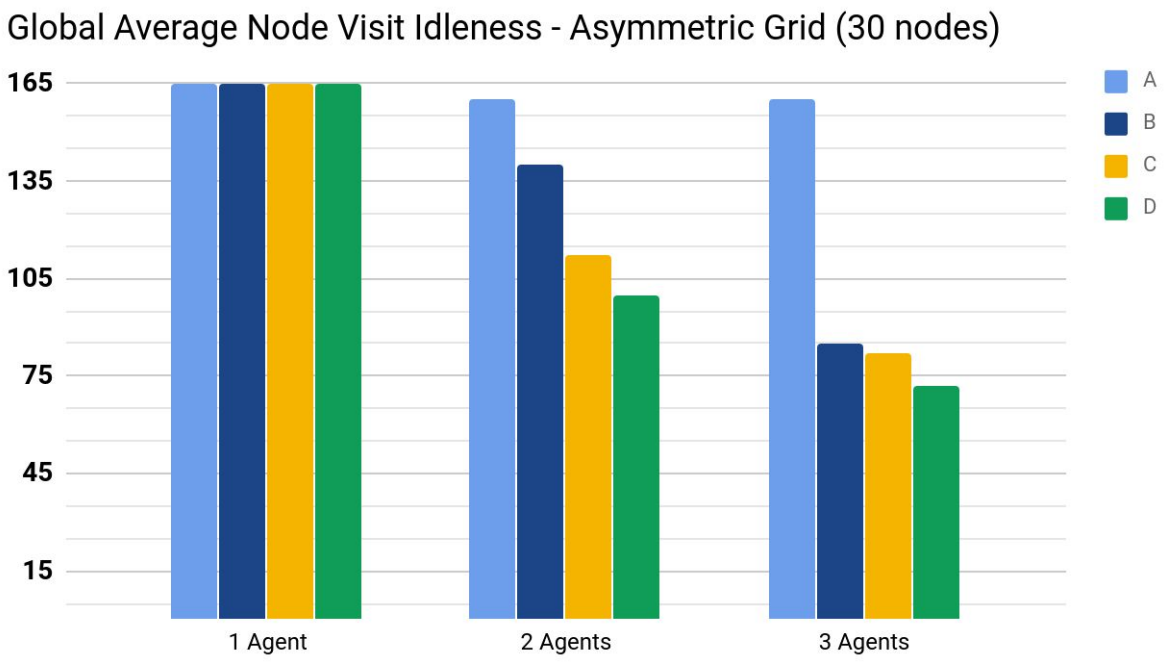
I. **Testing of previous model:**

- Created a 30-node asymmetric environment:



- In addition to the “conflict” negotiation to prevent multiple agents from approaching a single node, added a “tracing” negotiator to prevent agents from following/tracing the other agents’ paths.
- Implemented “negotiator” for 3-agent case in addition to previous 1 and 2-agent cases:
    A. Shared idleness, no negotiation (Machado et. al.)
    B. Shared idleness, coordinated greedy approach with conflict only negotiator
    C. Unshared Idleness, greedy approach (Machado et. al.)
    D. Shared idleness, coordinated greedy approach with both negotiators





II. Reinforcement Learning

A. Assumptions and Constraints:

1. Model of the patrolling environment should follow Markov Property (MDP)
2. Node-to-node idleness values are dynamic, each node cannot be taken as a state in the MDP
3. This leads to about 7000 to 200,000 (single agent to multi-agent) states in the MDP statespace (Santana et.al.) thus tabular/look-up table approach of RL algorithms such as tabular Q-learning(Santana et. al.) becomes inefficient
4. If inter-agent communication is considered, idleness values needs to be shared as states. This makes the state-space unbounded.
5. The reward function as “neighbouring idleness” is a greedy approach, better reward functions might exist that make the agent learn long-term benefitial trajectories
6. The discounted long-term Q-values (reward,action mapping) will lead to agent learning optimal strategies that break locally optimal decisions but lead to globally optimality

