14th International Conference on Applications of Statistics and Probability in Civil Engineering, ICASP14
Dublin, Ireland, July 9-13, 2023

Maintenance Planning for Bridges using Hierarchical Reinforcement Learning

Zachary Hamida
Postdoctoral Researcher, Dept. of Civil, Geologic and Mining Engineering, Polytechnique Montreal, Montreal, Canada

James-A. Goulet
Professor, Dept. of Civil, Geologic and Mining Engineering, Polytechnique Montreal, Montreal, Canada

ABSTRACT: This paper presents a hierarchical deep RL framework for maintenance planning on bridges. The proposed HRL framework provides advantages in scalability, and interpretability by allowing to visualize the decision boundaries of policies. The RL environment in this study is based on state-space models (SSM), which enables including the deterioration speed alongside the condition in the decision-making analyses. The performance of the proposed approach is evaluated by learning a maintenance policy for the beams structural category within a bridge in the Quebec province, Canada.

1. INTRODUCTION

The aim of bridge maintenance planning is to sustain the safety and connectivity of the transportation network, while adhering to constraints, such as the availability of resources and budget (Lei et al., 2022). In this context, infrastructure owners have to make decisions at different levels, such as when to maintain a bridge, which structural category (e.g., beams, slabs,..., etc.) to maintain, and what type of maintenance action to take on each structural element. Accordingly, a maintenance policy is required to take into account information about the overall health state of a bridge, as well as the health states for each of its components. Such requirements impose challenges on formulating and solving maintenance planning problems on bridges, mainly because of the high number of structural elements in each of them.

Existing research work in the context of maintenance planning have mainly adopted a Markov decision process (MDP) formulation to describe the planning problem and facilitate the search for optimal maintenance policies (Du and Ghavidel, 2022; Lei et al., 2022). The MDP approach is well-suited for deterministic tasks with small state and action spaces; however, in the context of maintenance planning the state space is inherently large due to including and acting on the health states of many structural elements (Andriotis and Papakonstantinou, 2019; Fereshtehnejad and Shafieezadeh, 2017). This challenge caused relying on modelling simplifications on the state space, such as merging the state representation of similar structural elements (Fereshtehnejad and Shafieezadeh, 2017). Moreover, the large state space has also propelled the use of deep reinforcement learning (RL) methods, which have the capacity to approximate near optimal policies in MDPs (Andriotis and Papakonstantinou, 2019). Applications included the use of deep reinforcement learning (DRL) and multi-agent DRL due to their capacity of handling large and continuous state and action spaces (Tavakoli et al., 2018; Andriotis and Papakonstantinou, 2019), yet their applicability remains limited to small scale problems. This is because the large state-space (i.e., the deterioration state of each structural element) in maintenance planning coincides with a large discrete action-space (i.e., element-level actions), which is challenging for standard deep RL methods (Lillicrap et al., 2015). DRL frameworks also suffer from performance instability during the training, especially as the size of the action space increases. In addition, the policy obtained is not interpretable, such that it is not possible to plot the decision boundaries for the policy, and with the lack of a clear stopping criteria for training agents in the context of planning problems, it becomes difficult to evaluate the validity of the reward function or the policy in practical applications. Another common limitation in the context of maintenance planning for transportation infrastructure is the use of discrete Markov models (DMM) for modelling the deterioration process over time. The use of the DMM framework in this context induces drawbacks related to overlooking the uncertainty associated with each inspector, and the incapacity to estimate the deterioration speed (Hamida and Goulet, 2021b).

In this paper, a hierarchical RL framework is proposed for decision-making on transportation infrastructure, which naturally adapts to the hierarchy of information and decisions in maintenance planning. The hierarchical formulation relies on state and temporal abstractions to enhance the scalability of the deep RL framework (Abel et al., 2016). The main contributions in this paper are: 1) formulating a scalable bridge maintenance planning framework via hierarchical deep reinforcement learning, which provides advantages in interpretability through visualizing the decision boundaries of the policies, and 2) Incorporating the deterioration speed alongside the deterioration condition in the decision-making analyses. The performance of the proposed framework is demonstrated using a case study for the beams structural category within a bridge from the network of bridges in the province of Quebec, Canada.

