UNIVERSITÀ DEGLI STUDI DI VERONA

DEPARTMENT OF COMPUTER SCIENCE

DOCTORAL PROGRAM IN COMPUTER SCIENCE

XXXVI CYCLE (2020)

Learning in Monte Carlo Tree Search Planning

Author:
Maddalena Zuccotto

Supervisor:
Dr. Alberto Castellini
Co-Supervisor:
Prof. Alessandro Farinelli

PhD Program Coordinator: Prof. Ferdinando Cicalese

Submitted in fulfillment of the requirements for the degree of Doctor of Philosophy at the Department of Computer Science

S.S.D. INF/01

Learning in Monte Carlo Tree Search Planning
Maddalena Zuccotto
PhD thesis
ISBN xxxxx-xxxx-xxx

Abstract

Artificial intelligence is playing an increasingly important role in both industry and society, as evidenced by recent applications in areas such as autonomous driving, personalized shopping, and fraud prevention. Reinforcement Learning (RL) is a prominent machine learning paradigm that focuses on learning policy functions, i.e., functions able to select sequences of actions that allow agents to optimally achieve their goals in the environment in which they act. RL has recently demonstrated strong potential in scenarios where agents must operate in unknown environments adapting to unexpected (or partially specified) situations.

This thesis covers three topics related to RL. The first topic addresses the problem of learning state-variable relationships in Partially Observable Markov Decision Processes (POMDPs) to improve planning performance. Specifically, we focus on Partially Observable Monte Carlo Planning (POMCP) and we represent the acquired knowledge with a Markov Random Field (MRF). Three methods are proposed to compute MRF parameters while the agent acts in the environment. Our techniques acquire information from the outcomes of agent actions and from the agent's belief. We answer a key question: "When can the learned state-variable relationships be trusted?". Criteria, based on confidence intervals and convergence, are introduced to determine when the MRF is accurate enough and the learning process can be stopped. We test this technique on two domains, rocksample, a standard rover exploration task, and a problem of velocity regulation in industrial mobile robotic platforms. Results show that the proposed approach allows to effectively learn state-variable probabilistic constraints and to outperform standard POMCP with no computational overhead. Finally, a ROS-based architecture is proposed which allows to perform MRF learning, adaptation, and usage in POMCP on real robotic platforms.

The second topic tackles the problem of learning the transition model of the environment in Monte Carlo Tree Search (MCTS) using information from state transition traces. The transition model is then used to improve the performance of the policy generated by MCTS. In this case, we focus in particular on fully observable environments represented by Markov Decision Processes (MDPs). We propose a MCTS-based planning approach that assumes a black-box approximated model of the environment developed by an expert using any kind of modeling framework, and it improves the model as new information from the environment is collected. This is crucial in real-world applications since having a complete knowledge of complex environments is impractical. The expert's model is first translated into a neural network and then it is updated periodically using data (i.e., state-action-next state triplets), collected as the agent acts in the environment. We propose three different methods to integrate this data with prior knowledge provided by the expert, and we evaluate our approach in a domain concerning air quality and thermal comfort control in smart buildings. We compare

the performance of MCTS using each of the three proposed model learning techniques with the performance of standard MCTS using the expert's model (without adaptation), Proximal Policy Optimization (a popular model-free DRL approach), and Stochastic Lower Bounds Optimization (a popular model-based DRL approach). Results show that our approach achieves the best results, outperforming all the competitors.

Finally, the third topic of this thesis concerns the recent application of RL to environmental sustainability, an application domain in which uncertainty challenges strategy learning and adaptation. We survey the literature to identify the main applications of RL in this domain and the predominant methods employed to address the main challenges. We analyzed 181 papers and answered seven research questions, e.g., "How many academic studies have been published from 2003 to 2023 about RL for environmental sustainability?" and "What were the application domains and the methodologies used?". Our analysis reveals an exponential growth in this field over the past two decades, with a rate of 0.42 in the number of publications (from 2 papers in 2007 to 53 in 2022), a strong interest in sustainability issues related to energy fields, and a preference for single-agent RL approaches to deal with sustainability. Finally, the survey provides practitioners with a clear overview of the main challenges and open problems in this topic that should be tackled in future research.

In summary, this thesis delves into three aspects of RL and its applications in different scenarios. In all research lines, we observe that explicitly modeling some elements of the environment and the information gathered during the agent-environment interaction, as in model-based RL, can improve the policy learning process in terms of sample efficiency and policy performance. Furthermore, MCTS and POMCP have demonstrated to lend themselves to the implementation of model-based RL algorithms since they natively use the model of the environment in their simulations. Learning this model (or part of it) as the agent evolves is an interesting challenge that we started to tackle in this work and that can have interesting future developments in model-based RL. The code of the proposed methodologies is open-source and available at `https://github.com/Zucchy/MCTS_planning`.

Contents

Chapter 1

Introduction

Artificial Intelligence (AI) is taking an increasingly important role in industry and society. AI techniques have been recently introduced in autonomous driving, personalized shopping, and fraud prevention, just to make a few examples. A crucial aspect of AI systems is their ability to learn. Reinforcement Learning (RL) (Sutton and Barto, 2018), a prominent machine learning paradigm, focuses on learning policy functions able to select the sequence of actions that allow agents to optimally reach their goals in the environment where they act. RL systems have made significant progresses in recent years, with the notable example of AlphaGo which recently defeated the world champion of the game of Go (Silver et al., 2017, 2016). In the area of AI, researchers are actively exploring the possibility of transferring the remarkable achievements obtained in games (e.g., Go, Chess, and Atari) to real-world scenarios. Games offer environments with entirely known dynamics where the uncertainty arises from the opponent's actions. On the other hand, in real-world problems, the environment is very complex and only partially known. As an example, consider a RL-controlled robot navigating in a warehouse alongside humans and other robots. In such scenarios, uncertainty arises from human-robot interaction, the presence of other robots, and ever-changing elements like weather and lighting conditions.