B. Solutions proposed:

1. As the state-space is dynamic, high dimensional and could be infinite, I have **proposed the use of Deep Reinforcement Learning** as a function approximator for patrolling as this would help in estimating state space
2. Deep Q-Network(DQN) approach for patrolling is promising in such a scenario, though the training process might lead to oscillations and instabilities

3. Following state-of-the-art methods to avoid instabilities
    a) Experience replay: The “states-actions-reward” transitions experienced by the agents are stored in a buffer memory which is accessed later by the neural networks as inputs in batches of a fixed batch size.
    b) fixed Q-targets: The ground truth of the DQN only updates every constant time steps to avoid “cat-and-mouse” scenario between policy(trajectory) neural network and target(state estimation) neural network
    c) Soft weighted updates: The target network’s parameters are updated by a small factor 0<tau<1 of the policy network
4. State space taken as a semi-observable MDP (Santana et. al):
   [f_n, t_n, idl1, idl2, idl3, idl4]
    a) f_n: node number from where the agent is coming from
    b) t_n: node number to where the agent is going to
    c) Idl1to4: neighbouring idleness values of t_n at that time instant
5. Reward functions:
    a) Idleness value of visiting node, greedy (Santana et. al.)
       [might not be optimal]
    b) Worst idleness value (Lauri et.al)
       [unstable Q value learning, hit-or-miss]
    c) Proposed reward function
6. Randomised exploration extended to the output model: The epsilon-greedy randomised exploration helps the agent during training only, to learn from various random scenarios even though it thinks that it has learned the optimal trajectory. Idea is to extend this randomised exploration to the output model and observe the agent’s behaviour

C. <u>Deep Q-Network Implementation</u>

1. **The proposed reward function:**
   -Proposed reward function encapsules both greedy decision making while keeping in mind minimising the global idleness.
    a) Reward for visiting node_i:

$$Rnode_i = (instant\ idleness\ of\ node_i)^k / (instant\ global\ idleness)$$

    b) The reward for greedily approaching nodes will result in a lower reward if the global idleness value is too high, therefore the agent has to both maximise the numerator while minimising the denominator, resulting in a robust policy
    c) The value of k was varied between k=1 to k=2 in steps of 0.25, k=1.5 was found to give the best results.
2. Training and testing results(single agent case for symmetrical grid):
    a) All the RL methods show better results than the non-RL approaches. The better result can be due to the agent taking

long-term decision from the discounted return natue of the Q-state-action-reward formulation.

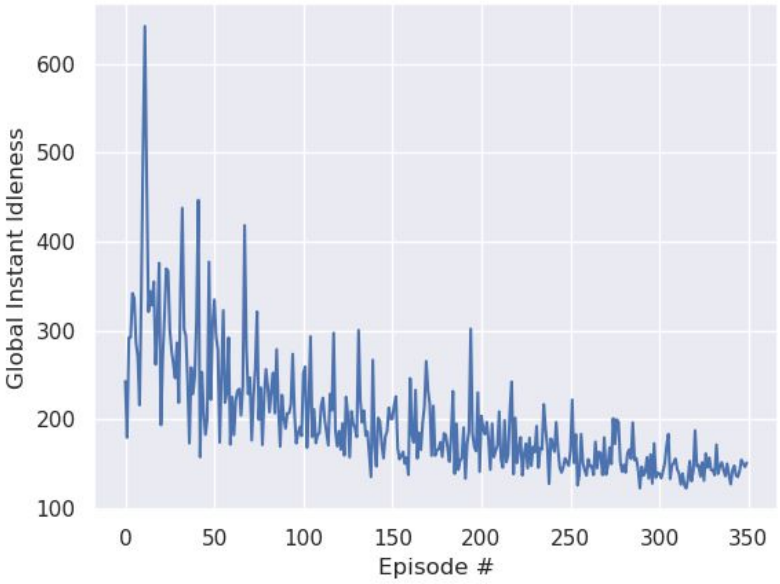
b) The global average node visit idleness was averaging at around 135, with the worst case being around 140 for proposed reward function which The amplitude of global idleness periodicity was low as it was factored in the reward function.

