Zero-Incentive Dynamics: a look at reward sparsity through the lens of unrewarded subgoals

Yannick Molinghen[1], **Tom Lenaerts**[1,2,3]
yannick.molinghen@ulb.be, tom.lenaerts@ulb.be

[1]**Machine Learning Group, Université Libre de Bruxelles, Belgium**
[2]**AI Lab, Vrije Universiteit Brussel, Belgium**
[3]**Center for Human-compatible AI, UC Berkley, USA**

Abstract

We re-examine the commonly held assumption that the frequency of rewards is a reliable measure of task difficulty in reinforcement learning and show that it is not always true. Then, we identify and formalize a structural challenge that undermines the effectiveness of current policy learning methods: when the completion of mandatory subgoals does not directly yield rewards. We characterize such settings as exhibiting Zero-Incentive Dynamics, where transitions critical to success remain unrewarded. We show that state-of-the-art deep subgoal-based algorithms fail to overcome this challenge and that learning efficiency is highly sensitive to the temporal proximity between subgoal completion and eventual reward. Our findings reveal a fundamental limitation in current approaches and point to the need for mechanisms that can infer latent task structure without relying on immediate incentives.

1 Introduction

In reinforcement learning (Sutton & Barto, 2018, RL), agents learn an optimal policy by maximizing the expected cumulative reward through interactions with the environment. In cases where there is little-to-no feedback until the completion of the task, reward is said to be sparse. In such cases, learning becomes significantly more difficult due to the sparsity (even absence) of informative gradients for policy improvement (Ocana et al., 2023). This challenge has motivated substantial research into mitigation techniques such as reward shaping (Y. Ng et al., 1999), intrinsic motivation (Pathak et al., 2017; Burda et al., 2018), and hierarchical RL (Sutton et al., 1999).

While these approaches aim to compensate for sparse signals by injecting auxiliary rewards or by introducing structure, they often treat sparsity as a scalar property of the reward function, namely the frequency or density of non-zero rewards. However, this view neglects the structure of the problem and the location of the reward with regard to subgoals in the tasks.

In this work, we aim to shed light on why reward sparsity should not be considered to be the sole metric of a problem difficulty and look at reward sparsity through the lens of subtasks whose accomplishment is unrewarded. Two recent works (Ocana et al., 2023; Molinghen et al., 2025) respectively introduced Partially Ordered Subtasks and Zero-Incentive Dynamics (ZID) intuitively to refer to essential transitions that are not directly reinforced despite being necessary for the task success.

We begin by empirically showing in Section 3 that reward sparsity alone is an insufficient indicator of task difficulty. Specifically, we construct a series of environments with increasing reward density where exploration actually becomes harder, demonstrating that the distribution and alignment of rewards with subtask transitions matters more than their global frequency in that example. Then, we formalize in Section 4.1 the notion of ZID as a structural property of the underlying Markov Decision

Process (Bellman, 1957, MDP) characterized by unrewarded but mandatory transitions that shape the state space in bottlenecks. Using a graph-theoretic perspective, we define ZID in terms of cut-sets over the transition graph of the MDP and distinguish it from general reward sparsity by focusing on the causal and temporal relationship between subgoal completion and eventual reward. Next, in Section 4.2 we provide evidence that current state-of-the-art subgoal-oriented methods (Sutton et al., 1999; Jeon et al., 2022; Xu et al., 2023) fail to exploit the presence of subgoals when they have ZID. Despite their strong performance in other benchmarks, these methods perform no better than other RL algorithms under ZID. Finally, we investigate in Section 4.3 the impact of reward delay on deep RL by introducing reward shaping with controlled delays between subgoal completion and reward delivery. We show that even when the reward density remains constant, decreasing this delay substantially improves the quality of the policy. This suggests that the proximity of rewards to subtask completions, rather than their mere presence, is critical for effective learning.

Our findings highlight a gap in the current RL paradigm: the lack of mechanisms to identify unrewarded subtasks. Addressing this gap may require new architectures or representations that can detect structural dependencies in the environment, even in the absence of immediate reward signals.

2 Related work

2.1 Markov Decision Processes

Markov Decision Process (Bellman, 1957, MDP) provide a model of sequential decision-making under uncertainty and are the main formalism used in Reinforcement Learning (Sutton & Barto, 2018, RL). An MDP M are defined by the tuple $M = \langle S, A, R, T \rangle$ where S is the set of state (state space), A is the set of actions (action space), $R : S \times A \times S \to \mathbb{R}$ is a reward function, and $T : S \times A \times S \to [0, 1]$ is the transition function that determines the probability of that in state s, taking action a results in state s'. In particular, we note $S_0 \subset S$ the set of initial states and $S_G \subset S$ the set of goal states.
Given an MDP $M = \langle S, A, R, T \rangle$, a directed weighted graph (Wilson, 2009) $G = (V, E, W)$ can be constructed to represent M, where $V = S$ is the set of vertices, $E = \{(s, a, s') \mid s, s' \in S,\ a \in A\}$ is the set of edges, and $W = R$ is the weight function that associates to each edge the reward for taking the corresponding transition.

2.2 Reward sparsity

In recent years, the concept of reward sparsity has regularly been discussed in the field of deep RL and has generally been presented as the source of the difficulty of the problem under study (Andrychowicz et al., 2017; Burda et al., 2018; Ladosz et al., 2022; Trott et al., 2019). However, only a handful of these studies define reward sparsity, let alone a justify why reward sparsity is problematic. Regardless, a wide variety of methods have been introduced to cope with reward sparsity. Meng (2024) splits them into three categories: reward shaping (or manual labelling), intrinsic motivation, and hierarchical RL.

