Subgoal Reachability in Goal Conditioned Hierarchical Reinforcement Learning

Michał Bortkiewicz[1][a], Jakub Łyskawa[1][b], Paweł Wawrzyński[1,2][c], Mateusz Ostaszewski[1][d], Artur Grudkowski[1], Bartłomiej Sobieski[1] and Tomasz Trzciński[1,2,3,4,5][e]

[1]*Warsaw University of Technology, Institute of Computer Science, Poland*

[2]*IDEAS NCBR, Poland*

[3]*Jagiellonian University, Poland*

[4]*Tooploox, Poland*

[5]*Ensavid, Poland*

Keywords: Hierarchical Reinforcement Learning, Deep Reinforcement Learning, Control.

Abstract: Achieving long-term goals becomes more feasible when we break them into smaller, manageable subgoals. Yet, a crucial question arises: how specific should these subgoals be? Existing Goal-Conditioned Hierarchical Reinforcement Learning methods are based on lower-level policies aiming at subgoals designated by higher-level policies. These methods are sensitive to the proximity threshold under which the subgoals are considered achieved. Constant thresholds make the subgoals impossible to achieve in the early learning stages, easy to achieve in the late stages, and require careful manual tuning to yield reasonable overall learning performance. We argue that subgoal precision should depend on the agent's recent performance rather than be predefined. We propose Adaptive Subgoal Required Distance (ASRD), a drop-in replacement method for subgoal threshold creation that considers the agent's current lower-level capabilities for appropriate subgoals. Our results demonstrate that subgoal precision is essential for HRL convergence speed, and our method improves the performance of existing HRL algorithms.

1 INTRODUCTION

Hierarchical reinforcement learning (HRL) performs remarkably well on complex tasks, unsolvable by flat methods (Gehring et al., 2021; Gürtler et al., 2021; Nachum et al., 2018; Levy et al., 2017; Ghosh et al., 2019; Eysenbach et al., 2019). This is because the control of sequential decision making in complex dynamical systems is often easier to synthesize when decomposed hierarchically (Nachum et al., 2019). The high-level agent breaks down the problem into a series of subgoals to be sequentially executed by the low-level policy. To illustrate this concept, consider how a child learns to walk: they don't need to master it perfectly from the beginning; instead, they initially grasp the fundamental dynamics of the skill and then progressively refine it as they go along (Adolph et al.,). This hierarchical approach simplifies learning higher-level skills while continuously improving basic abilities.

Most existing works in Goal Conditioned HRL (GCHRL) assume fixed criteria of *subgoal required distance* (SRD), i.e. a radius of the region in the state space, which the agent should reach to accomplish a subgoal (Nachum et al., 2018; Gürtler et al., 2021; Lee et al., 2022). In a sparse reward setting, if the lower-level (LL) policy achieves SRD within a higher-level (HL) action time, it receives a positive reward. Notably, SRD is usually predefined by a human expert or found using a hyperparameter search (Chane-Sane et al., ; Colas et al., ; Liu et al., 2022).

However, two serious issues can arise from a predefined SRD when the HL policy is not well-trained. If the SRD is too small, it can create subgoals that are too narrow and impossible to achieve. As a result, the LL policy may never accomplish any subgoal, making it difficult to learn anything. However, if the SRD is too large, the LL policy may learn to reach given subgoals

[a] https://orcid.org/0000-0001-5470-7878
[b] https://orcid.org/0000-0003-0576-6235
[c] https://orcid.org/0000-0002-1154-0470
[d] https://orcid.org/0000-0001-7915-6662
[e] https://orcid.org/0000-0002-1486-8906

Bortkiewicz, M., Łyskawa, J., Wawrzyński, P., Ostaszewski, M., Grudkowski, A., Sobieski, B. and Trzciński, T.
Subgoal Reachability in Goal Conditioned Hierarchical Reinforcement Learning.
DOI: 10.5220/0012326200003636
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Conference on Agents and Artificial Intelligence (ICAART 2024) - Volume 1*, pages 221-230
ISBN: 978-989-758-680-4; ISSN: 2184-433X
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

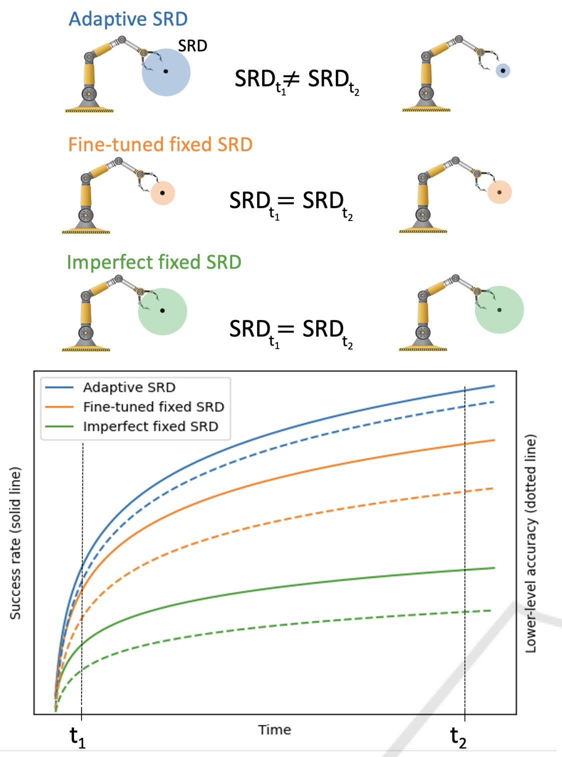

Figure 1: Conceptual illustration of the SRD impact on algorithm performance. Fixed SRD yields high-performing algorithms only when adjusted to agent capabilities. However, in practice, it usually needs to be found using fine-tuning. Thus, too wide or narrow SRD may limit LL policy accuracy in subgoal reaching and result in lower performance.

imprecisely, leading to clumsy control and unsatisfactory progress in the main task. Thus, this approach is sensitive to SRD and works only with expert domain knowledge or after meticulous SRD tuning (Fournier et al., 2018). In this work, we empirically demonstrate this sensitivity.