c) The greedy reward function was comparable, was averaging below 140, but worst cases were high, above 150, and the amplitude of global idleness periodicity was higher than case(a)

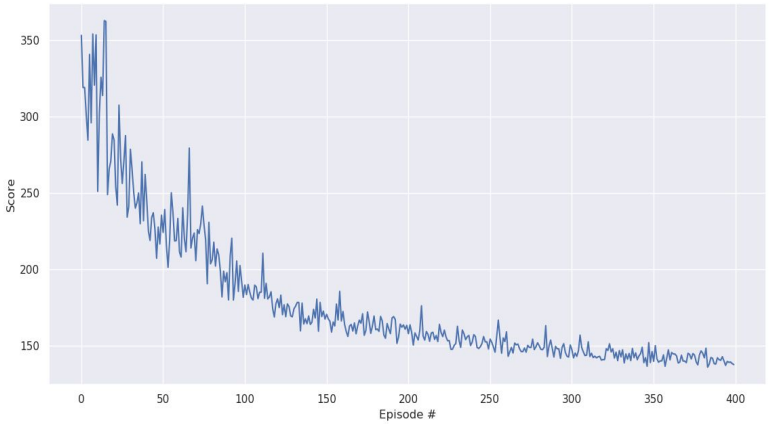
d) The worst case idleness reward function often resulted in unstable trajectories, diverging to high idleness values thus worst case scenarios were bad. But the runs which were stable were averaging around 135 and amplitude was low as well.

3. Graphs:

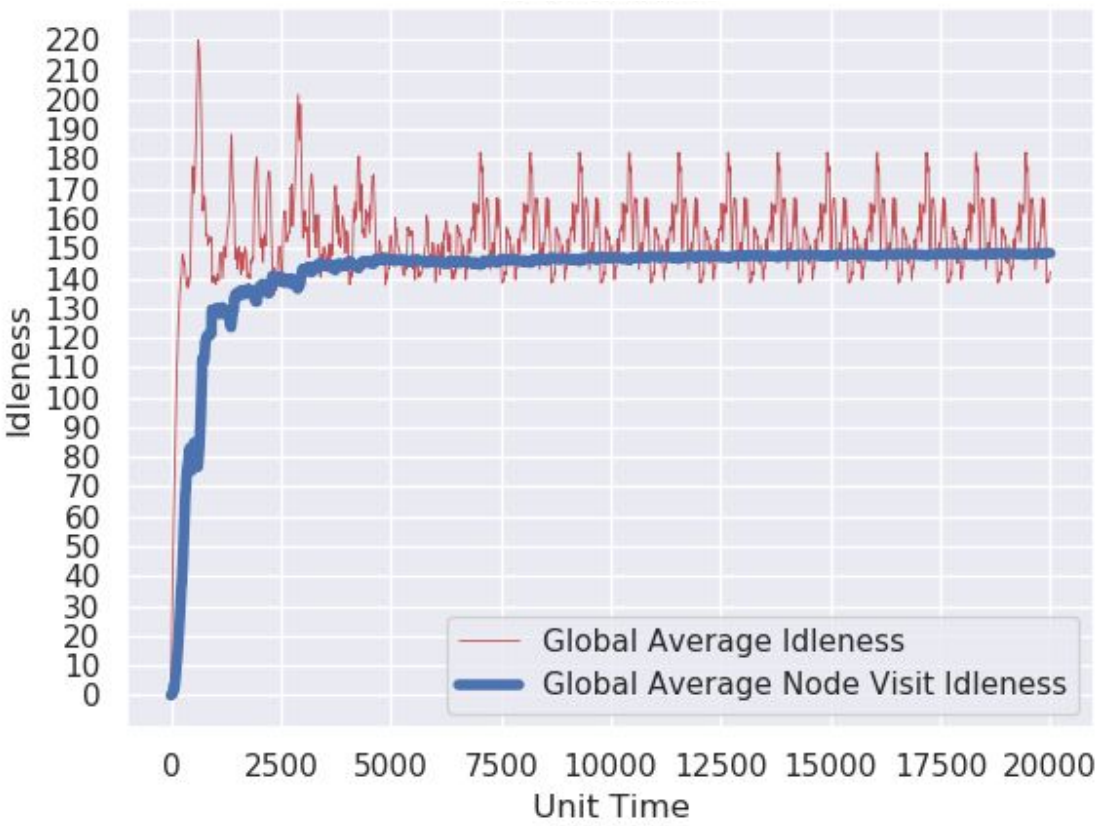
a) Worst case idleness training graph (reward vs no. of runs) [unstable learning, took many tries to reach convergence]