Reward shaping is the most straightforward way to tackle reward sparsity and relies on the introduction of additional rewards based on domain knowledge. Randløv & Alstrøm (1998) and Amodei et al. (2016), among others, have illustrated how naive reward shaping can alter the definition of the task, resulting in a change to the optimal policy π^* and causing the agent's policy to no longer fulfil the task designer's intended objective. This is why, as early as 1999, Y. Ng et al. (1999) introduce Potential-Based Reward Shaping (PBRS) that modifies the reward function R as shown in Equation 1. PBRS relies on the definition of a potential function ϕ and ensures that the optimal policy π^* remains unchanged. However, even though PBRS guarantees the invariance of π^*, this technique relies on domain knowledge, is problem-specific and can therefore not readily be generalised to any task. Furthermore, PBRS can be extremely challenging to implement in practice, if not impossible due to a lack of domain-knowledge.

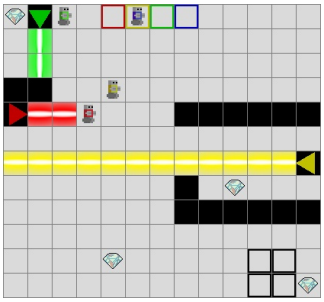

Figure 1: Laser Learning Environment. This map has four agents and three lasers that can be blocked. The red agent is currently blocking the red laser.

$$R'(s, a, s') = R(s, a, s') + \gamma\phi(s) - \phi(s') \tag{1}$$

Exploration bonuses aim at providing an extra reward to the agent according to the novelty of the state visited and can take various forms. In its most simplistic form, intrinsic curiosity is a statewise counter and the agent receives an extra reward that is annealed over the number of times this state has been visited (Tang et al., 2017). More complex approaches rely on the training of neural networks to predict an embedding of a state and use the prediction error as measure of novelty (Pathak et al., 2017; Burda et al., 2018). Although these methods have shown very good results in environments with sparse rewards, it it important to realise that they do not guarantee policy invariance and can therefore suffer the same problems as the ones discussed above, unless combined with recent techniques such as Potential-Based Intrinsic Motivation (Forbes et al., 2024).

Hierarchical RL is an approach whose objective is to identify intermediate states of interest that are referred to as subgoals (Sutton et al., 1999) in order to decompose a large task in smaller and more manageable ones. The literature distinguishes two categories of subgoal-oriented problems, a former where subgoals are explicitly disclosed by the environment (Ahilan & Dayan, 2019; Geng et al., 2024) and a latter where they are not (Xu et al., 2023; Jeon et al., 2022). Leveraging subgoals often involves hierarchical structures where a high-level agent, or meta-agent, trains a high-level policy π^h to select subgoals for a low-level policy π^l to achieve. This decomposition reduces the exploration space and enables the agent to focus on intermediate objectives, making it easier to solve long-term tasks, but also relies on the ability of the meta-agent to identify these subgoals.

2.3 Partially Ordered Subtasks

Ocana et al. (2023) identify nine features of the underlying MDP that negatively or positively impact deep RL under reward sparsity. Amongst others, the presence of *Partially Ordered Subtasks* (POS) is one of them. From a high-level point of view, the authors discuss that the presence of subtasks whose completion is not rewarded nor indicated by the environment constitutes an obstacle to deep RL. These subtasks are partially ordered in the sense that their completion can be performed in different orders but the agent has to complete them all to fulfil the overall task, which yields a positive feedback.

Ocana et al. (2023) discuss POS with the example of a problem where the agent has two subtasks to complete: learn to swim and learn to climb. The authors explain that even if exploration bonuses (discussed in Section 2.2) can have the agent learn how to swim initially, there is no guarantee that when the agent starts to learn to climb, it does not forget how to swim, which French (1999) refer to as catastrophic forgetting. Although the authors indeed discuss the theoretical implications of POS, they do not provide evidence of why their presence can negatively impact the learning process.

2.4 Laser Learning Environment

The Laser Learning Environment (Molinghen et al., 2025, LLE) illustrated in Figure 1 has recently been introduced as a challenging MARL environment with sparse rewards. LLE is a deterministic fully observable cooperative grid-world in which agents can collect gems and must reach the exit represented by the black squares. The cooperative dynamics of LLE revolve around laser-blocking: agents can block lasers of the same colour as theirs, allowing other agents to pass, but die if they walk into a beam of a different colour. As such, collecting a gem or exiting the game is rewarded by +1, while dying in a laser is punished by -1 and terminates the episode.

Molinghen et al. (2025) showed that state-of-the-art methods such as Value Decomposition Networks (Sunehag et al., 2018, VDN) or QMIX (Rashid et al., 2018) were unable to complete LLE and hypothesize that the fact that crossing lasers is not rewarded – which they refer to as Zero-Incentive Dynamics (ZID) – is the main cause of this failure but provided no evidence of their claim.

3 Reward sparsity

Although the methods mentioned in Section 2.2 have shown some success to mitigate reward sparsity in some tasks, we argue that there remain misconceptions about reward sparsity, notable as to whether or why it can be a challenge. In this section, we first provide a formal definition of reward sparsity and then show why reward sparsity alone is not a reliable indicator of the difficulty of a task.

3.1 Definitions

Intuitively, reward sparsity refers to the property of an MDP whose reward signal where most transitions yield the same base reward r_b, typically zero, and only a handful of transitions provide a reward superior to r_b, typically upon completing a goal or a task.

Definition 1 (Reward density)**.** *Let $M = \langle S, A, R, T \rangle$ be an MDP and $G_M = (V, E, W)$ be the directed weighted graph induced by M. Let E^+ be the set of edges with a meaningful reward, i.e. $E^+ = \{e \in E \mid W(e) > r_b\}$. The reward density of M is $\mathcal{D}_M = \frac{|E^+|}{|E|}$.*

Definition 2 (Reward sparsity)**.** *Let M be an MDP. M has sparse rewards if $0 < \mathcal{D}_M \ll 1$.*

As a result of definitions 1 and 2, a sparse MDP has a low reward density, and a dense MDP does not have sparse rewards. If these definitions do not provide a threshold from which a reward is sparse, they enable the comparison and ordering of MDPs with regard to reward sparsity.