In this thesis, we propose methodologies that allow to learn different elements of the environment in Monte Carlo Tree Search-based planning. First, we develop a technique for learning state-variable relationships present in the environment during the execution of standard Partially Observable Monte Carlo Planning (POMCP). These relationships are then used to improve the performance of the policy generated by POMCP in the context of Partially Observable Markov Decision Processes (POMDPs). State variables could represent, for instance, task difficulties in a robotic problem, and a possible relationship could say that two tasks have similar difficulty. The approach we propose is an extension of Castellini et al. (2019b), and the original contribution is that here we learn probabilistic state-variables relationships from observations collected in the environment while in Castellini et al. (2019b) these relationships were assumed to be known a priori. As a second result, we propose techniques to learn the model of the dynamics in Monte Carlo

Tree Search (MCTS) using information from state transition traces. The transition model is then used to improve the performance of the policy generated by MCTS. In this case we focus, in particular, on completely observable environments represented by Markov Decision Processes (MDPs). The focus is on complex real-world domains for which it is not trivial to have a pre-existing specification of the model of the environment (i.e., the transition model). We only assume to have an approximated model developed by an expert. This model is used in MCTS as an initial transition model and it is periodically updated by considering the information about the dynamics of the environment that is acquired as the agent acts in the environment itself. RL, initially employed in gaming contexts, has been recently applied to real-world domains, including the environmental sustainability realm, where uncertainty challenges strategy learning and adaptation. Environmental sustainability is a worldwide key challenge attracting increasing attention due to climate change, pollution, and biodiversity decline. As a third contribution, this thesis puts forward a survey about recent applications of RL in environmental sustainability. We offer a comprehensive overview of different application domains where RL has been employed and RL methods used to tackle the problems. The objective is to introduce practitioners with state-of-the-art RL techniques currently applied to different problems related to environmental sustainability.

1.1 Reinforcement learning and Monte Carlo Tree Search based planning in MDPs and POMDPs

Planning is a problem of sequential decision making which has important applications in artificial intelligence and robotics. Over the last two decades, the interest in this topic has grown rapidly due to significant methodological advancements and its successful applications in various real-world domains, including smart buildings (Alanne and Sierla, 2022), industrial machinery controllers (Nian et al., 2020), and mobile robot navigation (Patle et al., 2019). RL has recently proved to have strong potential in applications where agents must operate in unknown environments adapting to unexpected (or partially specified) situations. The goal of RL algorithms is to learn optimal policies, namely, functions that map system states to actions that allow the agent to reach its goal. POMDPs (Kaelbling et al., 1998; Sondik, 1978) are a powerful framework for planning under uncertainty. They extend MDPs (Russell and Norvig, 2010) to the case of partially observable environments. To tackle partial observability they consider all possible states of the (agent-environment) system and assign to each of them a probability value expressing the related likelihood of being the true state. These probabilities, considered as a whole, constitute a probability distribution over states, called belief. A solution for a POMDP is a policy that maps beliefs into actions. The computation of optimal policies is unfeasible in practice (Papadimitriou and Tsitsiklis, 1987), therefore a lot of effort was put into the development of approximate (Hauskrecht, 2000) and online (Ross et al., 2008)

solvers. Among the main MCTS-based solvers (Kocsis and Szepesvári, 2006; Thrun, 2000) a meaningful improvement was obtained by POMCP (Silver and Veness, 2010), a pioneering algorithm that allows applying model-based reinforcement learning to very large state spaces, overcoming the scalability problem that has limited the usage of POMDPs for many years. The standard version of POMCP does not consider any kind of prior knowledge about state-variable relationships. Castellini et al. (2019b) proposed an extension of POMCP that considers these relationships, demonstrating that introducing such knowledge enhances planning performance without increasing time complexity. However, it assumes full knowledge of state-variable constraints.

RL methods (Sutton and Barto, 2018) can be split into two main categories, model-free and model-based (Moerland et al., 2023). Model-free RL has reached strong results in the last twenty years and focuses, in particular, on algorithms that learn the policy without making any assumption about the dynamics of the environment. These methods directly learn the policy from relationships among states, actions, and rewards observed in the environment while the agent acts. On the other hand, model-based RL leverages interactions with the environment to learn a model that is used in the computation of the policy. This approach provides a richer formalism allowing integrating and, therefore, exploiting prior knowledge about the environment (Zuccotto et al., 2022a,b, 2021; Castellini et al., 2019b), hence achieving better performance, or dealing with the risk introduced by taking actions in a partially observable (Mazzi et al., 2023a; Simão et al., 2023; Mazzi et al., 2021b; Castellini et al., 2018b) or completely known environment (Castellini et al., 2023a). However, model-based RL is appropriate in domains in which some useful information about the environment (e.g., the mathematical form of the model) is available a priori, although other information (e.g., the parameters of the model) is unknown.

1.2 Reinforcement learning applications to environmental sustainability

A key challenge faced by today's society for which AI can bring an important advancement is environmental sustainability. Climate change, pollution, biodiversity decline, poor health, and poverty have led in the last years governments and companies to focus more and more their efforts and investments on solutions to environmental sustainability problems, which are usually characterized by an inefficient and increased use of resources. Environmental sustainability can be defined as a set of constraints regarding the use of renewable and nonrenewable resources on the one hand, pollution, and waste assimilation on the other (Goodland, 1995). In this regard, in 2015, the United Nations published the "2030 Agenda for Sustainable Development" the centerpiece of which is 17 Sustainable Development Goals (United Nations, 2015) to be fully achieved by 2030 to attain sustainable development in the economic, social, and environmental contexts, and eliminate all forms of poverty.

AI-based algorithms can control autonomous drones used in water monitoring (Bianchi et al., 2023; Marchesini et al., 2021; Steccanella et al., 2020), extract from acquired data new insight about environmental conditions (Castellini et al., 2023b, 2022b; Azzalini et al., 2020; Castellini et al., 2020a,b, 2019e,a,d, 2018a), improve the healthiness of indoor environments (Capuzzo et al., 2022), or demand forecast in district heating networks (Castellini et al., 2022a, 2021a; Bianchi et al., 2019). Several AI techniques have been employed to address various environmental sustainability challenges. These approaches enable the efficient management of distributed resources within smart grids (Orfanoudakis and Chalkiadakis, 2023; Roncalli et al., 2019), improve the power flow for DC grids (Der Blij et al., 2020), increase the utilization of renewable resources for electric vehicle charging (Koufakis et al., 2020), and mitigate carbon emissions in urban transportation by fostering ridesharing and reducing traffic congestion (Bistaffa et al., 2021, 2017). Furthermore, a crucial aspect of climate change prevention involves optimizing the energy consumption associated with heating and cooling residential properties. To tackle this issue, AI-based approaches have been developed to enhance the efficiency of home systems (Auffenberg et al., 2017; Panagopoulos et al., 2015) and quantify the thermal efficiency of residences (Brown et al., 2021).

The application of RL to environmental sustainability has attracted, in the last decade, strong interest from both the computer science community and the communities of environmental sciences and business. Reducing carbon emissions requires increasing renewable resources usage, such as solar and wind power. While these resources are economically efficient, their stochastic and intermittent nature poses challenges in replacing nonrenewable energy sources within energy networks. RL, with a systematic trial-and-error interaction with dynamic environments, offers a promising approach for learning optimal policies that can adapt to changing system dynamics and effectively manage environmental uncertainty. Thus, an RL agent is capable of handling variations in operating conditions, for instance, due to a change in resource availability or weather conditions.