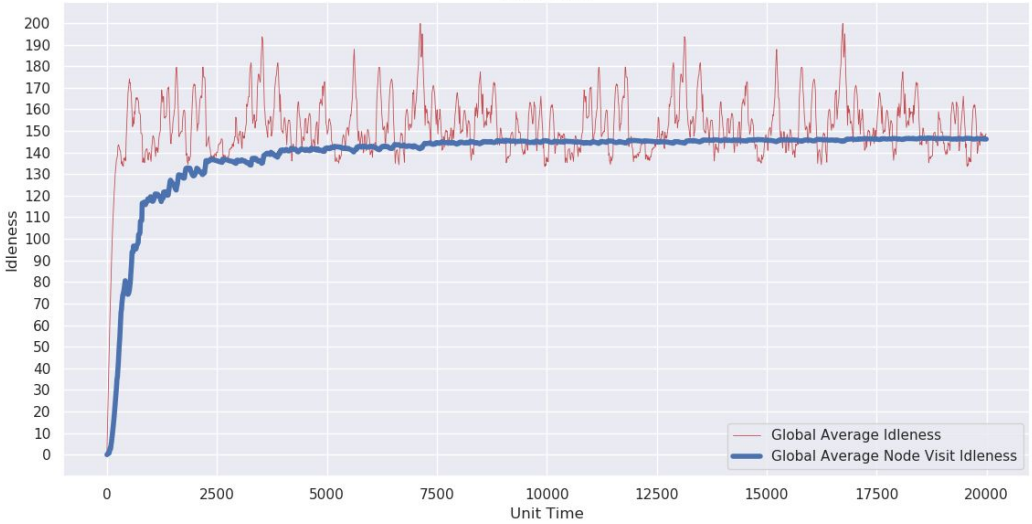
b) Proposed reward training graph (reward vs no. of runs) [stable training]



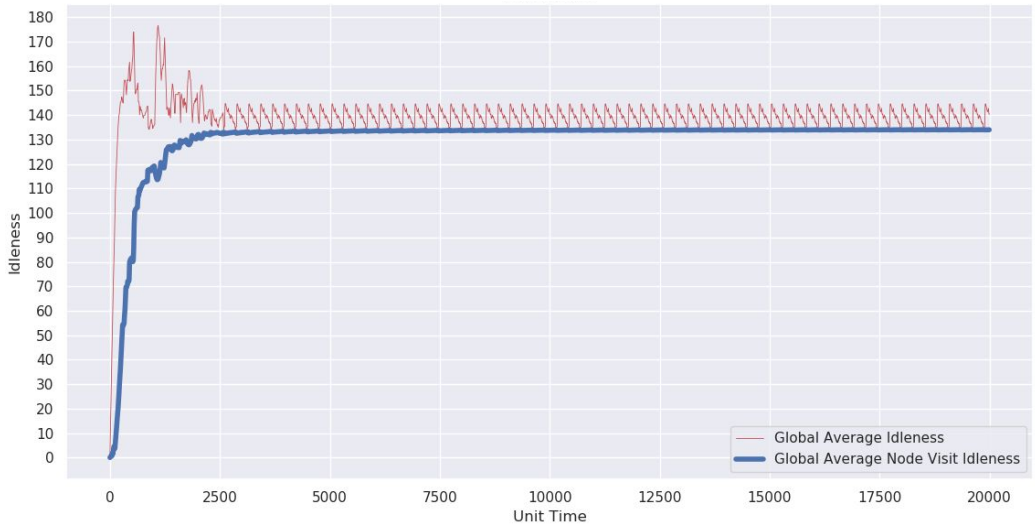
c) Greedy approach idleness graph [high amplitude]



d) Worst case idleness reward graph [ocassionally leads to unstable trajectory]