3.2 Reward sparsity, an incomplete metric

We illustrate that reward sparsity is not a reliable indicator of the difficulty of a problem with the environment M shown in Figure 2. The underlying graph has 101 edges, 3 of which lead to the goal state in black and therefore yield a positive reward (details can be found in the Supplementary Materials E). As a result, $\mathcal{D}_M = \frac{3}{101} \approx 0.0297$.

Consider $M_1, \ldots, M_4$, four variations of M where M_n is the same MDP as M except that the transitions indicated by the arrows in Figure 2 whose label is $\leq n$ are disabled. For instance, M_2 corresponds to M with the arrows labelled with 1 and 2 disabled. Hence, n unrewarded edges are removed in M_n in comparison to M. Note that the state space of M_n remains identical to M and that $\mathcal{D}_{M_n} = \frac{3}{101-n}$, therefore $\mathcal{D}_{M_n} < \mathcal{D}_{M_{n+1}}$.

We randomly explore the state space for 200k steps with a maximal time horizon h of 12, 13 and 14 and record the exit rate at the end of the episode. We plot the mean exit rate according to the reward density in Figure 2 (left). We can see that the exit rate decreases when the reward sparsity increases, thereby illustrating that reward sparsity is not a reliable indicator of the difficulty of an MDP.

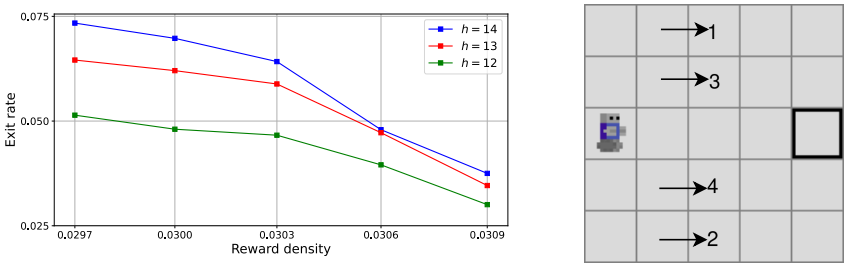

Figure 2: (Left) Average exit rate over with random exploration for 200k steps for different time horizons h. Counter-intuitively, the exit rate decreases when the reward density increases. (Right) The map on which the left plot is based on. The arrows labelled with $1, \ldots, 4$ indicate which state-actions are disabled for $M_1, \ldots, M_4$.

4 Zero-Incentive Dynamics

We argue that Ocana et al. (2023) and Molinghen et al. (2025) respectively describe the same phenomenon albeit with different terminology. The former use the term of Partially Ordered Subtasks (POS) to emphasize that there are subtasks from a high-level perspective, while the latter use the term of Zero-Incentive Dynamics (ZID) to emphasize the absence of reward for achieving these subtasks. In this work, we bring these two concepts together and use the term "ZID", arguing that the question of the reward is central, as we show further in this section.

In order to provide further insight into why ZID are challenging to deep RL, we first formally define ZIDs (equivalently POSs) from a graph-theoretical perspective, show the inability of state-of-the-art methods to cope with problems with ZID, and provide experimental evidence of the importance of rewarding subtasks shortly after their completion.

4.1 Definitions

Molinghen et al. (2025) refer to the fact that agents must collaborate in LLE to block lasers and allow other agents to pass as *agent interdependence* and argue that agent interdependence creates State Space Bottlenecks (SSB), although no formal definition is provided.

Intuitively, in an MDP with a mandatory subtask, an SSB is the set of state-action pairs that directly lead to the accomplishment of this subtask. Formally, we define an SSB in the context of a graph induced by an MDP. For the sake of conciseness, we provide definitions of the concepts of graph theory required for Definition 3 in Appendix A. Additionally, an extension to continuous state and action spaces is given in Appendix B.

Definition 3 (State Space Bottleneck)**.** *Let $M = \langle S, A, R, T \rangle$ be an MDP and $G_M = (V, E, W)$ be the directed weighted graph induced by M. Let S_0 be the set of initial states and S_G be the set of goal states.*
A state space bottleneck $\mathcal{B}$ is a minimum directed S_0-S_G cut-set of G_M.

We proceed to build on top of SSBs to propose a formal definition of ZID. Intuitively, an SSB has ZID if completing the according subtask is not rewarded.

Definition 4 (Zero-Incentive Dynamics)**.** *Let M be an MDP and $G_M = (V, E, W)$ be the directed weighted graph induced by M. Let $\mathcal{B}$ be a state space bottleneck of M. Let r_b be the base reward in M.*
$\mathcal{B}$ has zero-incentive dynamics if $\forall e \in \mathcal{B},\ W(e) \leq r_b$.

In some cases, it could be the case that the accomplishment of a subtask is not directly rewarded, for instance because a human has to validate the accomplishment or because of network latency. In those cases, one would rather talk about *Delayed Incentive Dynamics* with some delay d.

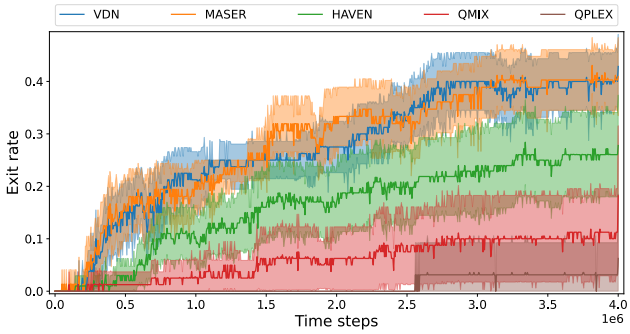

Figure 3: Average exit rate over the course of the training for subgoal-oriented methods on the map shown in Figure 1. Results are shown with 95% confidence intervals and averaged over 16 different seeds. The maximal achievable exit rate is 1.