We recognize this limitation in GCHRL and address it by adjusting SRD along the training. We argue that precision is more prominent in later training phases but may be eased initially. Thus, the SRD should depend on the agent's recent performance rather than be fixed. We propose a simple method that adapts the SRD to the current performance of the agent as measured during the training process, forming a curriculum of subgoal thresholds (Figure 1). We indicate that curricular approaches were widely surveyed regarding exploration (Portelas et al., 2020; Zhang et al., 2020b; Li et al., 2021); however, few works explored it in the context of subgoal reachability issue (Fournier et al., 2018), especially in HRL.

This work closely examines the problem of tuning SRD in GCHRL. Our contributions are as follows:

1. We show that HRL methods are highly sensitive to subgoal precision and environment goal precision.
2. We propose Adaptive Subgoal Required Distance (ASRD), a novel, easy-to-implement method adapting SRD that boosts training robustness, improves final policy performance and simplifies the hyperparameter tuning procedure.
3. We perform an extensive analysis of the proposed method, showing that it allows us to obtain higher control precision faster.

2 PROBLEM STATEMENT

We consider the typical RL setup (Sutton and Barto, 2018) based on a Markov Decision Process (MDP): An agent operates in its environment in discrete time $t = 1,2,\dots$. At time t it finds itself in a state, $s_t \in \mathcal{S}$, performs an action, $a_t \in \mathcal{A}$, receives a reward, $r_t \in \mathbb{R}$, and the state changes to s_{t+1}. The agent is trained to maximize the discounted rewards sum $\mathbb{E}(\sum_{i=0}\gamma^i r_{t+i}|s_t)$ for each state s_t where $\gamma \in [0,1)$ is the discount factor.

We assume that effective hierarchical control is possible in this MDP. Let there be $L > 1$ levels of the hierarchy. Each l-th level defines an MDP with its state space $\mathcal{S}^l$, its action space $\mathcal{A}^l$ and rewards. For the highest level $\mathcal{S}^L = \mathcal{S}$ and for the bottom level $\mathcal{A}^1 = \mathcal{A}$. Taking action at l-th level, $l \geq 2$, a_t^l, launches an episode of the MDP at $l-1$-st level. a_t^l defines the goal in this lower-level episode and may specify the time to achieve this goal, thus the states and rewards in this episode are co-defined by a_t^l. Once this episode is finished, another action at l-th level is taken.

Actions at the l-th level are defined by a policy,

$$a_t^l \sim \pi^l(\cdot|s_t^l),\ l = L,\dots,1, \tag{1}$$

where s_t^l is the state of the agent perceived at l-th level of the hierarchy at time t. For $l < L$, the goal of the episode at level l is for a certain state projection, $f^l(s_t)$, to approach a given goal $g_t^l \in G^l$ (optionally, g_t^L is the environment goal in the current episode). For $l < L$, $G^l = A^{l+1}$ The f^l function, $f^l : \mathcal{S} \to G^{l+1}$, usually just extracts certain coordinates from its vector input. For $l < L$, the state s_t^l also includes a subgoal of l-th level g_t^l. SRD value $\varepsilon^l > 0$ specifies the *precision* with which the policy π^l must approach the state projection $f^l(s_t)$ of subgoal g_t^l. The reward for the l-th level r_t^l depends on the successful approaching of $f^l(s_t)$ in the assumed time.

The overall task considered in this paper is to learn the hierarchy of policies (2) so that the expected sum of future discounted rewards is maximized in each state at each hierarchy level.

3 RELATED WORK

Hierarchical Reinforcement Learning (HRL) is a framework that introduces hierarchy into control by decomposing a Markov Decision Process (MDP) into smaller MDPs. This approach enhances structured exploration and facilitates credit assignment (family=Vries et al.,), particularly in scenarios with sparse rewards (Pateria et al., ; Nachum et al., 2019). HRL comprises three main branches: option-based (Sutton et al., 1999; Barto and Mahadevan, 2003; Precup, 2000; Bagaria and Konidaris, 2019; Shankar and Gupta, 2020), skill-based (Eysenbach et al., 2018; Sharma et al., 2019; Campos et al., 2020), and goal-based.

Our research primarily focuses on the goal-conditioned hierarchical reinforcement learning (GCHRL) approach, where the high-level policy communicates with a lower-level policy (Schmidhuber, 1991; McGovern and Barto, 2001; Vezhnevets et al., 2017; Zhang et al., 2020a). In this setup, the high-level policy typically returns a subgoal that conditions the lower-level policies. Consequently, the low-level policy is rewarded for approaching the designated subgoal. Nevertheless, training a hierarchy of policies using prior experience introduces non-stationarity issues (Jiao and Tsuruoka,), as selecting subgoals for lower-level policies can yield different results due to policy changes during learning. To address this challenge, various methods of subgoal re-labelling have been proposed.