1.3 Contributions of the thesis

This thesis proposes two methodologies for learning in MCTS-based planning and a survey about applications of RL to environmental sustainability. Specifically, the first learning method (Chapter 4) enables learning state-variable relationships in partially observable environments, and the second method (Chapter 5) allows learning the model of the dynamics of a fully observable environment by progressively refining a pre-defined approximate model. Both of these methodologies are used to improve planning performance. The survey about applications of RL for environmental sustainability (Chapter 6) provides a comprehensive overview of the different application domains where RL has been used, such as energy and water resource management and traffic management. In the following, we provide further

details on each of these three contributions and a summary of the related publications.

1.3.1 Learning environment properties in POMCP

The first contribution of the thesis (Section 4.3) is the development of algorithms to learn environment properties of interest while agents act in partially observable domains. In particular, we propose three different learning methods to express the knowledge acquired step-by-step by the agent during the execution of POMCP as probabilistic constraints on state variables. The first learning approach uses observations in the real world, while the other two consider information in the belief of the agent, and respectively, the maximum likelihood state and a weighted sum of the states (where weights are belief probabilities). Furthermore, we answer a key question: "When can the learned state-variable relationships be trusted?". To this aim, we propose two criteria, based on confidence intervals and convergence, respectively, to decide when the learning phase can be stopped and the acquired knowledge can be used by POMCP to achieve performance improvement (Section 4.4). The standard version of this algorithm does not consider any kind of prior knowledge about state-variable relationships. In Castellini et al. (2019b), an extension of POMCP has been proposed to consider these relationships. In that work, it is shown that the introduction of such knowledge provides an improvement in terms of planning performance, with no additional overhead in terms of time complexity. However, it is assumed to have full knowledge of the state-variable constraints. This knowledge could be provided, for instance, by experts. Here, instead, we deal with a methodology for learning this knowledge.

As part of this contribution, we implemented a framework (Section 4.5) to integrate POMCP into ROS to enable the use of the probabilistic state-variable relationship learning algorithms on real robotic platforms, with experiments performed on Gazebo simulators of known application domains. The ROS-based architecture allows learning such relationships on real robotic platforms. It consists of three ROS nodes, namely, environment, agent, and planning. The environment node discretizes the real world exploiting a task-specific representation. The agent node, instead, holds information about odometry and interfaces the ROS-based robotic platform with the environment and the planner. Finally, the planner node runs the learning algorithm.

Moreover, we developed an algorithm (Section 4.6), called Adapt, which deals with cases in which we use the learned state-variable relationships in episodes having unlikely state-variable configurations. This algorithm runs when the knowledge provided by the learned state-variable relationships does not reflect the true state-variable values. In such cases, the acquired knowledge is misleading, since it forces the belief probabilities towards configurations of state variables that are discordant from that of the true state, decreasing the probability of the true state. Thus, the proposed algorithm adapts (i.e. changes) the probabilistic relationships when the agent acquires knowledge about the true state-variable values and detects a mismatch be-

tween the learned information and the specific state-variable relationships of the episode to fix the mismatch. The adaptation is performed online, as POMCP works, limiting the performance decrease that could derive from using the acquired knowledge when the true state-variable configuration represents an unlikely state.

Finally, our approaches have been evaluated on two synthetic domains (Sections 4.8, 4.9, 4.10, and 4.11), namely, rocksample (Smith and Simmons, 2004), a benchmark domain in which an agent moving in a grid has to collect hidden rocks maximizing their values, and velocity regulation (Castellini et al., 2021b, 2020c), a domain in which a robot traveling on a predefined path has to regulate its velocity to minimize the time to reach the end and the collisions with obstacles in the path.

1.3.2 Learning the environment model in MCTS-based planning

The second contribution of this thesis is the development of a MCTS-based planning approach that improves an expert-defined approximate model of the dynamics according to the information gathered online from the environment (Section 5.2), which is formalized as an MDP (Puterman, 1994). The technique employs Artificial Neural Networks (ANNs) (Goodfellow et al., 2016) to represent the model of the dynamics and MCTS (Browne et al., 2012) as an internal planner using the ANN to perform simulations. The approach first transforms the approximate model provided by the expert into a neural network. In this way, it allows the usage of any kind of expert-defined model of the dynamics (e.g., rule-based, probability-based, data-driven, etc.). Then, it uses the ANN model as a simulator in MCTS to compute action Q-values in the visited states. Third, it performs optimal actions (i.e., optimal according to the current model of the environment and the number of simulations used by MCTS) in the real environment (which is typically different from the current model) and collects the observed next states and reward signals. Finally, it uses information about the next states and rewards to update the ANN model of the dynamics, which will be used by MCTS to compute optimal actions in the following steps. The online nature of MCTS well suits the online update of the ANN model. The choice of MCTS as a planner is related to its capability to compute the Q-function online and locally (i.e., only for the states actually visited by the agent), which allows scaling to very large domains typical of real-world applications. Furthermore, MCTS sampling does not require a full transition matrix, unlike several RL methods, but it only necessitates a black-box simulator, namely, a function providing the next state given the current state-action pair.

A key point of our approach is related to how to integrate new information into the current model. To this end, we propose three ways to perform the integration. The first version simply periodically updates the weights of the copy neural network using environment data as a training set. The second version merges data (i.e., state-action-next-state triplets) generated by the original expert's model with data from the environment and periodically re-trains the copy neural network using this dataset as a training set. The

third version keeps the copy neural network of the expert's model unchanged and trains a separate neural network with data from the environment then, during Monte Carlo simulations, it selects the best model to perform each step according to how close the current state-action pair is to the training set by which the model has been trained.

We evaluated our approach on a real-world application related to air quality and thermal comfort control in smart buildings (Section 5.5). In the domain we have developed (Bianchi et al., 2023; Capuzzo et al., 2022), the planner acquires online information about internal temperature, external temperature, CO_2 level, Volatile Organic Compounds (VOC) level, and people occupancy and suggests at each step the best action to perform among activation/deactivation of ventilation, activation/deactivation of sanitization, opening/closing windows. These actions should consider future presence in the room and future evolution of external temperature (which are considered available throughout the current day) hence the planning component is crucial to achieve good performance.

1.3.3 RL in environmental sustainability

The third contribution of this thesis (Chapter 6) is a survey of recent applications of RL to environmental sustainability. We provide a comprehensive overview of the different application domains where RL has been used, such as energy and water resource management, and traffic management. The goal is to show practitioners the state-of-the-art RL methods that are currently used to solve environmental sustainability problems in each of these domains. For each paper analyzed, we consider the problem tackled, the RL approach used, the challenges faced, and the formalization of the RL problem (i.e., type of state/action space, type of transition model, type of RL method, performance measures used to evaluate the results).

1.3.4 Summary of contributions

This thesis provides the following contributions to the state-of-the-art.