4.2 Subgoal-oriented RL under ZID

There exist a wide literature on subgoal-oriented RL, and the question of whether state-of-the-art methods in subgoal-oriented RL are able to deal with ZID therefore naturally arises. As such, we evaluate MASER (Jeon et al., 2022) and HAVEN (Xu et al., 2023), two state-of-the-art methods that aim at leveraging the presence of subgoals in the environment and that have shown to outperform other algorithms such as VDN (Sunehag et al., 2018) and QMIX (Rashid et al., 2018) in the StarCraft Multi-Agent Challenge (Samvelyan et al., 2019). Internally, MASER uses the expected return at both the individual and at the collective level as a metric to identify subgoals and design an intrinsic reward signal to encourage transitions that reach the identified subgoals. HAVEN uses a hierarchical structure (see Equation 2.2) where the meta-agent uses its own policy to identify subgoals and assign them to the agents.

We train VDN, QMIX, QPLEX (Wang et al., 2021), HAVEN and MASER agents (see Appendix C for the full methodology) on the environment illustrated in Figure 1 for which Molinghen et al. (2025) have already shown that VDN and QMIX were unable to complete it. We plot the mean exit rate over the course of the training in Figure 3 and observe that neither HAVEN nor MASER are able to outperform VDN in this setup, thereby showing the inability of these methods to identify – or at least leverage the presence of – subgoals in the environment. We hypothesize that due to their internal working principles, HAVEN and MASER both implicitly require subgoals to be rewarded to exploit their presence and are therefore unable to outperform other algorithms under ZID. We further motivate this hypothesis in Appendix C by showing that removing ZID enables these methods to complete the task seamlessly.

4.3 Impact of the SSB-to-reward distance

To further understand the impact of ZID on deep RL, let us consider the environment shown in Figure 4 (right) where the two lasers are SSBs with ZID. To show the effects of ZID on deep RL, we design a slightly different version of this environment in which, once per episode, a collective reward is received the first time each agent crosses each laser. To ensure that the optimal policy π^* remains unchanged, we perform this reward shaping with PBRS as detailed in Appendix D.

We train agents with VDN and analyse the exit rate over the course of the training. Our full methodology can be found in Appendix D. In Figure 4 the curve labelled "No shaping" (brown) is the exit rate for the initial problem with ZID, and the curve labelled "$d = 0$" (blue) is the exit rate for the shaped version. The plot shows the blatant effects of ZID as the agents in the shaped version (blue) learn a good policy significantly faster than the others (brown).

We further investigate the effect of ZID by introducing a delay of d steps between the accomplishment of a subtask and the collection of the corresponding reward. Our results in Figure 4 clearly

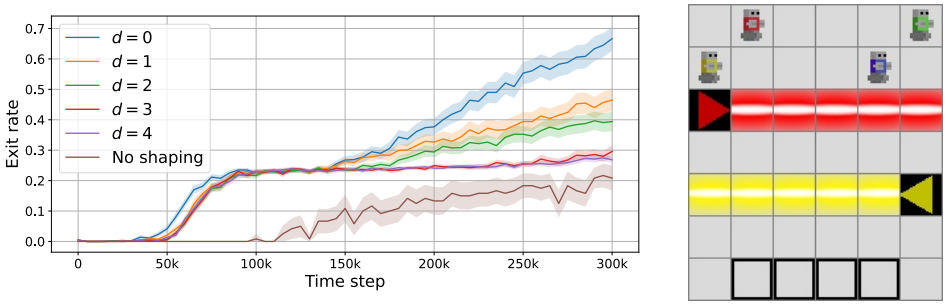

Figure 4: (Left) Exit rate over training time step for shaped reward delay $d = 0, 1, \ldots, 4$ on the environment shown on the right. Results are averaged on 30 different seeds and shown with 95% confidence intervals. (Right) Environment used to analyse nuances of ZID. The agents randomly spawn in the two top rows and must reach the exits represented by the tiles with a black frame.

show that the closer the reward is to the accomplishment of the subtask, the more efficient the learning.

Hence, we argue that the delay between the accomplishment of a subtask and the collection of the according reward is a key factor to the learning of deep RL policies. This result also further disqualifies reward density as an indicator of the problem difficulty since the reward density is identical for $d = 0, \ldots, 4$ but the policy is different.

5 Discussion

On the one hand, the negative results of Section 4.2 show that subtasks with ZID remain a challenge to state-of-the-art deep RL methods to this day. On the other hand, the positive results of Section 4.3 suggest that if agents were able to automatically identify the accomplishment of subtasks, even a few steps late, it would be readily possible to mitigate or even overcome ZID.

Sunel et al. (2024) discuss three categories of automatic subgoal identification methods: graph-based approaches that build a (sub-)graph of the environment and use algorithms on graphs to identify useful edges or vertices; statistics-based approaches that compute metrics on states or transitions; and Multiple Instance Learning (MIL) based approaches that classify episodes as either positive (task success) or negative (task failure) and typically identify common features among positive instances. We readily disqualify graph-based approaches such as the one of Şimşek et al. (2005) on the grounds that their ability to identify subgoals scales poorly with the size of the state space as illustrated in Appendix F. MIL methods (McGovern & Barto, 2001) are not readily applicable to our problem because they rely on the comparison between positive (success) and negative (failure) trajectories. As shown in Section 4.2, SOTA algorithms are unable to complete ZID problems, which makes it impossible to gather positive instances and apply MIL methods. Statistics-based approaches include a wide variety of methods that assign a metric to each state and leverage that metric to identify subgoals. MASER and HAVEN are examples of such methods and have shown to be ineffective.