2. METHODOLOGY

2.1. Markov & Semi-Markov Decision Processes

Markov decision processes (MDP) is an approach to formulate and solve sequential decision-making problems, defined by the state space $\mathscr{S}$, the actions space $\mathscr{A}$, the set of rewards $\mathscr{R}$ and a transition function P (Sutton and Barto, 2018). Taking an action $a \in \mathscr{A}$ in the MDP is associated with a Markovian transition following the probability $P(s'|s,a)$, where the future state $s_{t+1} = s'$ only depends on the current state $s_t = s$, and action a. In the context of MDP, a policy π corresponds to a mapping between states and actions $\pi(s) : s \rightarrow a$, and each action a taken by policy π can affect the immediate rewards r_t as well as the total rewards G_t over the trajectory of future states. Accordingly, it is possible to evaluate and compare different policies based on the total rewards G_t, which correspond to two types of functions, the value function $V_\pi(s)$ and the action-value function $Q_\pi(s,a)$. The value function is a state-centric approach which quantifies the value of being in state s_t by estimating the expected total discounted return under the policy π,

$$V_\pi(s_t) = \mathbb{E}_\pi\left[\sum_{i=0}^{\infty} \gamma^i r(s_{t+i}, a_{t+i}) | S_t = s\right], \qquad (1)$$

where $\mathbb{E}_\pi$ is the expected value under policy π, $r(s_t,a_t) = \mathbb{E}[R_t|S_t = s, A_t = a]$ is the expected reward given the state $S_t = s$ and the action $A_t = a$, and γ is the discount factor $\gamma \in]0,1[$ (Sutton and Barto, 2018). On the other hand, the action-value function $Q_\pi(s,a)$ measures the quality of being in a state s and taking an action a under the policy π,

$$Q_\pi(s_t,a_t) = \mathbb{E}_\pi\left[\sum_{i=0}^{\infty} \gamma^i r(s_{t+i}, a_{t+i}) | S_t = s, a\right]. \qquad (2)$$

Based on the definitions above, a policy π^* is considered optimal when the state-value function $V_\pi(s_t)$ and action-value function $Q(s_t,a_t)$ are maximized for all states $s_t \in \mathscr{S}$ and actions $a_t \in \mathscr{A}$,

$$\begin{aligned} V^*(s_t) &= \max_\pi V_\pi(s_t), \\ Q^*(s_t,a_t) &= \max_\pi Q_\pi(s_t,a_t). \end{aligned} \qquad (3)$$

One of the main limitations in using a MDP is the scalability, as it becomes difficult to explore the state space of tasks with large number of states or long horizons. Alleviating this limitation is possible by breaking the long task into sub-tasks, which is possible by using a semi-Markov decision processes (SMDP). The main difference between SMDPs and MDPs is that each action in the

SMDP has a duration $\bar{T}$ to be completed (Pateria et al., 2021). This property enables decomposing a problem with a large state space and action space into a hierarchy of smaller MDP problems. For example, the task of maintaining a bridge can be decomposed into a bridge-level $\ell = 1$ and an element level $\ell = 0$, where an action on the bridge-level requires a duration $\bar{T}$ to perform the actions on each element within the bridge. The expected rewards $r(s_t, a_t^\ell)$ associated with a SMDP task a_t^ℓ at level ℓ is estimated using,

$$R(s_t, a_t^\ell) = \mathbb{E}_{\pi^{\ell-1}}\left[\sum_{i=0}^{\bar{T}} \gamma^i r(s_{t+i+1}, a_{t+i+1}^{\ell-1}) | s_t, a_t^{\ell-1}\right], \quad (4)$$

where the reward $r(s_t, a_t^\ell)$ for level ℓ is the expected total discounted rewards of level $\ell - 1$ under policy $\pi^{\ell-1}$ from time t until the termination of the higher-level action a_t^ℓ after $\bar{T}$ time-steps. From Equation 4, the action-value function $Q(s_t, a_t^\ell)$ for an optimal policy is,

$$Q(s_t, a_t^\ell) = R(s_t, a_t^\ell) + \sum_{s_{t+\bar{T}}} \sum_{\bar{T}} \gamma^{\bar{T}} p(s_{t+\bar{T}}, \bar{T} | s_t, a_t^\ell) \max_{a_{t+\bar{T}}^\ell} Q(s_{t+\bar{T}}, a_{t+\bar{T}}^\ell). \quad (5)$$