1. Learning environment properties in POMCP
   (a) Development of three approaches for learning probabilistic relationships between state-variables (e.g. traffic intensity in different isles of an automated warehouse) in the context of Partially Observable Monte Carlo Planning.
   (b) Introduction of a framework to integrate POMCP within ROS, targeting ROS-based mobile robots. The architecture supports both the phase in which state-variable relationships are learned and the phase in which such knowledge is used.
   (c) Development of an algorithm for adapting the variable values constraints in POMCP to episodes having unlikely state-variable configurations, as new observations are acquired from the environment.

   (d) Evaluation of contributions 1.a, 1.b, 1.c on two synthetic environments, namely, rocksample (Smith and Simmons, 2004), a benchmark domain for POMDP solvers inspired by robotic planet exploration, and velocity regulation (Castellini et al., 2021b, 2020c), where a mobile robot navigates a pre-defined path.

2. Learning the environment model in MCTS-based planning

   (a) Development of an adaptability approach to improve transition models in MCTS as new data from the environment is available.

   (b) Evaluation of contribution 2.a on a designed real-world domain concerning air quality and thermal comfort control in an indoor environment (Bianchi et al., 2023; Capuzzo et al., 2022).

3. RL in environmental sustainability

   (a) Survey of applications of RL to environmental sustainability. We provide a comprehensive overview of the different application domains and RL methods for environmental sustainability.

1.4 Organization of the thesis

The rest of the thesis is organized as follows:

- Chapter 2 provides background and mathematical notions. It describes the MDPs and POMDPs frameworks for modeling sequential decision-making problems and provides an in-depth description of the MCTS and POMCP algorithms. Then, it introduces the mathematical framework used to express probabilistic relationships among state values, and outlines the fundamental concepts of ANNs. Finally, the benchmark domains used to test the presented methodologies are introduced.

- Chapter 3 presents the related work. It discusses the state-of-the-art on model-free and model-based reinforcement learning methods, connections between probabilistic planning and learning focusing on Monte Carlo methods, model learning in MCTS-based planning, the application of MCTS-based methods to robotic platforms, planning under uncertainty, continual learning, and applications of RL approaches to environmental sustainability.

- Chapter 4 presents the three methodologies to learn probabilistic state-variable relationships as the agent acts in the environment to improve the performance of POMCP, describes the two proposed stopping criteria and introduces the framework to enable the use of such learning algorithms on robotic platforms. It also outlines an approach that adapts the learned state-variable relationships if a mismatch with the true values is detected. Finally, it reports the empirical evaluation of the approaches proposed to learn state-variable relationships on the rocksample and velocity regulation domains.

- Chapter 5 presents the adaptive model-based approach to adapt MCTS transition models in MCTS, outlining the three techniques proposed to integrate new information into the current model. Then it describes the application domain we developed to simulate a real-world problem concerning air quality and thermal comfort control in smart buildings and reports the empirical evaluation of the approaches proposed on this domain.

- Chapter 6 presents the results of a survey about the application of RL to environmental sustainability. It provides a comprehensive overview of the different application domains where RL has been used, such as energy and water resource management, and traffic management. The goal is to show practitioners the state-of-the-art RL methods that are currently used to solve environmental sustainability problems in each of these domains.

- Chapter 7 draws conclusions with final considerations and presents future research directions.

1.5 Publications

Most of the content presented in this thesis has been published in international conferences and journals. Specifically, the learning state-variable relationship approaches (Chapter 4) and their evaluation (Chapter 5) have been published in Zuccotto et al., Frontiers in Robotics and AI, 2022 (see point 3 below), Zuccotto et al., SAC, 2022 (see point 2 below), and in Zuccotto et al. AIRO, 2021 (see point 1 below). The use of MCTS-based planning with dynamics model adaptability has been introduced in Bianchi et al., Ital-IA, 2023 (see point 5 below) and in Capuzzo et al., RTSI, 2022 (see point 4 below). The methods for learning the environment dynamics model (Chapter 6) and the results of their performance assessment have been published in Zuccotto et al., Optimization and Engineering - Special Issue on Machine Learning and Inverse Problems, 2024 (see point 6 below). The survey on the application of RL techniques to environmental sustainability has been published in Zuccotto et al., Artificial Intelligence Review, 2024 (see point 7 below).

1. M. Zuccotto, A. Castellini, M. Piccinelli, E. Marchesini, and A. Farinelli. Learning environment properties in partially observable Monte Carlo planning. In Proceedings of the 8th Italian Workshop on Artificial Intelligence and Robotics - A workshop of the 20th International Conference of the Italian Association for Artificial Intelligence (AIRO@AI*IA), volume 3162, pages 50–57. CEUR-WS.org, 2021.

2. M. Zuccotto, A. Castellini, and A. Farinelli. Learning state-variable relationships for improving POMCP performance. In Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing (SAC), pages

739–747. Association for Computing Machinery (ACM), 2022a. doi: 10.1145/3477314.3507049.

3. M. Zuccotto, M. Piccinelli, A. Castellini, E. Marchesini, and A. Farinelli. Learning state-variable relationships in POMCP: A framework for mobile robots. Frontiers in Robotics and AI, 9:663–704, 2022b. doi: 10.3389/FROBT.2022.819107

4. M. Capuzzo, A. Zanella, M. Zuccotto, F. Cunico, M. Cristani, A. Castellini, A. Farinelli, and L. Gamberini. IoT systems for healthy and safe life environments. In Proceedings of the 7th IEEE Forum on Research and Technologies for Society and Industry Innovation (RTSI), pages 31–37. IEEE Press, 2022. doi: 10.1109/RTSI55261.2022.9905193.

5. F. Bianchi, D. Corsi, Luca M., D. Meli, F. Trotti, M. Zuccotto, A. Castellini, and A. Farinelli. Safe and efficient reinforcement learning for environmental monitoring. In Proceedings of the Italia Intelligenza Artificiale - Thematic Workshops co-located with the 3rd CINI National Lab AIIS Conference on Artificial Intelligence (Ital-IA), volume 3486, pages 610–615. CEUR-WS.org, 2023.

6. M. Zuccotto, E. Fusa, A. Castellini, and A. Farinelli. Online model adaptation in Monte Carlo tree search planning. Optimization and Engineering - Special Issue on Machine Learning and Inverse Problems, 2024. doi: 10.1007/s11081-024-09896-2.

7. M. Zuccotto, A. Castellini, D. La Torre, L. Mola, and A. Farinelli. Reinforcement learning applications in environmental sustainability: A review. Artificial Intelligence Review, 57, 2024. doi: 10.1007/S10462-024-10706-5.