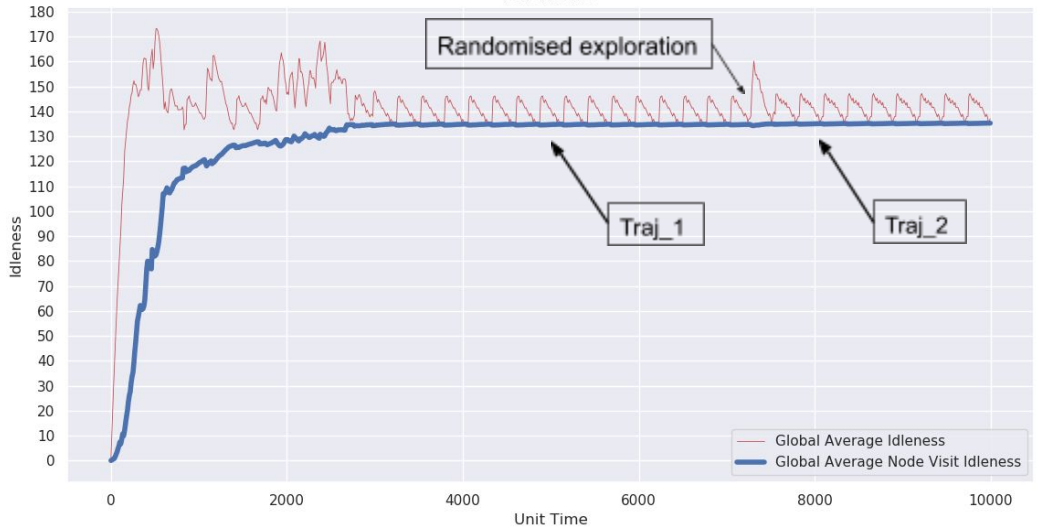


e) Proposed reward function graph, stable with low amplitude.