6 Conclusion

In this paper, we have first shown that reward sparsity is a topic more complex than it looks. Specifically, we showed that increasing the reward density of a task could counter-intuitively lower its success rate, thereby showing that reward sparsity is not a reliable indicator of the difficulty of a problem. Then, we specifically focused on Zero-Incentive Dynamics, a particular case of reward sparsity where the accomplishment of subgoal is not rewarded. We defined the concept formally, and then showed that state-of-the-art subgoal-oriented methods are unable to identify subgoals under Zero-Incentive Dynamics and therefore performed as poorly as more general-purpose deep RL

methods. To give more insights on the importance of rewarding subtasks, we showed that the distance between the achievement of a subtask and the collection of the corresponding reward is a key factor to the quality of the learned policy, thereby raising the question of whether subtasks could be discovered automatically. We discuss this possibility and argue that there exist no such method to this day.

Together, our results shed light on a specific kind of reward sparsity, Zero-Incentive Dynamics, and show that there exist no counter-measure to this day, thereby indicating a path for future research.

A Graphs and cuts

The following definitions of graph theory (Wilson, 2009) are used to formalize the concepts of State Space Bottleneck and Zero-Incentive Dynamics in Section 4.1.

Definition 5 (Directed cut)**.** *Let $G = (V, E)$ be a directed graph. A directed cut $\langle S, T\rangle$ (or dicut) of G is a partition of V into two disjoint sets S and T, such that each edge is directed from S to T.*

Definition 6 (Cut-set)**.** *A cut $\langle S, T\rangle$ defines a cut-set C, the set of edges that have one endpoint in S and one endpoint in T.*

Definition 7 (Minimum cut)**.** *A cut is minimum if the size of the cut-set is not larger than the size of any other cut-set.*

Definition 8 (S-T cut)**.** *Let $G = (V, E)$ be a graph.*
Let S and T be two disjoint subsets of V, i.e. $S \subset V$, $T \subset V$ and $S \cap T = \emptyset$.

An S-T cut is a cut $\langle A, B\rangle$ of G such that $\forall s \in S, s \in A$ and $\forall t \in T, t \in B$.

Definition 9 (Winning walk)**.** *A winning walk is a finite sequence of pairwise adjacent edges $e_1, e_2, \ldots, e_n$ such that $e_1 = \langle s_0, a, s'\rangle$ and $e_n = \langle s, a, s_g\rangle$ with $s_0 \in S_0$ and $s_g \in S_G$.*

B Continuous SSB and ZID

In this section, we extend Definition 3 and Definition 4 to the continuous case. In such setting, set of states become regions in S and set of edges become regions in $S \times A \times S$.

Definition 10 (Continuous SSB)**.** *Let $M = \langle S, A, R, T\rangle$ be an MDP where $S \subset \mathbb{R}^n$, $A \subset \mathbb{R}^m$, $R : S \times A \times S \to \mathbb{R}$ and $T : S \times A \times S \to [0, 1]$. Let $S_0 \subset S$ be the starting region and $S_G \subset S$ be the goal region.*
A state space bottleneck $\mathcal{B} \subset S \times A \times S$ is the minimal region such that for any winning trajectory $\tau = (s_0, a_0, s_1, a_1, \ldots, s_{t-1}, a_{t-1}, s_t)$ where $s_0 \in S_0$ and $s_t \in S_G$, $\exists\, i \in \{0, 1, \ldots, t-1\} \mid (s_i, a_i, s_{i+1}) \in \mathcal{B}$.

Definition 11 (Continuous ZID)**.** *Let $M = \langle S, A, R, T\rangle$ be an MDP with continuous state and actions spaces. Let $\mathcal{B}$ be a state space bottleneck of M. Let r_b be the base reward in M.*
$\mathcal{B}$ has Zero-Incentive Dynamics if $\forall (s, a, s') \in \mathcal{B}, R(s, a, s') \leq r_b$.

C Subgoal-oriented approaches methodology

We train the agents with VDN, QMIX, QPLEX, HAVEN and MASER on the map shown in Figure 1 for 4 million time steps with double deep Q-learning (Van Hasselt et al., 2016), an ϵ-greedy exploration policy where ϵ is annealed from 1 down to 0.05 over 50k time steps, and set a maximal episode time limit to $\lfloor \frac{width \times height}{2} \rceil = 78$ steps. The Q-network is a Convolutional Neural Network (Lecun et al., 1998, CNN) of three layers (with 32, 64 and 32 filters respectively and a kernel of size 3) that are flattened and then fed through three linear layers of 64 neurons. All layers have a ReLU activation function. Since the agents share the same parameters, we concatenate the flattened output of the CNN with their one-hot encoded agent ID. We optimize the Q-network and the mixer after every episode on a batch of 32 episodes with an ADAM optimizer for both the Q-network and the mixer with a learning rate of $5{\times}10^{-4}$, use a $\gamma = 0.95$, clip the norm of the gradients to 10, update