Chapter 2

Background

In this chapter, we formally describe the background necessary to tackle the *learning* and *planning* problems discussed in this thesis. Section 2.1 defines MDPs, a mathematical framework useful for modeling sequential decision-making problems; Section 2.2 presents Partially Observable Markov Decision Processs (POMDPs) which are an extension of MDPs considering uncertainty in planning problems; Section 2.3 introduces MCTS, a state-of-the-art approach to efficiently compute approximated and online solutions in high dimensional domains; Section 2.4 explains POMCP, a Monte Carlo based algorithm to plan in partially observable environments; Section 2.5 describes Markov Random Fields (MRFs) a framework for expressing probabilistic relationships among variable values; Section 2.6 details some concepts about ANNs, a widely used computational model; finally, Section 2.7 describes the benchmark domains used in the experiments.

2.1 Markov Decision Processes

A Markov Decision Process (MDP) (Puterman, 1994) is a mathematical framework to model sequential decision problems in stochastic and fully observable environments. An MDP can be defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$ (Kaelbling et al., 1998) where

- $\mathcal{S}$ is a finite set of *states*,
- $\mathcal{A}$ is a finite set of *actions*,
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Pi(\mathcal{S})$ is the *transition model*,
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the *reward function*, and
- $\gamma \in [0, 1)$ is the *discount factor*.

$\Pi(\mathcal{S})$ defines the space of probability distributions over states. A *finite MDP* is an MDP with finite set of states $\mathcal{S}$, set of actions $\mathcal{A}$, and rewards $\mathcal{R}$. The transition model $\mathcal{T}$ defines the *dynamics* of the environment. More precisely,

the function $\mathcal{T}$ specifies the probability of transition to a certain state s' after performing the action a in the state s, that is

$$\mathcal{T}(s'|s,a) = Pr(s_t = s'|s_{t-1} = s, a_{t-1} = a). \tag{2.1}$$

In an MDP, a state has to present the *Markov property*, i.e., each state encloses all relevant information about past agent-environment interactions (Figure 2.1a) necessary to describe the environment's dynamics. The agent's goal consists in maximizing the *expected discounted return*, namely, the sum of weighted rewards over long runs

$$\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t)]. \tag{2.2}$$

The solution of an MDP is a *policy* $\pi : \mathcal{S} \times \mathcal{A}$, namely, a function that specifies the action the agent should perform in any reachable state. A policy is optimal if it maximizes the expected discounted return, namely, the state-value for each state. The value function $v_\pi(s)$ specifies the expected discounted return from each state reached by following the given policy π from s. A well-known formulation of the value function is the *Bellman equation* (Bellman, 1966):

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} \mathcal{T}(s'|s,a)\big[\mathcal{R}(r|s,a) + \gamma v_\pi(s')\big], \text{ for all } s \in \mathcal{S}. \tag{2.3}$$

The Bellman equation for v_π decomposes the value function into two components, namely, the immediate reward an agent receives by performing action a in state s (i.e., $\mathcal{R}(r|s,a)$) and the discounted future reward obtained by computing the value function in the successor state s' (i.e., $\gamma v_\pi(s')$). The discount factor γ guarantees the convergence by reducing the weight of long-term rewards.

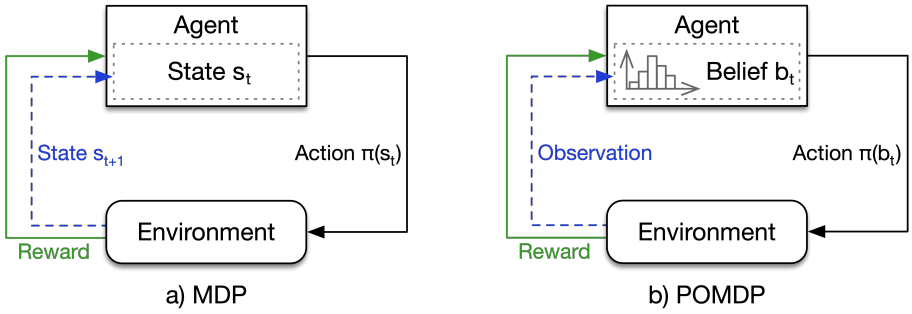

Figure 2.1: The agent-environment interaction a) in MDPs and b) in POMDPs.

2.2 Partially Observable Markov Decision Processes

A Partially Observable Markov Decision Process (POMDP) (Smallwood and Sondik, 1973) is an extension of a MDP for partially observable environ-

ments. A POMDP is defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \Omega, \mathcal{R}, \gamma)$ (Kaelbling et al., 1998) where

- $\mathcal{S}$ is a finite set of *states*,
- $\mathcal{A}$ is a finite set of *actions*,
- Ω is a finite set of *observations*,
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Pi(\mathcal{S})$ is the *transition model*,
- $\mathcal{O} : \mathcal{S} \times \mathcal{A} \to \Pi(\Omega)$ is the *observation model*,
- $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the *reward function*,
- $\gamma \in [0, 1)$ is the *discount factor*.

$\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}$, and γ are defined as in the MDP framework. The set of observations Ω and the observation model $\mathcal{O}$ are introduced because in POMDPs the state is only partially observable and the agent can acquire at each step only observations in Ω. The stochastic relationship between states and observations is modeled by $\mathcal{O}$. In particular, $\Pi(S)$ and $\Pi(\Omega)$ define the space of probability distribution over states and observations, respectively. In Figure 2.1, we schematically show the agent-environment interaction for both MDPs and POMDPs. The agent's goal, as in a MDP (Sutton and Barto, 2018; Russell and Norvig, 2010), is to maximize the *expected discounted return*

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right] \tag{2.4}$$

acting optimally (i.e. choosing, in each state s_t, at time t, the action a_t with the highest expected reward). In the POMDP framework, however, the agent is not able to directly observe the current state s_t but it maintains a probability distribution over states S, called *belief* which updates at each time-step by using observations received from the environment. In the following, we represent by symbol $b(s)$ the probability of being in state s according to belief b, namely $b'(s') = Pr(s'|o, s, b)$. The belief summarizes agent's previous experience, i.e. the sequence of actions and observations that the agent took from an initial belief b_0 to the belief b. The sequence of actions and observations up to discrete time-step t is called *history* (h) and is represented as $h = \langle a_0, o_0, ..., a_t, o_t \rangle$. With hao, we denote that, after history h, the agent performs action a receiving observation o. The solution of a POMDP is an optimal or approximated *policy*, namely, a function that maps belief states into actions, i.e. π: $B \to A$, where B is the belief space. A policy is optimal if it maximizes the expected discounted return. The value function for a history h, i.e., $v_\pi(h)$, specifies the expected return obtained following the given policy π from h. Computing the *optimal value function*, $v^* = \max_\pi v_\pi(h)$, it is possible to generate an optimal policy π^*.