From Equation 5, the transition model $p(s_{t+\bar{T}}, \bar{T} | s_t, a_t^\ell)$ and the reward $R(s_t, a_t^\ell)$ depend directly on the subsequent policy $\pi^{\ell-1}$ Pateria et al. (2021). The application of the SMDP formulation in the context of RL relies on state and temporal abstractions. State abstraction aims at reducing the search space by aggregating states having similar properties without inducing changes to the essence of the problem. Accordingly, there exist a mapping from a state $s \in \mathcal{S}$ to an abstract state $s_\phi \in \mathcal{S}_\phi$ while maintaining a near-optimal policy search in the space with a fewer states (i.e., $|\mathcal{S}_\phi| \ll |\mathcal{S}|$) (Abel et al., 2016). Figure 1 shows an illustrative example for different levels of abstraction. As for temporal abstraction, it is applied when actions are taking place at different time scales (Sutton and Barto, 2018). For example, applying an intervention on a bridge $\mathcal{B}$ from time t to time $t + 1$, involves many actions at the element-level over a sub-timestamp τ, where, $t < (t + \tau) < t + 1$.

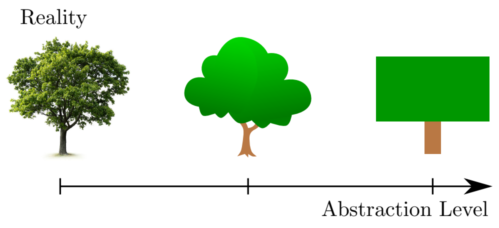


Figure 1: Illustrative example for the concept of abstraction starting from the state of reality on the left towards two abstract states on the right.

2.2. *Hierarchical Deep RL Framework*

The hierarchical framework relies on a hierarchical environment that decomposes the state space of each bridge $\mathcal{B}$ into three level, which are a bridge level with state $\boldsymbol{s}_t^b$, a structural-category level with $\boldsymbol{s}_{t,k}^c$, and an element-level with $\boldsymbol{s}_{t,p}^e$. Each of these states provide information about the health of its corresponding level. For example, the state of the bridge $\boldsymbol{s}_t^b$ could contain information about the overall deterioration condition $\tilde{x}_t^b$ and speed $\tilde{\dot{x}}_t^b$ of the bridge $\mathcal{B}$. Figure 2 shows an illustration for the states at each level within the proposed hierarchical RL architecture. From Figure 2, the hierarchical framework is composed of a RL agent for the bridge level represented by the policy π^b, and a number of decentralized agents for each structural category represented by the policy π_k. The bridge-level agent proposes a target improvement $\delta^b \leftarrow \pi^b(\boldsymbol{s}_t^b)$ for the health condition of the bridge x_t^b, such that the health condition of the bridge at time $t + 1$ is $x_{t+1}^b + \delta^b$. An improvement value $\delta^b = 0$, implies that no maintenance is applied on the bridge; while for any $\delta^b > 0$, maintenance actions are performed. It should be noted that δ^b is defined within the range $\delta^b \in [0, (u - l)]$, where l is the lower bound and u is the upper bound for the condition.

One of the main goals in the hierarchical framework is to translate the target improvement δ^b signal into a vector of actions for all structural elements in $\mathcal{B}$. This can be achieved sequentially by distributing δ^b on the structural categories according to their current deterioration condition $\tilde{x}_{t,k}^c$ using,

$$\delta_k^c(\delta^b) = \frac{u - \tilde{x}_{t,k}^c}{u \cdot \mathtt{K} - \sum_{k=1}^{\mathtt{K}} \tilde{x}_{t,k}^c} \cdot \mathtt{K} \cdot \delta^b, \quad (6)$$

where δ^c_k is the target improvement for the k-th structural category $\mathscr{C}_k$, $\mathtt{K}$ is the total number of structural categories within the bridge, and u is the perfect condition. From Equation 6, if $\delta^c_k > 0$, then the structural element $e^k_p \in \mathscr{C}_k$ is maintained according to the policy π_k. Thereafter, the states of the structural category $\tilde{s}^c_{t,k}$, and the bridge $\tilde{s}^b_t$ are updated with the state after taking the maintenance action $a^k_{t,p} \leftarrow \pi_k(\boldsymbol{s}^e_{t,p})$ on the element e^k_p. In order to determine if the next structural element $p+1$ requires maintenance, the target improvement δ^c_k and δ^b are updated using,

$$\begin{aligned}\delta^c_k &= \max\left(\tilde{x}^c_{t,k(\text{before maintenance})} + \delta^c_k - \tilde{x}^c_{t,k\,(\text{updated})}, 0\right),\\ \delta^b &= \max\left(\tilde{x}^b_{t\,(\text{before maintenance})} + \delta^b - \tilde{x}^b_{t\,(\text{updated})}, 0\right).\end{aligned} \tag{7}$$

Once the updated target improvement δ^c_k reaches $\delta^c_k = 0$, the remaining structural elements within $\mathscr{C}_k$ are assigned the action (a_0:*do nothing*). The aforementioned steps are repeated for each structural category $\mathscr{C}_k$ in bridge $\mathscr{B}$ until all elements e^k_p are assigned a maintenance action $a^k_p \in \mathscr{A}^e$.