Levy et al. (Levy et al., 2017) introduced hierarchical experience replay (HAC), employing hindsight experience replay, where actual states achieved are treated as if they had been selected as subgoals. On the other hand, Nachum et al. (Nachum et al., 2018) presented HIRO, a method based on off-policy correction, where unattained subgoals in transition data are re-labelled with alternatives drawn from the distribution of subgoals that maximize the probability of observed transitions.

Furthermore, recent advancements have enhanced high-level guidance and introduced timed subgoals in Hierarchical Reinforcement Learning with Timed Subgoals (HiTS) (Gürtler et al.,). This method enhances the precision of communication between hierarchical levels, which is particularly crucial in dynamic environments.

Nevertheless, as far as we can tell, there is a notable absence of literature on utilising varying SRD within a hierarchical setup, even though there is an extensive body of knowledge concerning subgoals impact on exploration (Portelas et al., 2020; Zhang et al., 2020b; Li et al., 2021). In HRL, varying subgoal criteria emerge naturally due to the ongoing training of low-level and high-level policies. Therefore, we must gain a deeper understanding of this interaction.

3.1 Varying Subgoal Achievement Criterion in Sparse Reward GCHRL

In most of the GCRL methods (Levy et al., 2017; Gürtler et al.,), the goal (or subgoal) is reached when the distance between each of the corresponding elements of the current state and the subgoal state g_t^l is smaller than $\varepsilon^l > 0$. Thus, the goal is not reached when part of the achievement criteria is unsatisfied. This formulation enforces strict control over goal-conditioned policy. However, (Fournier et al., 2018) shows that reachability is poor at the beginning of the training and adaptive strategies of SRD improve the training efficiency in flat RL.

The SRD impacts the performance of goal-conditioned RL (Liu et al., 2022), especially GCHRL methods with sparse rewards on every level. This is because HRL methods like HAC (Levy et al., 2017) and HiTS (Gehring et al., 2021) use a mechanism to test if the subgoals defined by the higher level are attainable. This mechanism, dubbed testing transitions, results in additional penalties for HL if generated subgoals are not reachable by the deterministic LL policy. As a consequence, HL policy is encouraged to produce mostly reachable subgoals.

However, the frequent cause of failure of GCHRL methods with sparse rewards and testing transition mechanism arises from the wrong ratio of HL rewards environmental and reachability (Levy et al., 2017). Usually, the environment yields sporadic positive feedback for task completion, and reachability rewards are designed a priori by the algorithm author. The failure manifests itself with optimization of reachability rather than environmental return, which results in easily achievable subgoals that do not lead to task progression. In practice, there is a narrow range of possible penalty values for testing transitions that do not interfere with environmental reward, i.e. balances these two sources of the reward signal. Thus, for efficient learning of the HRL method, there is a need to either find a proper hyperparameter of the percentage of testing transitions or ε^l that defines the subgoal SRD.

We propose a new method that effectively identifies suitable SRDs to address this limitation. Unlike Fournier's approach (Fournier et al., 2018), our method does not make any assumptions about the SRD and instead learns it directly from past experience. This makes our approach more flexible and applicable to a wider range of scenarios with limited prior knowledge

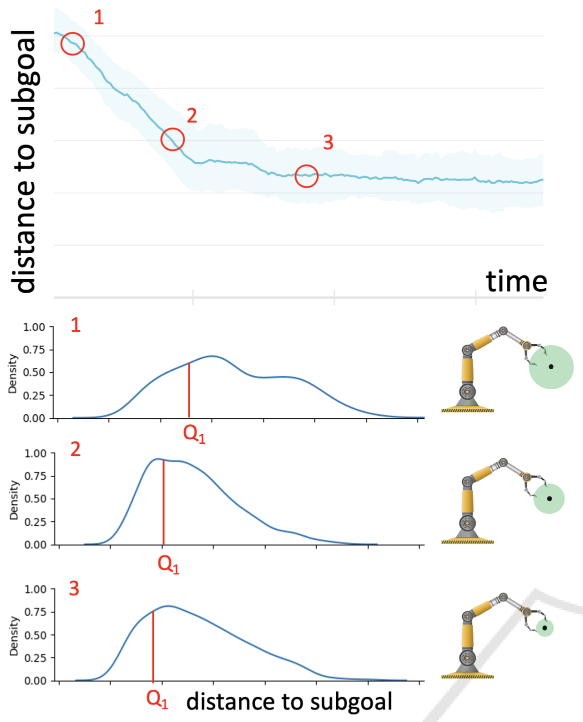

Figure 2: During the training, the average distance to the subgoal, i.e. subgoal error, decreases over time due to higher LL policy precision. In this symbolic diagram, black dots represent the subgoal position, while green circle represents SRD. We propose SRD defined as the function of recent distances to subgoals to consider current LL performance. As the training progresses, the distribution errors approach a distribution determined by the physical constraints of the manipulator.

about the problem domain.

4 ADAPTIVE SUBGOAL REQUIRED DISTANCE (ASRD)

Within our approach, SRD is designated online for a given fraction of recent final episode states to have got close enough to their goals. The method's primary goal is to aid the training robustness in finding a proper SRD and, as a result, increase higher final performance. The following sections call the proposed method adaptive subgoal required distance (ASRD).