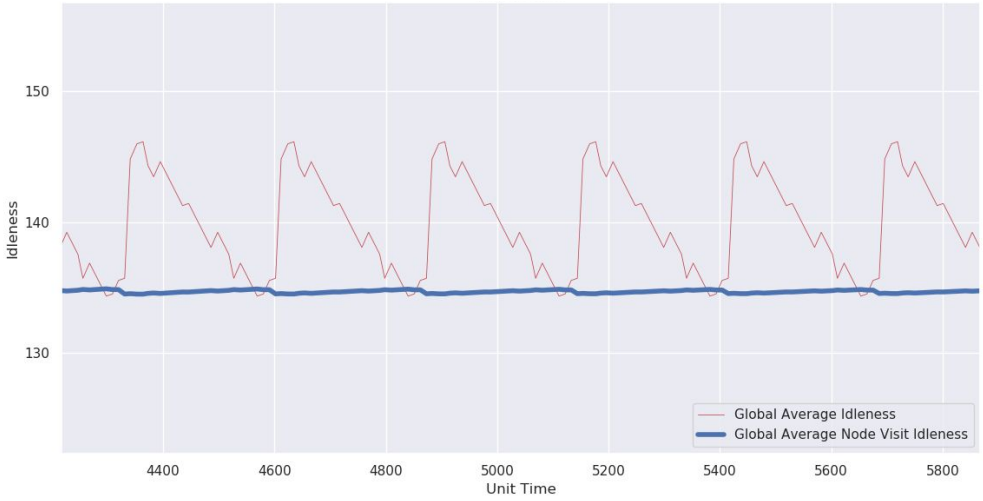


4. **Proposed** Epsilon Greedy Randomised Exploration extended to output RL optimal policy(trajectory)
    a) On top of the proposed reward function, the agent will randomly explore a sub-optimal action with probability epsilon. The best epsilon found from testing was 0.003
    b) With the agent trained on the proposed reward function, a unique phenomenon was observed: **The agent would initially converge to a periodic trajectory with low global idleness value, upon randomised exploration, the agent would lose track of its trajectory, but after some time steps the agent was observed to converge quickly to a different trajectory than before, following a similar trend of idleness values.**
    c) The agent was observed to switch between different trajectories in a randomised time interval without hurting the global idleness.
    d) For non-RL approaches, randomised decisions either lead to increasing idleness values or the agent converging to the same trajectory again
    e) Through this randomised trajectory switching based patrolling, it would be near impossible for any adverserial agent to predict the patrolling agent's trajectory or recognise any patters. Thus, proving to be a promising patrolling strategy.
5. **Further Work:**
    - Train for multiple agents and also for the asymmetric grid
    - Can the RL agent adapt to assymtric grid without re-training?

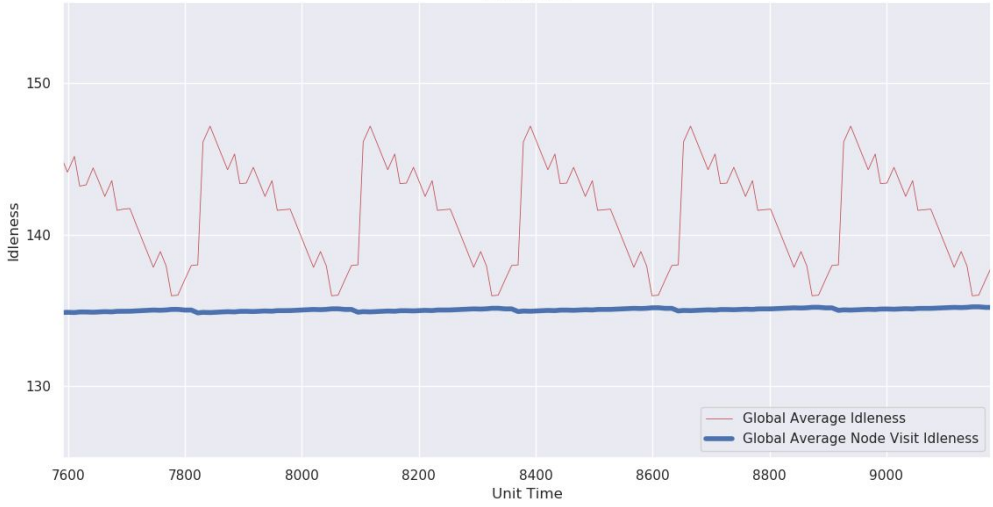
a) The graphs (randomised trajectory switching):



b) Traj_1:



c) Traj_2:



III. References

1. Machado, A., Ramalho, G.L., Zucker, J.-D., Drogoul, A.: Multi-agent Patrolling: An Empirical Analysis of Alternative Architectures. In: Sichman, J.S., Bousquet, F., Davidsson, P. (eds.) MABS 2002. LNCS (LNAI), vol. 2581, pp. 155–170. Springer, Heidelberg (2003)PDF

2. Santana, H., Ramalho, G., et al.: Multi-Agent Patrolling with Reinforcement Learning. In: 3rd International Joint Conference on Autonomous Agents and Multi-Agent Systems, pp. 1122–1129 (2004) PDF

3. Lauri F., Koukam A. (2014) Robust Multi-agent Patrolling Strategies Using Reinforcement Learning. In: Siarry P., Idoumghar L., Lepagnot J. (eds) Swarm Intelligence Based Optimization. ICSIBO 2014. Lecture Notes in Computer Science, vol 8472. Springer, Cham PDF