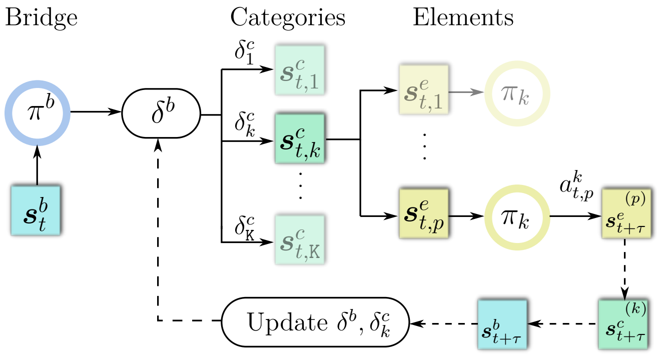


Figure 2: Hierarchical deep RL for performing maintenance using a hierarchy of policies with the policy π^b for the bridge level, and element-level policies π_k. Based on the bridge state $\boldsymbol{s}^b_t$, the policy π^b produces a target improvement δ^b, which is distributed on the structural categories to provide the category improvement δ^c_k. Each δ^c_k is sequentially translated to a vector of maintenance actions at the element-level using the policies π_k.

The element-level actions are defined by the set $\mathscr{A}^e = \{a_0, a_1, a_2, a_3, a_4\}$, where a_0: do nothing, a_1: routine maintenance, a_2: preventive maintenance, a_3: repair, and a_4: replace (MTQ, 2014). The corresponding effect associated with each of the aforementioned actions and the costs are detailed in A.2. In addition to the maintenance action costs, there are costs related to the bridge service-stoppage and penalties for reaching a critical state. The service-stoppage costs are defined to prevent frequent interruptions of the bridge service, as well as to encourage performing all of the required maintenance actions at the same time. On the other hand, the penalties are applied when a predefined critical state is reached and no maintenance action is taken. The critical state in this work is defined in accordance with the definition provided by the *Manual of Inspections* (MTQ, 2014), for a deterioration state that requires maintenance.

2.3. *Learning the Policies in the HRL*

The training for the deep RL agents is done by using a bottom-to-top approach (Pateria et al., 2021), which starts by training the element-level agents followed by training the bridge-level agent. In this study, structural elements from the same structural category (e.g., all slabs in a bridge) are assumed to share a similar deterioration kinematics, and similar maintenance actions and costs. Accordingly, the number of element-level agents is equal to the number of structural categories in $\mathscr{B}$. Moreover, the element-level agents are considered to be decentralized where each element-level agent has an individual policy based on local element-level observations (Gronauer and Diepold, 2022).
Learning element-level policies is done by using an environment that emulate the deterioration process and provides information about the health condition $\tilde{x}^k_{t,p}$ and deterioration speed $\dot{\tilde{x}}^k_{t,p}$ of the structural elements (further information about the environment in Appendix A.1). The element-level state space is defined by, $\boldsymbol{s}^e_t = [\tilde{x}^k_{t,p}, \dot{\tilde{x}}^k_{t,p}]$ and the action space is defined by the set $\mathscr{A}^e$. The element-level rewards $r(\boldsymbol{s}^e_{t,p}, a^k_{t,p})$ are defined by,

$$r(\boldsymbol{s}^e_{t,p}, a^k_{t,p}) = x_c(\tilde{x}^k_{t,p}, a^k_{t,p}) + r^p, \tag{8}$$

where $x_c(\cdot)$ is the maintenance cost based on the condition $\tilde{x}^k_{t,p}$ and action a, while r^p is the penalty applied when a predefined critical state is reached.

Further details about the costs and effects are provided in Appendix A.2. Training the element-level agents $\pi_{1:\mathtt{K}}$ is done using off-policy methods, such as deep Q-learning with experience replay and the dueling DQN (Sutton and Barto, 2018).