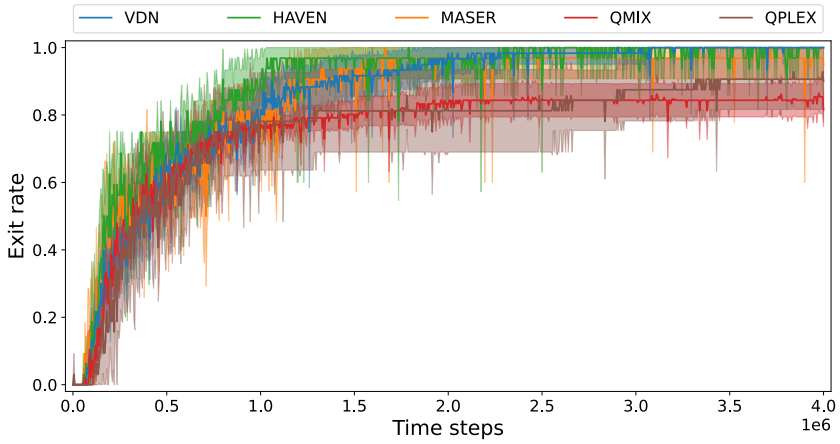

Figure 5: Exit rate over the course of the training for subgoal-oriented methods when PBRS is applied. Results are averaged on 16 different seeds and shown with 95% confidence intervals. All algorithms successfully complete the collaborative task (i.e. an exit rate of 1).

the target network every 200 time steps, and use a replay memory of 5k episodes. For HAVEN, we use VDN as mixing network and the meta-agent time scale $k = 3$. For MASER, the intrinsic reward weight $\lambda = 0.03$, the weight of individual vs global targets is 0.5 and we use VDN as mixing method.

Additionally, we provide in Figure 5 the results with the same methodology but when the completion of subtasks is rewarded, which shows that all methods successfully complete the collaborative task when PBRS is applied as described in Appendix D with $d = 0$.

D Delayed PBRS methodology

Let C be 2-dimensional a matrix initialized at -1 that indicates at $C_{i,l}$ since how many steps agent i has crossed laser l during the current episode. We define the potential function ϕ as shown in Equation 2.

$$\phi(s) = -\sum_{i=1}^{n}\sum_{l}\begin{cases}1 \text{ if } C_{i,l} \leq d\\ 0 \text{ otherwise}\end{cases} \tag{2}$$

In order to let agents discriminate between a state where a shaped reward has already been received for blocking a specific laser or not, we concatenate the inputs of agent i with C_i, i.e. the value of its corresponding countdowns.

We train agents with VDN (Sunehag et al., 2018) for 300k time steps on the LLE map shown in Figure 4 where the agents randomly spawn above the top laser. We use an ϵ-greedy policy where ϵ is annealed from 1 down to 0.05 over 100k time steps, the maximal episode time limit is set to 28 steps and the replay memory stores up to 50k transitions. The Q-network is identical to the one described in Appendix C. We optimize the Q-network and the mixer every 5 steps on a batch of 64 transitions with an ADAM optimizer with a learning rate of 5×10^{-4}, clip the norm of the gradients to 10 and update the target network every 200 time steps. Since the agents share the same parameters, we concatenate the flattened output of the CNN with their one-hot encoded agent ID and with their own row C_i.

Note that when an episode is truncated because the time horizon has been reached, the pending rewards are flushed in order to keep the same reward density across all values of d.

References

Sanjeevan Ahilan and Peter Dayan. Feudal multi-agent hierarchies for cooperative reinforcement learning, 2019. URL http://arxiv.org/abs/1901.08492.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete Problems in AI Safety, July 2016. URL http://arxiv.org/abs/1606.06565. arXiv:1606.06565 [cs].

Marcin Andrychowicz, Dwight Crow, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, Advances in Neural Information Processing Systems 30. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2017. ISBN 978-1-5108-6096-4.

Richard Bellman. *Dynamic programming*. Princeton Univ. Pr, 1957. ISBN 978-0-691-07951-6.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by Random Network Distillation, October 2018. URL http://arxiv.org/abs/1810.12894. arXiv:1810.12894 [cs, stat].

Grant C. Forbes, Nitish Gupta, Leonardo Villalobos-Arias, Colin M. Potts, Arnav Jhala, and David L. Roberts. Potential-based reward shaping for intrinsic motivation, 2024. URL http://arxiv.org/abs/2402.07411.

Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135, 1999. ISSN 1364-6613. DOI: https://doi.org/10.1016/S1364-6613(99)01294-2. URL https://www.sciencedirect.com/science/article/pii/S1364661399012942.

Minghong Geng, Shubham Pateria, Budhitama Subagdja, and Ah-Hwee Tan. HiSOMA: A hierarchical multi-agent model integrating self-organizing neural networks with multi-agent deep reinforcement learning. *Expert Systems with Applications*, 252:124117, 2024. ISSN 0957-4174. DOI: 10.1016/j.eswa.2024.124117. URL https://www.sciencedirect.com/science/article/pii/S0957417424009837.

Jeewon Jeon, Woojun Kim, Whiyoung Jung, and Youngchul Sung. MASER: Multi-agent reinforcement learning with subgoals generated from experience replay buffer. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

Seyed Jalal Kazemitabar and Hamid Beigy. Automatic discovery of subgoals in reinforcement learning using strongly connected components. In Mario Köppen, Nikola Kasabov, and George Coghill (eds.), *Advances in Neuro-Information Processing*, pp. 829–834, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-02490-0.

Pawel Ladosz, Lilian Weng, Minwoo Kim, and Hyondong Oh. Exploration in deep reinforcement learning: A survey. *Information Fusion*, 85:1–22, 2022. ISSN 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2022.03.003. URL https://www.sciencedirect.com/science/article/pii/S1566253522000288.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. DOI: 10.1109/5.726791.

Amy McGovern and Andrew G Barto. Automatic discovery of subgoals in reinforcement learning using diverse density. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 361–368, 2001.

Fanxiao Meng. Research on Multi-agent Sparse Reward Problem. *Highlights in Science, Engineering and Technology*, 85:96–103, March 2024. ISSN 2791-0210. DOI: 10.54097/er0mx710. URL https://drpress.org/ojs/index.php/HSET/article/view/18307.