Specifically, at the end of every $(l+1)$-level action we store the distances to the subgoal $g^l, l < L$ in a buffer B^l. We use a cyclic buffer of maximum size N. We define a new SRD, ε^l, based on the distances stored in the buffer B^l. Specifically, j-th element of the subgoal equals the quantile of order q of the j-th

```
Require : $s^l_t$, $g^l_t$
Ensure  : $l$-level episode ended at $t$
Store $(g^l_t - f^l(s^l_t))$ in a FIFO buffer $B^l$ of max
  size $N$;
if $size(B^l) = N$ then
    for $j := 1, \dots, \dim(G^l)$ do
        /* Calculate new SRD for each
           goal dimension                */
        $\varepsilon^l_j := Q_q\left(\{|d_j| : d \in B^l\}\right)$
    end
else
    /* If the buffer is not yet
       filled, use a fixed SRD          */
    $\varepsilon^l := \varepsilon^l_0$
end
```

Algorithm 1: ASRD algorithm, to be run after gathering the final state of the l-th level episode.

elements of the distances stored in the buffer B^l

$$\varepsilon^l_j = Q_q\left(\{|d_j| : d \in B^l\}\right) \tag{2}$$

where Q_q denotes the quantile of order q of its argument. At the beginning of the training, when there is not enough data in the buffer to accurately determine the quantile, we use a fixed SRD ε^l_0. We present our ASRD method in Algorithm 1.

It is worth noting that the higher dimensional the subgoal, the lower the chances of successfully reaching it, which has already been shown in (Gehring et al., 2021). The overall likelihood of achieving a subgoal results from the conjunction of probabilities of achieving the subgoal criteria per element.

5 EMPIRICAL RESULTS

Our experiments aim to evaluate the effect of the proposed mechanism of ASRD in GCHRL methods with sparse reward in terms of SRD robustness. Specifically, we compare the performance of baseline GCHRL methods with their equivalent enhanced with ASRD.

In particular, we verify the following hypothesis in this section:

- H0: performance of HRL methods is highly sensitive to environment goal required distance.
- H1: the proposed method of varying SRD aids the robustness of the training process of GCHRL with sparse rewards.
- H2: adjustable SRD allows obtaining higher control precision faster.

5.1 Experimental Setup

We evaluate our method for two goal-conditioned hierarchical algorithms, namely HAC (Levy et al., 2017) with fixed-length subgoals and HiTS (Gehring et al., 2021) with timed subgoals, both using hindsight experience replay (Andrychowicz et al., 2017). These two methods are used because both use sparse rewards on every hierarchy level. We assess the effectiveness of our method using the same hyperparameters for HAC and HiTS as those in (Gehring et al., 2021), without making any changes. Results are reported across 4 different environments (shown in Figure 3) where spatial and timed precision of subgoal reachability is crucial in order to achieve high performance:

1. *Drawbridge:* The agent controls a ship travelling on a straight river. Its goal is to cross the drawbridge the moment it opens. Therefore, the agent must synchronise its actions with the position of the drawbridge and accelerate early enough to pass it without hitting it and losing all of the momentum. It is impossible for the agent to actively slow down the ship – this makes this simple problem challenging as it requires the agent to learn to wait.
2. *Pendulum:* This environment is based on the classic problem in control theory and consists of a pendulum attached at one end to a fixed point, and the other end being free. The agent's goal is to apply torque on the free end to swing it into an upright position, each time starting with a random angle and velocity. Despite being conceptually simple, this problem requires a sequence of properly timed swings to finish in the desired position.
3. *Platforms:* In this environment, the agent starts on the lower level and must get to the upper level using a pair of moving platforms. The first platform moves up and down independently of the agent's actions, while the second platform is only activated when the agent steps on a special button. In this challenging scenario, the agent must learn to synchronize the activation of the second platform with the position of the first, and then use both to get to the higher level.
4. *UR5Reacher:* The agent's task is to move a robotic arm, which it controls by rotating its joints, to a specified location. Because the joints affect each other's position, it must learn and synchronise these interdependencies. Decomposing this problem into a sequence of sensible subgoals can solve it much faster than using flat RL.

Our method considers distances to subgoals over the most recent 1000 HL actions. It calculates a predefined quantile of them to use as an SRD in the following HL action. In the following experiments, we use three different quantiles of orders 0.1, 0.25, and 0.5 for adapting SRD with different target goal achievability. It is worth noting that the proposed method reduces the burden of tuning every element of subgoal achievement criteria (which may be as big as state space) separately. Instead, using the proposed method one can replace this tunning with a grid search for one scalar (quantile) that works best.

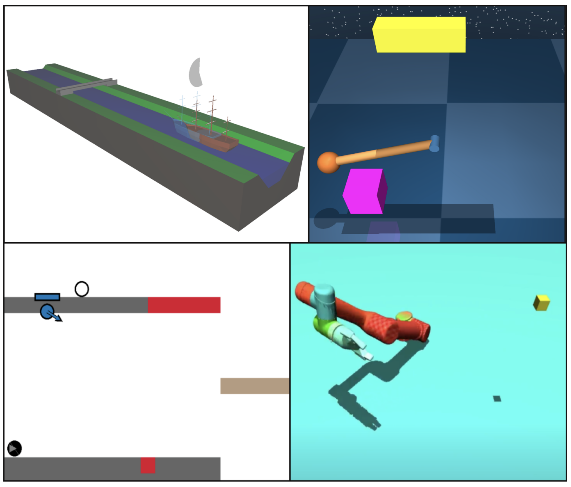
Figure 3: The environments used in our experiments. From the left: *Drawbridge*, *Pendulum*, *Platforms* and *UR5Reacher*.