The trained agents with policies $\pi^*_{1:\mathtt{K}}$ provide the foundations for bridge-level decision-making, where the bridge-level agent observes the state $\boldsymbol{s}^b_t = [\tilde{x}^b_t, \tilde{\dot{x}}^b_t, \sigma^b_t]$, with $\tilde{x}^b_t$ is the overall health condition of the bridge, $\tilde{\dot{x}}^b_t$ is the overall deterioration speed of the bridge, and σ^b_t is the standard deviation for the condition of the structural categories in the bridge $\sigma^b_t = \text{std.}(\tilde{\boldsymbol{x}}^c_{t,1:\mathtt{K}})$. The environment emulates the bridge-level deterioration by aggregating the deterioration states from the elements using the framework by Hamida and Goulet (2021b). Training the bridge-level agent is done using an off-policy deep Q-learning approach with experience replay, with experience transition, $(\boldsymbol{s}^b_t, \delta^b_t, r^b_t, \boldsymbol{s}^b_{t+1})$, where r^b_t is the total costs from all actions performed on the bridge and is defined by,

$$r^b(\boldsymbol{s}^b_t, \delta^b_t) = r^s + \sum_{k=1}^{\mathtt{K}} \sum_{p=1}^{\mathtt{P}} r(\boldsymbol{s}^e_{t,p}, a^k_{t,p}), \qquad (9)$$

where r^s is the service-stoppage cost for performing the maintenance actions.

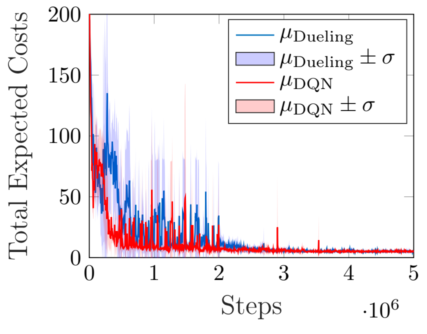
3. CASE STUDY

The case study is performed using the *InfraPlanner* RL environment (link: https://github.com/CivML-PolyMtl/InfrastructuresPlanner), which is calibrated based on the inspection and interventions database for the network of bridges in the Quebec province, Canada.

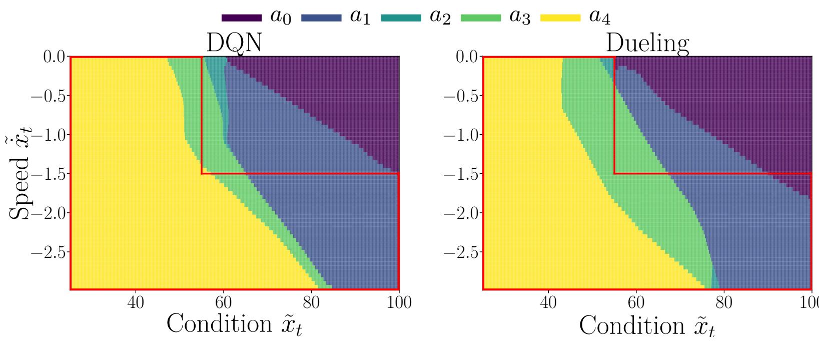
3.1. *Maintenance Policy: Structural Category*

This example demonstrates the capacity of the HRL framework to learn a maintenance policy based on a simple hierarchy, where the planning scope on bridge $\mathscr{B}$ considers only one structural category $\mathtt{K} = 1$, which corresponds to the *beams* structural category $\mathscr{C}_1$. All the beam elements in $\mathscr{C}_1$ have the same critical deterioration condition $\tilde{x}_t$ and deterioration speed $\tilde{\dot{x}}_t$ defined as, $\tilde{x}_t = 55, \dot{x}_t = -1.5$. The critical state values are defined in accordance with the *manual of inspections* (MTQ, 2014), and imply that a maintenance action is required, where taking no-action after reaching the critical state will incur a cost penalty (see Equation 8).