Yannick Molinghen, Raphaël Avalos, Mark Van Achter, Ann Nowé, and Tom Lenaerts. Laser learning environment: A new environment for coordination-critical multi-agent tasks. In Frans A. Oliehoek, Manon Kok, and Sicco Verwer (eds.), *Artificial Intelligence and Machine Learning, BNAIC/Benelearn Conference Proceedings*, volume 2187 of *Communications in Computer and Information Sciences*, pp. 135–154, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-74650-5.

Jim Martin Catacora Ocana, Roberto Capobianco, and Daniele Nardi. An overview of environmental features that impact deep reinforcement learning in sparse-reward domains. *Journal of Artificial Intelligence Research*, 76:1181–1218, 2023. ISSN 1076-9757. DOI: 10.1613/jair.1.14390. URL https://www.jair.org/index.php/jair/article/view/14390.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 2778–2787. JMLR.org, 2017.

Jette Randløv and Preben Alstrøm. Learning to Drive a Bicycle using Reinforcement Learning and Shaping. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 463–471, San Francisco, 1998. Morgan Kaufmann Publishers Inc.

Tabish Rashid, Mikayel Samvelyan, and Christian Schroeder. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of Machine Learning Research*, 2018. arXiv: 1803.11485v2.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The StarCraft Multi-Agent Challenge, December 2019. URL http://arxiv.org/abs/1902.04043. arXiv:1902.04043 [cs, stat].

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 22(8):888–905, 2000.

Özgür Şimşek, Alicia P. Wolfe, and Andrew G. Barto. Identifying useful subgoals in reinforcement learning by local graph partitioning. In *Proceedings of the 22nd international conference on Machine learning - ICML '05*, pp. 816–823. ACM Press, 2005. ISBN 978-1-59593-180-1. DOI: 10.1145/1102351.1102454. URL http://portal.acm.org/citation.cfm?doid=1102351.1102454.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, 3: 2085–2087, 2018. ISSN 15582914.

Saim Sunel, Erkin Çilden, and Faruk Polat. Faster MIL-based subgoal identification for reinforcement learning by tuning fewer hyperparameters. *ACM Transactions on Autonomous and Adaptive Systems*, 19(2):1–29, 2024. ISSN 1556-4665, 1556-4703. DOI: 10.1145/3643852. URL https://dl.acm.org/doi/10.1145/3643852.

Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: an introduction*. Adaptive computation and machine learning series. The MIT Press, Cambridge, Massachusetts, second edition edition, 2018. ISBN 978-0-262-03924-6.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181–211, 1999. ISSN 00043702. DOI: 10.1016/S0004-3702(99)00052-1. URL https://linkinghub.elsevier.com/retrieve/pii/S0004370299000521.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. #exploration: A study of count-based exploration for deep reinforcement learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3a20f62a0af1aa152670bab3c602feed-Paper.pdf.

Alexander Trott, Stephan Zheng, Caiming Xiong, and Richard Socher. Keeping your distance: Solving sparse reward tasks using self-balancing shaped rewards. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/64c26b2a2dcf068c49894bd07e0e6389-Paper.pdf.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-Learning. *30th AAAI Conference on Artificial Intelligence, AAAI 2016*, pp. 2094–2100, 2016. arXiv: 1509.06461 ISBN: 9781577357605.

Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. QPLEX: Duplex Dueling Multi-Agent Q-Learning, October 2021. URL http://arxiv.org/abs/2008.01062. arXiv:2008.01062 [cs, stat].

Robin J. Wilson. *Introduction to graph theory*. Prentice Hall, Harlow Munich, 4. ed., [nachdr.] edition, 2009. ISBN 978-0-582-24993-6.

Zhiwei Xu, Yunpeng Bai, Bin Zhang, Dapeng Li, and Guoliang Fan. HAVEN: Hierarchical cooperative multi-agent reinforcement learning with dual coordination mechanism. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11735–11743, 2023. DOI: 10.1609/aaai.v37i10.26386. URL https://ojs.aaai.org/index.php/AAAI/article/view/26386. Number: 10.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *Proceedings of the 16th International Conference on Machine Learning*, pp. 278–287, San Francisco, 1999. Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-612-8.

Supplementary Materials

The following content was not necessarily subject to peer review.

E Reward density computation

The number of edges of Figure 2 is computed as follows.

- We consider that bidirectional edges account for two edges;
- There are 5 actions (north, east, south, west and stay);
- The three centre states each have 5 neighbours, i.e. $9 \times 5 = 45$;
- The four corners have 3 neighbours, i.e. $4 \times 3 = 12$;
- The exit tile has no neighbour, it is a goal state;
- The 11 remaining tiles on the border each have 4 neighbours, i.e. $11 \times 4 = 44$.

The total number of edges is $(5 \times 9) + (3 \times 4) + (11 \times 4) = 101$ edges. Among these edges, there are three leading to the goal state that are rewarded by +1. The reward density is therefore $\frac{3}{101}$.

F Graph-based approaches

There are multiple graph-based approaches, and we investigate the one of Şimşek et al. (2005) who build an approximation of the MDP during training and then use spectral clustering to identify bridges in the graph, i.e. SSBs. We adapt this method to the multi-agent case and test it on the very simple map shown in Figure 6a that has one explicit SSB and matches the one of the original work.

Methodology We record training episodes and after every 5 episodes, we build a connected local graph of the MDP according to the encountered states and edges. Then, we perform a binary spectral clustering (Shi & Malik, 2000) of vertices to identify edges that lie at the intersection of densely connected areas of the state space. Over the course of the training, agents build a database of how many times each edge has been identified as a bottleneck which we refer to as its score.