2.3 Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) (Browne et al., 2012) is an online algorithm for choosing optimal actions in MDPs. It combines the scalability of sample-based methods with the computational efficiency of approaches based on tree search. This method takes random samples in the search space and builds a search tree according to the results. The strength of MCTS consists of not exhaustively exploring the search space but focusing instead on the most promising sub-spaces, which are determined by using rewards computed at the end of conducted simulations. More precisely, MCTS progressively builds the search tree by incrementally exploring the environment using simulation of executions starting from the root node and descending the tree guided by the results of previous descents, i.e., based on the estimated rewards of the actions taken into account. MCTS relies on the following key concepts:

- the true value of an action may be approximated by random simulation,
- action values may be used to effectively adjust the policy towards a best-first strategy.

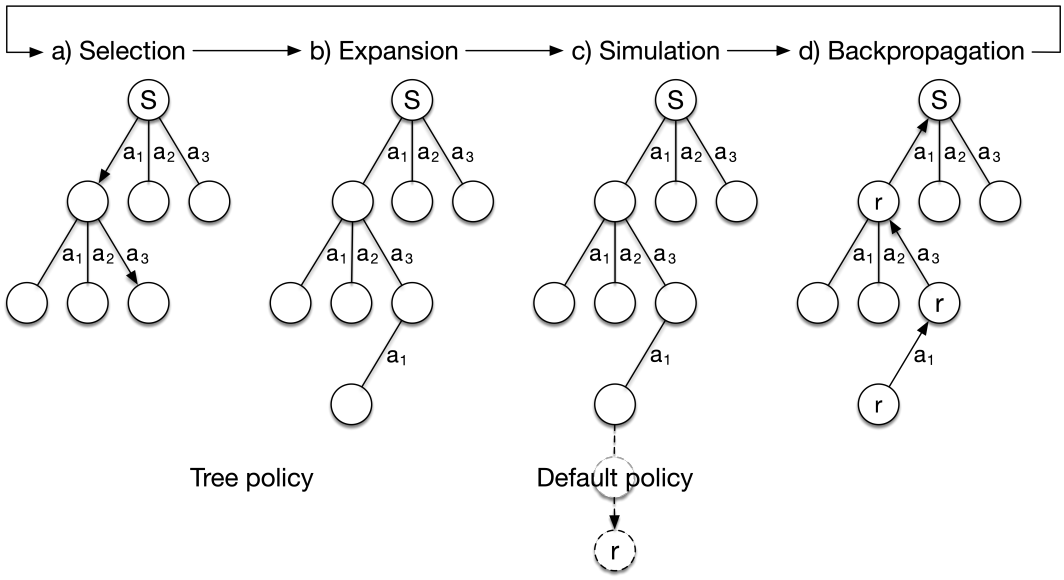

Figure 2.2: Schematic representation of MCTS inspired from Browne et al. (2012). S represents the starting node, $\{a_1, a_2, a_3\}$ is the action set, and r is the reward for the terminal state.

The main data structure used in the algorithm is a tree that initially contains only the root node. Each node of the tree represents a state of the domain, while directed links represent actions that lead to a particular state. The MCTS algorithm repeats a sequence of four steps until a predefined computational bound is reached, e.g. time, memory, or iteration constraints (Chaslot et al., 2008).

1. *Selection*: the *tree policy* is recursively applied starting from the root node to descend the tree and determine the best node to expand (Figure 2.2a). The tree policy estimates the utility value of nodes within the existing search tree. The node with the highest estimated value is selected to expand.

2. *Expansion*: it expands the selected node with at least a child node chosen among the available actions (Figure 2.2b).

3. *Simulation*: a *default policy* is applied to simulate a run from the newly expanded node and produce a reward. The default policy manages the simulation from a given non-terminal state until a stop condition is met, then estimates a reward accordingly (Figure 2.2c).

4. *Backpropagation*: it propagates the reward from the expanded node up to the root node and updates the statistics describing at least the reward value and the number of visits of the nodes traversed during the previous descent (Figure 2.2d).

The simulation step uses the model of the environment to generate the trajectories on which the reward values, and therefore the policy, are computed. Thus, it is important to use an accurate model of the environment to obtain a high level of accuracy in the computation of the reward values and, as a consequence, of a good policy. Algorithm 1 summarizes the schematic representation of MCTS in pseudocode. At each iteration, the tree is descended (line 3) and expanded with at least a new node (line 4). Then, the algorithm estimates the reward obtained by running a simulation process from the recently added nodes (line 5) and, finally, it updates the statistics of the nodes visited during the previous descent by backpropagating the reward value (line 6). When a predefined computational limit is reached, the algorithm ends and returns the best-estimated action to perform (line 8). In Chapter 5, we present an approach to improve an approximated model of the environment, which will be used by MCTS to compute optimal policies.

Algorithm 1: Monte Carlo Treee Search (Browne et al., 2012)

Data: s: start state;
b: computational budget
Result: Best action

```
1 root ← MAKENODE(s)
2 while available b do
3 |   n_s ← SELECTION(root)
4 |   n_e ← EXPANSION(n_s)
5 |   r ← SIMULATION(n_e, r)
6 |   BACKPROPAGATION(n_e, r)
7 end
8 return a(BESTCHILD(root))
```

2.3.1 Upper Confidence Bound applied to Trees

The effectiveness of MCTS relies on the implementation of both the default and tree policies. The default policy is typically tailored to the specific application, as a simulation has to follow environment rules and constraints, whereas the tree policy can be domain-agnostic in many scenarios. Modeling the selection step as a multi-armed bandit problem, the value of a node corresponds to the expected reward estimated in the simulation step, i.e., to random variables with unknown distributions. Thus, Kocsis and Szepesvári (2006) propose the simplest Upper Confidence Bound (UCB) (Auer et al., 2002) policy as the tree policy, namely

$$UCB1 = \overline{X_i} + \sqrt{\frac{2 \ln N}{N_i}} \tag{2.5}$$

where N_i is the number of times arm i was played, N is the number of plays so far, and $\overline{X_i}$ corresponds to the average reward of arm i. The UCB strategy is used to address the *exploration-exploitation dilemma* (Auer et al., 2002), that is, finding a trade-off between selecting actions that seem optimal (*exploitation*) and actions believed to be sub-optimal (*exploration*) but may be superior in the long run. The first term favors the exploitation of choices associated with higher rewards, while the second term encourages to explore less visited choices, therefore with more uncertainty about their rewards. In the selection step, the tree is descended by choosing the nodes that maximize the Upper Confidence Bounds applied to Trees (UCT) (Kocsis and Szepesvári, 2006), namely,

$$UCT = \overline{X_i} + 2C_p \sqrt{\frac{2 \ln N}{N_i}} \tag{2.6}$$

where N is the number of visits of the current node (i.e., the parent node), N_i is the number of visits of the child node i, $\overline{X_i}$ corresponds to the average reward of the child node i, and $C_p > 0$ is the *exploration constant*, whose value determines the amount of exploration to perform. Previously unvisited child nodes are assigned the highest possible UCT value to guarantee at least one visit to all children before a node at the same level is expanded further. As each node is visited, the denominator of the exploration term increases leading to a decrease in its contribution. In contrast, if another node at the same level is visited, the denominator increases and consequently the contribution of the less visited sibling nodes also increases.