5.2 H0: Sensitivity to Goal Required Distance

We recognize that the task of achieving sparse reward heavily depends on the environmental goal required distance. In particular, the task becomes harder if the environmental goal requires higher precision. Environmental requirements might not be fulfilled if the agent does not adjust subgoals precision to learn fine-grained control. This issue is depicted in Figure 4, which illustrates how the need for greater precision affects the agent's ability to achieve the environmental goal.

In our observations, we found that when we increased the necessary precision by 4 and 10 times, both algorithms were affected, but HAC showed a greater sensitivity. Furthermore, when comparing vanilla HAC and HiTS to cases where the ASRD mechanism was used, we noticed that the former two algorithms took longer to reach their final performance. Notably, our method only slightly delayed convergence in scenarios where environmental precision remained unaltered, but it enhanced performance in cases with disturbances.

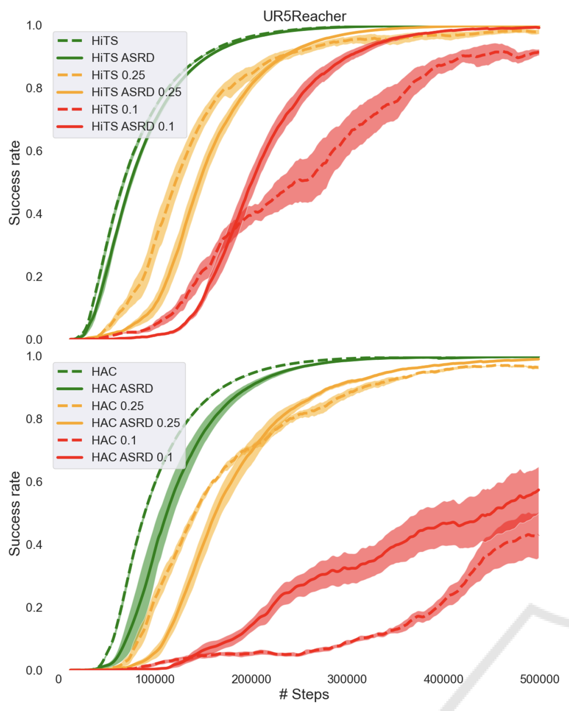

Figure 4: Sensitivity to environmental goal required distance on UR5Reacher environment for HAC and HiTS with and without ASRD. In the legend, numbers describe the coefficient by which we scaled the environmental goal required distance. The smaller the number, the harder the task of reaching the environmental reward.

5.3 H1: Robustness of GCHRL Training to Varying SRD

To test the sensitivity of the considered HRL methods to the SRD parameter, we disturb its value fine-tuned by their authors. Specifically, we scale them with a predefined factor called the SRD multiplier. We examine the agents' performance with and without ASRD in this setting. In particular, we evaluate agents' performance based on the average success rate from the final training performance, i.e. last 10% of time steps, to reduce the noise in the results.

We show the aggregated results obtained by the base HAC and HiTS algorithms with constant SRD and with SRD adapted by our method in Table 1 for better clarity. The results of methods enhanced with ASRD are the same or higher for almost all environment/algorithm/SRD multiplier combinations for at least one quantile. The performance of agents in the undisturbed scenario, i.e. where SRD Multiplier is 1, our method yields a boost in performance for every environment/algorithm combination except for HiTS in the Platforms environment.

The SRD Multiplier has a greater impact on the final outcome of the HiTS algorithm than the tested ASRD variations on Pendulum, Platforms, and UR5Reacher environments. On the other hand, the HAC algorithm is more affected by the initial SRD on Drawbridge and UR5Reacher. HAC achieves a low average success rate on the Platforms environment regardless of SRD. However, when using SRD adaptation with $q = 0.25$ and $q = 0.5$, it consistently achieves a higher success rate compared to the base algorithm.

For all environments, HAC with SRD adaptation obtains good results when using $q = 0.25$, although for Drawbridge, it obtains even better results using $q = 0.1$ and for UR5Reacher using $q = 0.5$. HiTS with SRD adaptation obtains good results using any q value, however, the best q value varies between environments. The best results on HiTS with SRD adaptation obtains using $q = 0.25$ for Drawbridge, $q = 0.1$ for Pendulum, $q = 0.5$ for Platforms, and any q for UR5Reacher.

Our method helps address the issue of varying levels of achieved testing transitions by defining SRD based on recent data. The level of testing transitions success is determined by the quantile and the exploration of the lower level, which in our experiments forces it to stay low along the training. As a result, our method improves the training stability of HiTS and HAC. However, it should be noted that the quantile order hyperparameter should be tuned per environment because there is no clear indication of one quantile that performs best across all environments. Also, due to the inherent noisiness of sparse reward HRL, the reported standard deviations indicate a significant discrepancy in performance across runs. Overall, adaptive SRD results in faster algorithm convergence to optimal policy in HiTS and a boost in the performance of HAC, which without our method cannot solve the Drawbridge environment.

5.4 H2: Adaptive SRD Convergence

In this section, we closely examine the impact of ASRD on agents' precision, defined as a distance to the subgoal, i.e. subgoal error, at the end of the HL action. We also explore the implications of using different length time windows in ASRD. Given the current capabilities of the lowest level, we would assume that adaptive SRD should result in sufficiently demanding but still realistic subgoals.