Similarly to Şimşek et al. (2005), we represent the results on the vertex level rather than on the edge level for visualization purposes. To do so, we assign to each vertex $v = (x, y)$ the sum of the scores of all the edges where at least one agent is in (x, y), as shown in Equation 3 where (x_i, y_i) is the location of agent i in a given state s.

$$\text{score}_{x,y} = \sum_{s \in S} \sum_{i=1}^{n} \begin{cases} \text{score}(s) \text{ if } (x_i, y_i) = (x, y) \\ 0 \text{ otherwise} \end{cases} \tag{3}$$

We randomly explore the state space for 1 million time steps with one to four agents to identify subgoals in the map shown in Figure 6a. We show the subgoal identification for one to three agents in Figure 6, and the corresponding computation duration in Figure 7.

Results As shown in Figure 6, although the bottleneck identification works very well for a single agent, the bottleneck identification decreases with the number of agents. With two agents, the bottleneck can still be identified but with much less confidence, as indicated by the slightly lighter colour of the bottleneck state, but from three agents onwards, the bottleneck state is no longer identified. Moreover, the computation time exponentially increases with the number of agents, as shown in Figure 7, which makes this method unusable in practice.

We conclude that this method scales poorly with the number of agents in terms of ability to identify subgoals and in terms of computation time. We attribute this phenomenon to the low connectivity of

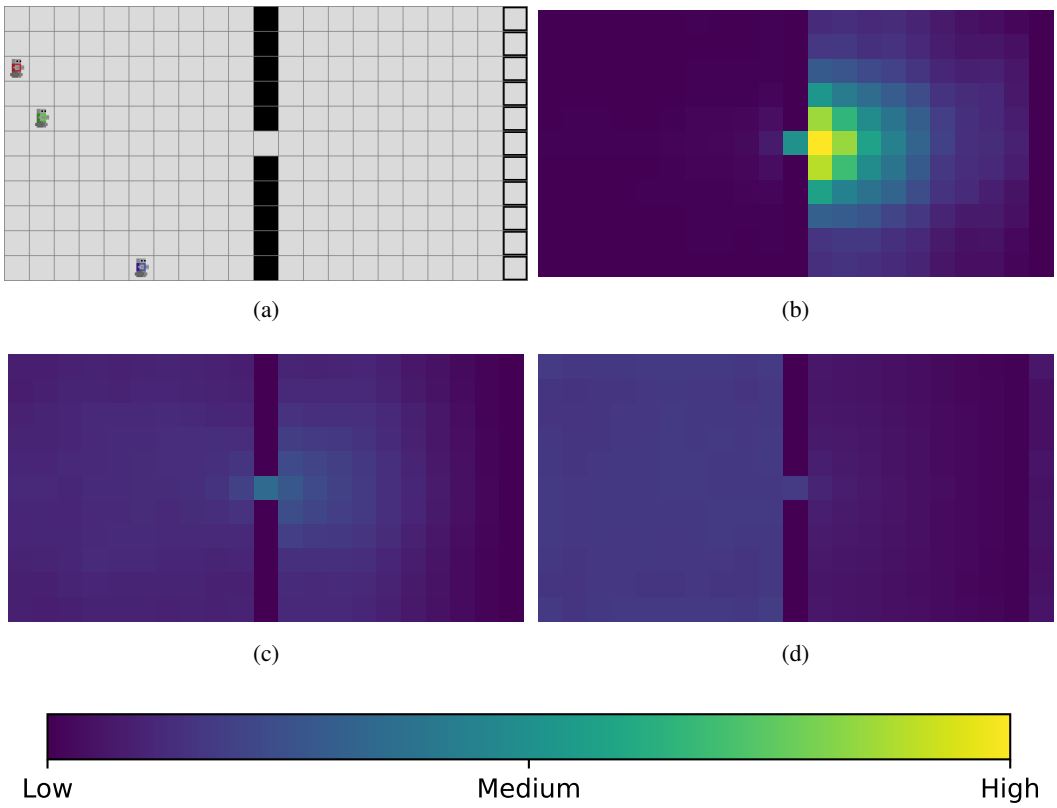

Figure 6: (a) Map with a State Space Bottleneck. (b), (c) and (d) respectively show the bottleneck score for each vertex to be identified as a bottleneck for one, two and three agents. Bright colours indicated a high score while dark colours indicates a low score.

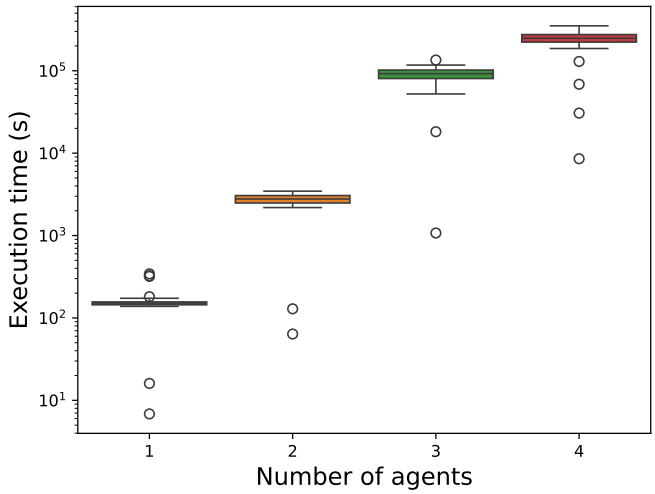

Figure 7: Average execution time (in seconds, logarithmic scale) to identify bottlenecks in the map shown in Figure 6a for one to four agents. Results are averaged on 30 runs.

the sub-graphs because of the exponential growth the the state space with the number of agents that make it unlikely to encounter the same state multiple times on the course of an episode. We expect analogous methods (Kazemitabar & Beigy, 2009) to face similar challenges.