The training of the HRL framework starts by learning the task of maintaining beam structural elements, which corresponds to determining the type of maintenance actions, given a deterministic deterioration condition $\tilde{x}^1_{t,p}$ and speed $\tilde{\dot{x}}^1_{t,p}$. Accordingly, the element-level state space is defined by $\boldsymbol{s}^e_{t,p} = [\tilde{x}^k_{t,p}, \tilde{\dot{x}}^k_{t,p}]$, and the action space $\mathscr{A}^e = \{a_0, a_1, a_2, a_3, a_4\}$ (See Section 2.2). The cost of maintenance actions and their corresponding effects are described in Appendix A.2. Training the element-level agent is done using the *InfraPlanner* RL environment at the element level based on a total of 5×10^4 episodes. The episode length is considered as, $\mathtt{T} = 100$ years, where such a span of time allows considering replacement actions, given the assumption that on average the structural element life span is 60 years (Hamida and Goulet, 2021a). The planning horizon is considered infinite with a discount factor $\gamma = 0.99$, and the initial state in the RL environment is randomized, such that at the start of an episode, a structural element is equally likely to be either in a poor health state or a perfect health state. Learning the policy π_k is done while comparing the performance of two RL agents, which are the DQN and the Dueling agent. Both agents have similar configurations (see Appendix A.3), and their performance is shown in Figure 3, along with two realizations for the optimal policy map obtained at the end of the training. By examining the training curves in Figure 3a, the DRL agents reach a stable policy after 3×10^6 steps. From Figure 3b, it is noticeable that the critical state region (highlighted by the red boundary) is dominated by major repairs. This is expected due to applying a penalty when the agent reaches the critical state. Despite the similarity between the two policy maps, the dueling agent achieves a lower cost on a test set of 1000 episodes with the average $\mu_{\text{Dueling}} = 3.2 \pm 2.6$, compared to $\mu_{\text{DQN}} = 3.8 \pm 4.7$. Nonetheless, based on the policy maps in Figure 3b, the policy map by the DQN agent is favourable because the action a_0 : *do nothing* did not leak into the predefined critical state. The leakage of the action



(a) Average performance based on 5 seeds for DQN and Dueling agents in learning the maintenance policy $\pi_{k=1}$ for a beam structural element.



(b) Two realizations for the optimal policy maps $\pi^*_{k=1}$ based on the DQN agent (left) and the Dueling agent (right), and according to the action space $\mathscr{A}^e$. The area within the red frame represents the predefined critical state region for the condition $\tilde{x}_t$ and speed $\tilde{\dot{x}}_t$.

Figure 3: The training process of deep RL agents along with two realizations for the optimal policy $\pi^*_{k=1}$ of a beam structural element.

a_0 : *do nothing* in the critical state region can occur due to the interpolation of the Q values for states that are rarely visited by the agent, such as structural elements with a perfect condition $\tilde{x}_t = 100$, and high deterioration speed $\tilde{\dot{x}}_t = -1.6$.

After learning the policy π^*_k, the bridge-level agent is trained based on the state, $s^b_t = [\tilde{x}^b_{t,1}, \tilde{\dot{x}}^b_{t,1}, \sigma^e_t]$, where $\tilde{x}^c_{t,1}, \tilde{\dot{x}}^1_{t,1}$ represent the overall deterioration condition and speed for the bridge, and σ^e_t is the standard deviation for the condition of the elements within $\mathscr{C}_1$. The action-space has one action δ^b, which corresponds to the target improvement, with $\delta^b = 0$ being equivalent to *do nothing*, and $0 > \delta^b \geq (u - l)$ is *maintain* the beam structural elements using π^*_k. Learning the policy π^b is done using the RL environment at the bridge-level, and the same DQN agent described in Appendix A.3. The continuous action space is discretized with $\delta^b = \{\delta^b_1, \dots, \delta^b_{\mathtt{A}}\}$, to make it compatible with discrete action algorithms (Kanervisto et al., 2020), where $\mathtt{A} = 10$ discrete action that are equally spaced over the continuous domain.

In order to examine the scalability of the proposed HRL framework, the total number of beam elements in $\mathscr{C}_1 \in \mathscr{B}$ is varied with $\mathtt{P} = \{5, 10, 15\}$ elements. The performance of the HRL framework is evaluated using 5 different environment seeds, and is compared with the branching dueling Q-network (BDQN) framework. The BDQN framework architecture, hyper-parameters and configuration are adapted from Tavakoli et al. (2018). The comparison results are shown in Figure 4, where Figure 4a shows a case with $\mathscr{C}_1$ containing $\mathtt{P} = 5$ beam elements, Figure 4b shows a case with $\mathscr{C}_1$ containing $\mathtt{P} = 10$ beam elements, and Figure 4c with a case of $\mathscr{C}_1$ containing $\mathtt{P} = 15$ beam elements. From Figure 4, the performance of the proposed HRL framework is reported while considering the pre-training phase required for learning the policy $\pi_{k=1}$, which spans over 3×10^6 steps. From the results shown in Figure 4a, the HRL and BDQN frameworks reach a similar total expected costs, while the BDQN approach achieves a faster convergence due to the end-to-end training. Nonetheless, as the number of beam elements increases in the case of $\mathtt{P} = 10$ and $\mathtt{P} = 15$, the HRL framework outperforms the BDQN approach in terms of convergence speed and in the total expected costs achieved in this experiment. This can be attributed to the BDQN considering each additional beam element as a distinct branch, which leads to a significant increase in the size of the neural network model, and thus requiring a higher number of steps for the training.