To calculate the distance to the subgoal, we store the vectors representing the final state of the agent and the subgoal state for each HL action. We then subtract the state vector from the subgoal vector and calculate the l_2 norm of this difference. By splitting the data acquired in this way into 3 equal-sized parts from different training phases, we observe how precise our agent was in reaching subgoals and how its capability

Table 1: Comparison of final average success rates for different environments, algorithms, adaptive SRD quantile orders, and initial SRD value multipliers. Q order of '-' denotes the base algorithm with constant SRD. The reported averages and standard deviations are calculated using five different seeds.

			SRD Multiplier				
Env	**Method**	**Q order**	0.25	0.5	1	2	4
Drawbridge	**HAC**	-	97.5±1.7	60.1±30.4	10.8±6.7	0.1±0.0	0.4±1.0
		0.1th	94.9±7.6	88.7±18.7	98.5±1.8	94.3±9.1	92.6±8.0
		0.25th	66.9±22.4	65.8±27.4	78.2±21.6	76.6±28.7	65.4±5.2
		0.5th	0.7±0.1	4.5±4.8	2.4±3.9	9.9±9.7	19.9±27.3
	HiTS	-	99.9±0.0	100.0±0.0	99.6±0.7	99.8±0.1	100.0±0.0
		0.1th	99.9±0.1	99.9±0.1	99.9±0.1	99.9±0.1	87.7±28.7
		0.25th	99.8±0.3	99.8±0.2	99.9±0.0	99.9±0.2	96.6±6.7
		0.5th	99.9±0.1	99.8±0.3	99.9±0.0	99.9±0.1	83.6±32.7
Pendulum	**HAC**	-	47.3±33.3	73.3±11.5	66.0±15.0	77.1±13.6	64.9±23.9
		0.1th	41.8±30.5	50.8±36.1	58.6±27.0	81.1±15.8	87.2±4.7
		0.25th	34.1±31.4	51.1±33.8	80.0±7.1	85.3±5.6	84.0±7.5
		0.5th	18.5±21.7	42.6±36.2	57.4±27.6	75.1±8.2	82.8±5.1
	HiTS	-	81.2±10.9	91.7±0.4	90.2±2.0	84.6±7.9	70.8±7.3
		0.1th	92.0±4.1	89.8±6.9	91.1±3.6	89.9±2.4	86.9±5.5
		0.25th	91.2±3.2	75.2±18.5	89.3±5.7	86.9±4.2	83.5±4.6
		0.5th	87.8±1.5	79.4±7.9	85.4±8.1	82.6±3.1	82.3±5.5
Platforms	**HAC**	-	37.0±36.3	27.6±17.5	43.5±29.2	6.0±8.5	0.3±0.6
		0.1th	54.3±39.7	2.4±4.6	4.1±6.8	4.3±6.5	14.5±16.4
		0.25th	49.7±37.3	37.8±38.0	45.1±39.9	44.4±39.3	35.7±34.1
		0.5th	39.6±3.1	55.9±21.8	35.1±18.2	27.4±26.9	48.0±23.4
	HiTS	-	66.8±46.4	75.8±24.2	85.7±35.0	66.7±47.1	14.3±35.0
		0.1th	79.0±25.9	82.0±19.0	67.4±34.5	69.9±29.4	64.2±35.5
		0.25th	80.7±31.5	64.4±43.2	61.6±41.1	60.5±41.3	79.5±33.0
		0.5th	86.9±23.4	82.2±22.7	75.7±22.8	57.2±38.2	93.9±4.6
UR5Reacher	**HAC**	-	99.9±0.0	99.9±0.1	99.8±0.1	99.5±0.7	90.1±23.8
		0.1th	99.8±0.1	99.9±0.0	99.7±0.2	98.9±0.9	98.7±0.7
		0.25th	99.9±0.1	99.9±0.0	99.8±0.1	99.5±0.2	98.8±1.1
		0.5th	99.9±0.0	99.9±0.1	99.8±0.1	99.4±0.4	99.5±0.3
	HiTS	-	100.0±0.0	100.0±0.0	100.0±0.0	99.1±1.0	93.1±3.0
		0.1th	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0
		0.25th	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0
		0.5th	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0

evolved during the training.

Indeed, the effect of squashed subgoal error distribution across training can be observed in Figure 5. It shows that, from the beginning of the training, our method makes the agent more focused on finishing closer to the assigned subgoal. Because, during the training, the LL performance increases and HL learns what the lower level is currently capable of, subgoals are getting gradually more reachable in terms of distance as the training progresses.

We analyzed how the window length influence the adaptive SRD method by comparing the average success rates of HAC+ASRD and HiTS+ASRD for different window lengths, namely 50, 500, and 1000. In Table 2 we report the results of this experiment for each environment.

Clearly, HiTS method is less susceptible to window length changes in terms of final average performance than HAC. However, HiTS and HAC exhibit the highest variance in results in the Platforms environment, which is the hardest among tested environments. There is also a considerable difference between HAC performance in Drawbridge and Pendulum environments which indicates higher dependence on window length for HAC. The highest results for HiTS+ASRD are obtained using shorter window lengths, while there is no clear relation between window length and performance for HAC.