As shown in Kocsis and Szepesvári (2006), UCT presents two fundamental properties:

- the bound on the regret of UCT still holds with non-stationary reward distributions,
- the probability of choosing a sub-optimal action at the root level of the tree converges to zero at a polynomial rate as the number of simulations grows to infinity. Thus, with enough time and memory, UCT enables MCTS to converge to *minimax* and, consequently, to be optimal.

The use of UCT in MCTS helps make the algorithm efficient in searching high-dimensional domains. Indeed, the tree grows asymmetrically by focusing on the most promising branches (i.e., those leading to higher rewards) instead of performing an exhaustive search.

2.4 Partially Observable Monte Carlo Planning

Partially Observable Monte Carlo Planning (POMCP) (Silver and Veness, 2010) is a Monte Carlo based algorithm for planning in partially observable environments that combines MCTS, to compute an approximated policy, with a *particle filter*, to represent the belief. At each step, POMCP uses a MCTS to find the best action to perform. The MCTS is generated by iteratively *i)* sampling a particle from the particle filter, *ii)* performing a simulation with the state that corresponds to the sampled particle, according to the transition and observation models known by the agent. The UCT strategy is used to determine the subtrees to explore and to balance exploration and exploitation in the simulation phase. The reward of each simulation is backpropagated in the tree to compute the approximated values for the current belief and, at the end of the process, the action with the higher value is selected. After the selected action is performed in the real environment, a real observation is collected and particles in the belief are updated by keeping only particles that explain the observations. Particle reinvigoration is used if no more particles are available in the particle filter. Algorithm 2 summarizes POMCP in pseudocode.

Algorithm 2: Partially Observable Monte Carlo Planning (Silver and Veness, 2010)

Function SEARCH(h)
  Data: h: history
  Result: a: action
  while not TIMEOUT() **do**
    if h *empty* **then**
      s $\sim b_0$
    else
      s $\sim b(h)$
    end
    SIMULATE$(s, h, 0)$
  end
  return $\operatorname{argmax}_b V(hb)$

Function ROLLOUT(s, h, d)
  Data: s: state, h: history, d: depth
  Result: r: reward
  if $\gamma^d < \epsilon$ **then**
    return 0
  end
  $a \leftarrow \pi_{Rollout}(h)$
  $(s', o, r) \sim \mathcal{G}(s, a)$
  return $r + \gamma$ROLLOUT$(s', hao, d+1)$

Function SIMULATE(s, h, d)
  Data: s: state, h: history, d: depth
  Result: r: reward
  if $\gamma^d < \epsilon$ **then**
    return 0
  end
  if h not in $Tree$ **then**
    forall $a \in \mathcal{A}$ **do**
      $Tree(ha) \leftarrow (N_{init}(ha), V_{init}(ha), \varnothing)$
    end
    return ROLLOUT(s, h, d)
  end
  $a \leftarrow \operatorname{argmax}_b V(hb) + C_p\sqrt{\frac{\log N(h)}{N(hb)}}$
  $(s', o, r) \sim \mathcal{G}(s, a)$
  $r \leftarrow r + \gamma$SIMULATE$(s', hao, d+1)$
  $b(h) \leftarrow b(h) \cup s$
  $N(h) \leftarrow N(h) + 1$
  $N(ha) \leftarrow N(ha) + 1$
  $V(ha) \leftarrow V(ha) + \frac{r - V(ha)}{N(ha)}$
  return r

POMCP presents two significant features. First, the use of Monte Carlo sampling allows POMCP to break both the curse of dimensionality, in which the dimensionality of the planning problem directly relates to the number of states (Pineau et al., 2006; Kaelbling et al., 1998), and the curse of history, in which the number of possible POMDP histories (namely, action-observation traces) increases exponentially with the planning horizon (Pineau et al., 2006). Second, instead of requiring explicit probability distributions, POMCP uses a black-box simulator of the environment. These two characteristics make POMCP effective for planning in large POMDPs. In small state spaces, it is feasible to compute the exact belief update by using Basyes' theorem (Ross et al., 2008), while in large state spaces, such as in real-world problems, this proves to be intractable. Thus, to overcome this limitation, POMCP online builds a search tree of histories, where each node estimates the value of a history through Monte Carlo simulation. These simulations involve sampling the start state from the current unweighted particle filter, used to approximate the belief with each particle representing a possible state, and sampling state transitions, observations, and rewards from the black box simulator to update the belief. In Chapter 4, we present an approach for learning environment properties in partially observable environments during the execution of the standard POMCP algorithm.

2.4.1 POMCP Algorithm

The POMCP algorithm initializes the particle filter with k particles, each representing a state s, following an initial distribution b_0 that corresponds to a uniform distribution if no prior knowledge is available about the initial state (Algorithm 2, line 4). At each step, POMCP samples a state from the particle filter (Algorithm 2, line 6), and performs a Monte Carlo simulation starting from that state (Algorithm 2, line 8). A simulation consists in a sequence of actions and observations whereby the agent collects a discounted return. If a new history is encountered during the simulation, the value of an action in the history is estimated using a rollout policy (Algorithm 2, lines 22-27), otherwise, the simulation process chooses an action and updates the node statistics based on the results of the black box simulator $\mathcal{G}$ (Algorithm 2, lines 28-35). More precisely, the expected value of each action is approximated by computing the average discounted return of all simulations starting with that action, following a back-propagation procedure (Algorithm 2, line 34). Once the simulation ends, POMCP determines the best action based on the value function V in the current history and performs it in the real-world.

2.4.2 Belief update

The particle filter is updated after executing an action in the real environment and collecting an observation, keeping in the filter only the particles that explain the observation returned from the environment. More precisely, as we can see in Figure 2.3, the particle filter is updated after executing an action in the real environment, e.g., action a_1, considering the particles in

the tree branch related to the observation received from the environment, e.g., observation o_3. To avoid particle depletion, POMCP can leverage a particle reinvigoration method to generate new particles from the current belief when the particle filter does not present enough particles to approximate the true belief.