4. CONCLUSIONS

In this paper, a hierarchical RL formulation is proposed for planning maintenance actions on bridges. The proposed formulation enables decomposing the bridge maintenance task into sub-tasks

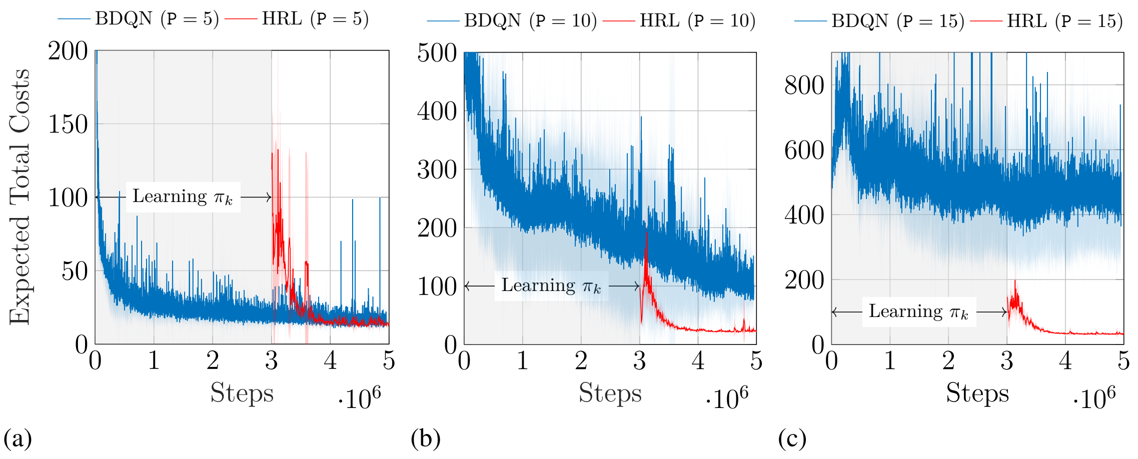


Figure 4: Comparison between the proposed HRL and BDQN for learning the maintenance policy of a structural category $\mathscr{C}_1$ with $\mathtt{P} = 5$ elements in Figure 4a, a $\mathscr{C}_1$ with $\mathtt{P} = 10$ elements in Figure 4b, and a $\mathscr{C}_1$ with $\mathtt{P} = 15$ elements in Figure 4c. The training results are reported based on the average performance on 5 seeds, with the confidence interval represented by $\pm\sigma$.

by using a hierarchy of policies, learned via deep reinforcement learning. The comparison of results in the case study have shown that the proposed hierarchical approach has a better scalability than BDQN while sustaining a similar performance. Future extensions to this framework include a multi-agent setup to learn maintenance policies for multiple structural categories in the bridge, and incorporating transfer learning.

Acknowledgements

This project is funded by Ministère des Transports du Quebec (MTQ).

A. Appendix

A.1. Environment Transitions

The RL environment is built based on the deterioration and intervention framework developed by Hamida and Goulet (2021b). The environment emulates the deterioration process by generating deterioration states for all the elements e_p^k, using the transition model,

$$\overbrace{\boldsymbol{x}_{t,p}^k = \boldsymbol{A}_t \boldsymbol{x}_{t-1,p}^k + \boldsymbol{w}_t}^{\text{transition model}},\ \underbrace{\boldsymbol{w}_t : \boldsymbol{W} \sim \mathcal{N}(\boldsymbol{w}; \boldsymbol{0}, \boldsymbol{Q}_t)}_{\text{process errors}}, \tag{10}$$

where $\boldsymbol{x}_{t,p}^k : \boldsymbol{X} \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ is a hidden state vector at time t associated with the element e_p^k. The hidden state vector $\boldsymbol{x}_{t,p}^k$ is a concatenation of the states that represent, the deterioration condition $x_{t,p}^k$, speed $\dot{x}_{t,p}^k$, and acceleration $\ddot{x}_{t,p}^k$, as well as the improvement due to interventions represented by, the change in the condition $\delta_{t,p}^e$, the speed $\dot{\delta}_{t,p}^e$, and the acceleration $\ddot{\delta}_{t,p}^e$. $\boldsymbol{A}_t$ is the state transition matrix, and $\boldsymbol{w}_t$ is the process error with covariance $\boldsymbol{Q}_t$. The RL environment relies on parameters that are learned based on a database of visual inspections and interventions from the network of bridges in the Quebec province, Canada (Hamida and Goulet, 2021b).