It is important to not that HAC and HiTS differ in how they determine the length of their high-level (HL) actions. HAC uses fixed times, whereas HiTS learns the optimal timing for each HL action. As a result, the length of HL actions may vary greatly between these two algorithms. This difference in action length can cause a delay between the current low-level (LL) capabilities and the subgoal distances recorded in the distances to the subgoal replay buffer. In fact, this is particularly noticeable in the HiTS algorithm, where

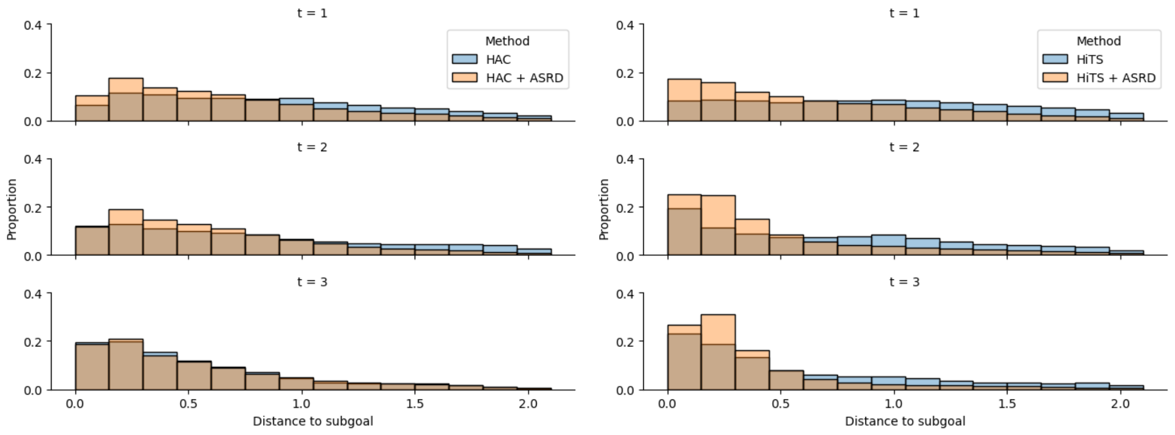

Figure 5: Normalized histograms of the distances to the subgoals throughout training on the *Platforms-v1* environment. In both cases (*left: vs. HAC*, *right: vs. HiTS*), our method leads to distributions with smaller mean and variance. The distribution change across training is evident for the HiTS+ASRD method where subgoal error forms distribution with a significantly thinner tail than HiTS.

using a smaller window length for learning HL actions leads to better performance in every environment.

Table 2: Influence of window length on ASRD performance for HiTS and HAC algorithm. The final success rate's reported averages and standard deviations are calculated using multiple seeds.

		Window length		
Env	**Method**	50	500	1000
Drawbridge	**HAC**	88.9± 15.7	87.7 ± 17.3	98.5 ± 1.8
	HiTS	99.9± 0.1	98.5± 1	99.9 ± 0.1
Pendulum	**HAC**	87.0 ± 3.0	76.0 ± 16.5	80.0 ± 7.1
	HiTS	91.4 ± 2.6	90.3 ± 2.8	91.1± 3.6
Platforms	**HAC**	21.0 ± 29.8	50.6 ± 14.7	35.1 ±18.2
	HiTS	99.3 ± 0.2	64.0 ±45.3	75.7±22.8
UR5Reacher	**HAC**	99.8 ± 0.1	99.5± 0.2	99.8±0.1
	HiTS	100 ± 0	100 ± 0	100 ± 0

6 DISCUSSION AND FURTHER WORK

We consider the ASRD a step forward in formulating suitable subgoals in the control task for GCHRL, especially in a sparse reward setting. The proposed method leads to higher results in 7 out of 8 environment/algorithm combinations while using the original SRD proposed by HAC and HiTS authors. While disturbing the SRD by the predefined multiplier, our method significantly boosts performance in most cases. However, our method's main limitation is that no single quantile order or window length universally works for all environments. Further research is needed to designate these parameters without trial-and-error tuning.

We speculate that environment dimensionality and, as a consequence, subgoal dimensionality are critical factors in determining the optimal quantile order for ASDR. This is supported by the monotonic tendency for performance increase/decrease in Table 1 with respect to the quantile order.

The relation between window length and performance is clear for HiTS. The window length should be short to consider only the most recent HL actions and LL capabilities. However, there is no clear prescription for window length in HAC, as it appears to be more environment-specific. Further research is needed to understand this difference.

One promising avenue for future research involves the development of adaptive masks for state vectors. These masks would identify critical state coordinates that could be used as subgoals, with varying levels of importance assigned to each component. This would enable the subgoal vector to be composed of coordinates with highly disparate levels of importance for successful task completion without sacrificing the agent's performance. Additionally, the proposed method could be extended to scenarios where each coordinate of the subgoal vector has its quantile order, allowing less important elements to use higher quantiles. This way, only key subgoal elements would determine the reachability bottleneck.

7 CONCLUSIONS

Our work addresses a significant limitation in GCHRL algorithms by proposing a novel method that adapts the subgoal range dynamically based on the agent's recent performance. We show that fixed subgoal ranges can lead to either too narrow or imprecise subgoals, which hinder the learning process. Our adaptive approach improves training robustness and method performance while simplifying the hyperparameter tuning procedure. Our findings also suggest that our method leads to higher control precision and faster convergence in most cases. Overall, our work contributes to the ongoing effort to improve HRL methods, especially in subgoal reachability issues in sparse reward GCHRL.

ACKNOWLEDGEMENTS

This research was partially funded by the Warsaw University of Technology within the Excellence Initiative: Research University (IDUB) programme LAB-TECH of Excellence (grant no. 504/04496/1032/45.010013), research project "Bio-inspired artificial neural network" (grant no. POIR.04.04.00-00-14DE/18-00) within the Team-Net program of the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund, and National Science Centre, Poland (grant no 2020/39/B/ST6/01511). This research was supported in part by PLGrid Infrastructure.

