Appendix for LgTS: Dynamic Task Sampling using LLM-generated sub-goals for Reinforcement Learning Agents

Anonymous Author(s)
Submission Id: 977

ACM Reference Format:
Anonymous Author(s). 2024. Appendix for LgTS: Dynamic Task Sampling using LLM-generated sub-goals for Reinforcement Learning Agents. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 2 pages.

A BASELINES

In this section, we present details on how the baseline approaches used in the paper are implemented.

A.1 Teacher-Student Curriculum Learning

The Teacher-Student Curriculum Learning approach [1] assumes a Teacher agent optimizing the sequence of the tasks in the curriculum for the Student. The Teacher proposes those tasks for which the Student shows the highest potential in learning, i.e. the tasks for which the Student learns the quickest. The ability of a Student to learn a task is given by its slope of the curve of plotted reward function. The steeper the slope, the quicker the Student is able to learn the task. One limitation of this approach is that it assumes the source tasks of the curriculum are defined. This is difficult in settings where the parameters of the task are continuous. We used the same PPO network (Hyperparameters in Section C) for Teacher-Student curriculum learning approach as we did for our proposed approach *AGTS*. We initialized 3 source tasks and one target task, with varying parameters for each source task. The tasks were: (1) Collect Key_1; (2) Collect Key_2; (3) Open $Door$; (4) Final task of reaching $Goal$. For each task, the agent starts from an initial state of the environment, and attempts a task proposed by the Teacher agent. For a fair comparison, the number of *interactions allocated* for this task is 500 to acknowledge the difficulty of the entire task.

For the tabular results in the paper, the performance of the agent in the final target task is displayed. The agent was not able to learn a successful policy for the entire task.

In Teacher-Student curriculum learning, the sunk cost of the algorithm is the interactions in the source task. The sunk cost is very high when the interactions are costly, as for our robotic environments.

A.2 Automaton-guided Teacher-Student

Automaton-guided Teacher-Student learning (AgTS) where the graphical structure of the sub-goal is generated using the finite-trace Linear Temporal Logic (LTL_f) formula given by an oracle. For this task, the LTL_f formula is: $\varphi_f := \mathbf{G}\neg Lava \wedge \mathbf{F}((Key_1|Key_2) \wedge \mathbf{F}(Door \,\&\, \mathbf{F}(Green\_Goal)))$ where G and F represent *Always* and *Eventually* respectively. We use the equivalent DFA representation of the above LTL_f formula as the graphical representation, and perform the Teacher-Student learning approach outlined in section 4. The LTL_f formula and the automaton is given in fig. 1.

$$\mathbf{G}\,\neg l \;\wedge\; \mathbf{F}\,((\,k_1 \mid k_2\,) \;\wedge\; \mathbf{F}\,(\,d \,\&\, \mathbf{F}\,(\,g\,)))$$

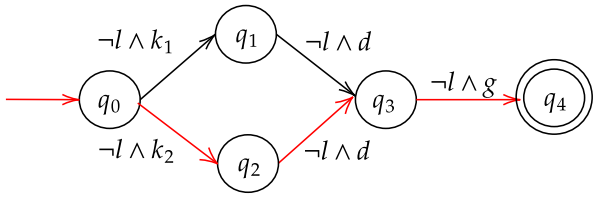

Figure 1: LTL_f formula and corresponding automaton as defined by an oracle. G and F corresponds to *Always* and *Eventually* respectively. $l, k_1, K-2, d$ and g correspond to $Lava, Key_1, Key_2, Door$ and $Green\_Goal$ respectively.

A.3 Automaton-guided Reward Shaping

For the automaton-guided reward shaping baseline, we used the LTL_f formula and the automaton as defined by the oracle in fig. 1. Every time the agent transitioned from one state in the DFA to another, the agent received a small positive reward inversely proportional to the distance from the goal node. This makes the task non-Markovian in nature as the shaped reward the agent receives depends on the high-level path the agent followed in that episode.

A.4 LLM-guided Reward Shaping

For the LLM-guided reward shaping baseline, we used the DAG given by the LLM. Every time the agent transitioned from one state in the DAG to another, the agent received a small positive reward inversely proportional to the distance from the goal node. This makes the task non-Markovian in nature as the shaped reward the agent receives depends on the high-level path the agent followed in that episode.

B DISCUSSION ON n - THE NUMBER OF SUB-GOAL SEQUENCES

We performed additional experiments to observe how LgTS performs when the number of sub-goal path n varies w.r.t to the number of objects and predicates.

To summarize, we observed high interaction cost and low success rate when n was too low (1 or 2), denoting that the LLM fails to consider different paths for satisfying the goal and generates a path that does not succeed given an unknown environment configuration.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand.

Parameter	Value
discount factor γ	0.99
learning rate α	$1 \times 10^{-}3$
Optimizer	Adam
PPO clipping parameter	0.2
GAE λ	0.95
Entropy regularization coefficient	0.001
Entropy regularization coefficient	0.001
Action distribution	Categorical with 6 bins
Network architecture	2 Conv layers of [64, 64] followed by 2 linear layers with $relu$ activation

Table 1: Parameters used for training the Proximal policy optimization

This is due to the fact that when n was too low, the path suggested by the LLM mostly involved picking up the sub-optimal Key (Key_1) and then unlocking the door. When the agent tried learning policies for these sub-goals, the agent was unable to learn successful policies as the sub-task $Key_1 \rightarrow Door$ is quite complex.

Similarly, when n was too high (12-13), the paths suggested by the LLM involved several transitions in the graphical representation of the task. While it included the optimal path, it also included several sub-optimal paths. This made the LLM-guided Teacher-Student learning algorithm consider several different sub-tasks, making the goal-reaching behavior sample inefficient.

The number of paths requested from the LLM depends on the complexity of the problem, which in turn depends on the number of entities and predicates present in the environment. We observed that a good metric for choosing the number of paths was equal to the number of entities present in the environment. Thus the graph will have at least one path that involves an entity, and thus the LgTS approach should be able to find a set of policies for achieving the high-level goal objective.

C HYPERPARAMETERS

Table 1 summarizes the hyperparameters for the PPO [2] used for the experiments.

All the experiments were conducted using a 64-bit Linux Machine, having Intel(R) Core(TM) i9-9940X CPU @ 3.30GHz processor, 126GB RAM memory and an Nnidia 2080 GPU. The maximum duration for running the experiments was set at 24 hours.

REFERENCES

[1] Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. 2020. Teacher-Student Curriculum Learning. *IEEE Trans. Neural Networks Learn. Syst.* 31, 9 (2020), 3732–3740. https://doi.org/10.1109/TNNLS.2019.2934906

[2] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* (2017). arXiv:1707.06347 http://arxiv.org/abs/1707.06347