A.2. Costs and Effects of Maintenance Actions

The maintenance effects on the condition are defined for the *beam* elements using a data-driven approach based on the intervention framework developed by Hamida and Goulet (2021a), where: $a_0 : 0$, $a_1 : 0.5$, $a_2 : 7.5$,$a_3 : 18.75$, and $a_4 : 75$. The cost function associated with each maintenance action is considered to be dependent on the deterioration state such that,

$$x_c(\tilde{x}_{t,p}^k, a) = \beta_1(a)\frac{1}{\tilde{x}_{t,p}^k} + \beta_2(a),$$

where $\beta_1(a)$ is the cost of performing the maintenance action a as a function of the deterioration state $x_{t,p}^k$, and $\beta_2(a)$ is a fixed cost associated with maintenance action a. The derivation of this relation is empirical and mimics the cost information provided by the ministry of transportation in Quebec. Figure 5 shows the proportional cost function for the beam elements.

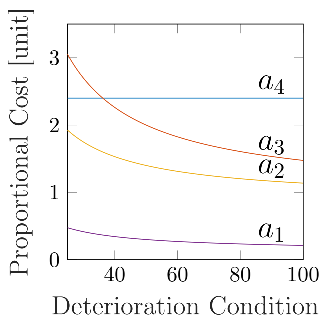


Figure 5: Proportional cost of actions as a function of the deterioration condition for Beams.

A.3. Deep RL Hyper-parameters

The RL agents at all levels are trained using a batch size of 50 samples, with the exploration performed using $\varepsilon-$greedy, which is annealed linearly over the first 200 episodes with minimum $\varepsilon_{\min} = 0.01$. Furthermore, the target model updates are performed every 100 steps in the environment. All neural networks have the same architecture (for the structural category and the bridge) which consists in 2 layers of 128 hidden units and $relu(\cdot)$ activation functions. The learning rate starts at 10^{-3} and is reduced to 10^{-5} after 800 episodes.

B. REFERENCES

Abel, D., Hershkowitz, D., and Littman, M. (2016). "Near optimal behavior via approximate state abstraction." *International Conference on Machine Learning*, PMLR, 2915–2923.

Andriotis, C. P. and Papakonstantinou, K. G. (2019). "Managing engineering systems with large state and action spaces through deep reinforcement learning." *Reliability Engineering and System Safety*, 191, 106483.

Du, A. and Ghavidel, A. (2022). "Parameterized deep reinforcement learning-enabled maintenance decision-support and life-cycle risk assessment for highway bridge portfolios." *Structural Safety*, 97.

Fereshtehnejad, E. and Shafieezadeh, A. (2017). "A randomized point-based value iteration pomdp enhanced with a counting process technique for optimal management of multi-state multi-element systems." *Structural Safety*, 65, 113–125.

Gronauer, S. and Diepold, K. (2022). "Multi-agent deep reinforcement learning: a survey." *Artificial Intelligence Review*, 55, 895–943.

Hamida, Z. and Goulet, J.-A. (2021a). "Quantifying the effects of interventions based on visual inspections of bridges network." *Structure and Infrastructure Engineering*, 1–12.

Hamida, Z. and Goulet, J.-A. (2021b). "A stochastic model for estimating the network-scale deterioration and effect of interventions on bridges." *Structural Control and Health Monitoring*, 1545–2255.

Kanervisto, A., Scheller, C., and Hautamäki, V. (2020). "Action space shaping in deep reinforcement learning." *2020 IEEE Conference on Games (CoG)*, IEEE, 479–486.

Lei, X., Xia, Y., Deng, L., and Sun, L. (2022). "A deep reinforcement learning framework for life-cycle maintenance planning of regional deteriorating bridges using inspection data." *Structural and Multidisciplinary Optimization*, 65.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). "Continuous control with deep reinforcement learning." *Arxiv*.

MTQ (2014). *Manuel d'Inspection des Structures*. Ministère des Transports, de la Mobilité Durable et de l'Électrification des Transports.

Pateria, S., Subagdja, B., Tan, A. H., and Quek, C. (2021). "Hierarchical reinforcement learning: A comprehensive survey." *ACM Computing Surveys*, 54.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Tavakoli, A., Pardo, F., and Kormushev, P. (2018). "Action branching architectures for deep reinforcement learning." *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.