REFERENCES

Adolph, K. E., Cole, W. G., Komati, M., Garciaguirre, J. S., Badaly, D., Lingeman, J. M., Chan, G. L. Y., and Sotsky, R. B. How Do You Learn to Walk? Thousands of Steps and Dozens of Falls per Day. 23(11):1387–1394.

Andrychowicz, M., Crow, D., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W. (2017). Hindsight experience replay. *ArXiv*, abs/1707.01495.

Bagaria, A. and Konidaris, G. (2019). Option discovery using deep skill chaining. In *International Conference on Learning Representations (ICLR)*.

Barto, A. G. and Mahadevan, S. (2003). Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1):41–77.

Campos, V., Trott, A., Xiong, C., Socher, R., Giró-i Nieto, X., and Torres, J. (2020). Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International Conference on Machine Learning*, pages 1317–1327. PMLR.

Chane-Sane, E., Schmid, C., and Laptev, I. Goal-Conditioned Reinforcement Learning with Imagined Subgoals.

Colas, C., Karch, T., Sigaud, O., and Oudeyer, P.-Y. Autotelic Agents with Intrinsically Motivated Goal-Conditioned Reinforcement Learning: A Short Survey. 74.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. (2018). Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*.

Eysenbach, B., Salakhutdinov, R. R., and Levine, S. (2019). Search on the replay buffer: Bridging planning and reinforcement learning. *Advances in Neural Information Processing Systems*, 32.

family=Vries, given=Joery A., p. u., Moerland, T. M., and Plaat, A. On Credit Assignment in Hierarchical Reinforcement Learning.

Fournier, P., Sigaud, O., Chetouani, M., and Oudeyer, P.-Y. (2018). Accuracy-based Curriculum Learning in Deep Reinforcement Learning.

Gehring, J., Synnaeve, G., Krause, A., and Usunier, N. (2021). Hierarchical skills for efficient exploration. *Advances in Neural Information Processing Systems*, 34:11553–11564.

Ghosh, D., Gupta, A., Reddy, A., Fu, J., Devin, C., Eysenbach, B., and Levine, S. (2019). Learning to reach goals via iterated supervised learning. *arXiv preprint arXiv:1912.06088*.

Gürtler, N., Büchler, D., and Martius, G. (2021). Hierarchical reinforcement learning with timed subgoals. *Advances in Neural Information Processing Systems*, 34:21732–21743.

Gürtler, N., Büchler, D., and Martius, G. Hierarchical Reinforcement Learning with Timed Subgoals.

Jiao, Y. and Tsuruoka, Y. HiRL: Dealing with Non-stationarity in Hierarchical Reinforcement Learning via High-level Relearning.

Lee, S., Kim, J., Jang, I., and Kim, H. J. (2022). Dhrl: A graph-based approach for long-horizon and sparse hierarchical reinforcement learning. *Neural Information Processing Systems*.

Levy, A., Konidaris, G., Platt, R., and Saenko, K. (2017). Learning multi-level hierarchies with hindsight. arXiv:1712.00948.

Li, S., Zhang, J., Wang, J., Yu, Y., and Zhang, C. (2021). Active hierarchical exploration with stable subgoal representation learning. In *International Conference on Learning Representations*.

Liu, M., Zhu, M., and Zhang, W. (2022). Goal-Conditioned Reinforcement Learning: Problems and Solutions.

McGovern, A. and Barto, A. G. (2001). Automatic discovery of subgoals in reinforcement learning using diverse density.

Nachum, O., Gu, S. S., Lee, H., and Levine, S. (2018). Data-efficient hierarchical reinforcement learning. In *Neural information processing systems (NeurIPS)*, volume 31.

Nachum, O., Tang, H., Lu, X., Gu, S., Lee, H., and Levine, S. (2019). Why does hierarchy (sometimes) work so well in reinforcement learning? *arXiv preprint arXiv: Arxiv-1909.10618*.

Pateria, S., Subagdja, B., Tan, A.-h., and Quek, C. Hierarchical Reinforcement Learning: A Comprehensive Survey. 54(5):1–35.

Portelas, R., Colas, C., Weng, L., Hofmann, K., and Oudeyer, P.-Y. (2020). Automatic Curriculum Learning For Deep RL: A Short Survey.

Precup, D. (2000). *Temporal abstraction in reinforcement learning*. University of Massachusetts Amherst.

Schmidhuber, J. (1991). Learning to generate sub-goals for action sequences. In *Artificial neural networks*, pages 967–972.

Shankar, T. and Gupta, A. (2020). Learning robot skills with temporal variational inference. In *International Conference on Machine Learning*, pages 8624–8633. PMLR.

Sharma, A., Gu, S., Levine, S., Kumar, V., and Hausman, K. (2019). Dynamics-aware unsupervised discovery of skills. arXiv:1907.01657.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction. Second edition*. The MIT Press.

Sutton, R. S., Precup, D., and Singh, S. (1999). Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211.

Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., and Kavukcuoglu, K. (2017). Feudal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, pages 3540–3549. PMLR.

Zhang, T., Guo, S., Tan, T., Hu, X., and Chen, F. (2020a). Generating adjacency-constrained subgoals in hierarchical reinforcement learning. *Advances in Neural Information Processing Systems*, 33:21579–21590.

Zhang, Y., Abbeel, P., and Pinto, L. (2020b). Automatic Curriculum Learning through Value Disagreement.