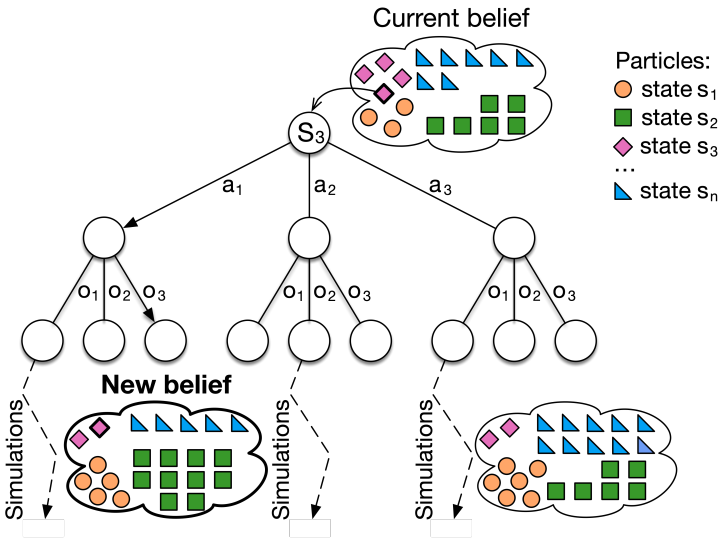

Figure 2.3: Graphical representation of the belief update process.

2.4.3 Partially Observable UCT

POMCP determines the action to perform by using an extension of UCT for dealing with partially observable environments, named Partially Observable UCT (PO-UCT) (Silver and Veness, 2010). This approach builds a search tree of histories instead of states, and each node stores the expected value function $V(h)$, i.e., the average return of all simulations started with h and the number of visits $N(H)$ of the history h. As in the fully observable UCT, the action to perform is chosen to maximize the function

$$V_{UCT} = V(ha) + C_p\sqrt{\frac{\log N(h)}{N(ha)}} \tag{2.7}$$

where $V(ha)$ is the average reward obtained performing action a in history h, C_p is the exploration constant, $N(h)$ is the number of simulations for the current history and $N(ha)$ is the number of simulation in which action a has been performed in history h (line 28).

2.4.4 POMCP with state-variable relationships

A method to introduce prior knowledge in POMCP has been presented in Castellini et al. (2019b). This approach allows the definition of probabilistic equality relationships among pairs of discrete state-variables using Markov

Random Fields (MRFs) that allow for the factorization of the joint probability function of state-variable configurations, which, in turn, is used to constrain the state space. In Chapter 4 this work has been extended with methods for learning the MRF.

2.5 Markov Random Fields

A Markov Random Field (MRF) (Murphy, 2012; Bishop, 2006) is an undirected graph representing a factorization of a probability distribution. Nodes represent discrete stochastic variables, and edges represent probabilistic relationships between variable values. According to the Hammersley-Clifford theorem (Upton and Cook, 2008), the joint probability represented by the MRF can be computed as the product of potential functions over the maximal cliques of the graph (Murphy, 2012). A potential function is a non-negative function of its arguments representing the relative "compatibility" of different variable assignments. Then, by having discrete variables, potentials can be represented as tables of non-negative values. As an undirected graph does not present any topological ordering, the chain rule of probability, which is a repeated application of the product rule, cannot be used for representing the joint distribution. Therefore, potential functions are associated with each maximal clique in the graph. A maximal clique is a subset of nodes in which every pair of nodes is connected by an edge, and no additional nodes can be added to the set without losing the clique property. The probability distribution is therefore represented by the MRF as the formula:

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = \frac{1}{\mathcal{Z}(\boldsymbol{\theta})} \prod_{c \in C} \psi_c(\boldsymbol{x}_c|\boldsymbol{\theta}_c), \tag{2.8}$$

where $\boldsymbol{x}$ is a variable configuration (e.g., $\boldsymbol{x} = (1, 0 \ldots, 0)$), $\boldsymbol{\theta}$ is a parametrization of the MRF (i.e., a specific set of values for the parameters θ that represent the MRF), C is the set of maximal cliques, $\psi_c(\boldsymbol{x}_c|\boldsymbol{\theta}_c)$ is the potential function, and $\mathcal{Z}(\boldsymbol{\theta})$ is the partition function, i.e., a normalization factor that can be computed as

$$\mathcal{Z}(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \prod_{c \in C} \psi_c(\boldsymbol{x}_c|\boldsymbol{\theta}_c). \tag{2.9}$$

Potentials can be represented by a Boltzmann distribution, i.e., exponentials, thus

$$\psi_c(\boldsymbol{y}_c|\boldsymbol{\theta}_c) = exp(-F(\boldsymbol{x}_c|\boldsymbol{\theta}_c)) \tag{2.10}$$

where $F(\boldsymbol{x}_c) > 0$ is the energy function. It has observed a correspondence between high probability states and low energy configurations. Models presenting this characteristic are named energy-based models.

Restricting the parametrization of the MRF to the edge rather than to the maximal clique of the graph, we obtain a *pairwise MRF* and, consequently, the product of potentials can be computed by summing the energies of all pairwise relationships. We call E the set of pairwise relationships (i, j) in the

MRF, where $i, j \in 1, \ldots, n$, and n is the number of state-variables. In Chapter 4, we present an extension of Castellini et al. (2019b), whose goal is to learn probabilistic equality relations between pairs of discrete state variables, which results in learning the potentials of a pairwise MRF.

2.6 Artificial Neural Networks

An Artificial Neural Network (ANN) (Goodfellow et al., 2016) is a computational model, namely a function $f_\theta(x)$ defined over parameters θ, that maps an input x into an output y. An ANN consists of information-processing units, *neurons* or *nodes*, interconnected by links, *synapses*, characterized by a real-valued *weight* representing their strength. More precisely, observing the graphical representation of a neuron in Figure 2.4, we define $w_{i,j}$ as the weight associated with a link from node i to j propagating the activation a_i from i to j. Each node also presents a "dummy" input $a_0 = 1$ whose corresponding weight is $w_{0,j}$. The set of weights is conventionally referred to as θ, i.e., the network parameters. Each *node* j aggregates its input signals by computing a weighted sum

$$in_j = \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} w_{i,j} a_i \tag{2.11}$$

and applies to the resulting one a nonlinear function, the *activation function* g, producing node's output

$$a_j = g(in_j) = g(\sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} w_{i,j} a_i). \tag{2.12}$$

The activation function parameters are the weights of network's connection. In Figure 2.5, we provide a graphical representation of three examples of common activation functions:

- Sigmoid, or logistic, defined as $Sigmoid(x) = \frac{1}{1+e^{-x}}$;
- Rectified Linear Unit (ReLU), defined as $ReLU(x) = \max(0, x)$;
- Hyperbolic tangent, or tanh, defined as $tanh(x) = \frac{e^{-2x}-1}{e^{-2x}+1}$;

Once the activation functions for neurons are determined, there are essentially two ways to connect neurons and form a neural network. The first method is to create connections in a single direction, without loops, thus obtaining a directed acyclic graph. This type of network called a Feed-forward Neural Network (FNN), represents a function of its current input, so the internal state coincides with the weights themselves. On the other hand, the second method creates bi-directional connections in which previous node outputs can be used as inputs of the node itself. In this type of network, known as a Recurrent Neural Network (RNN), the way the network responds

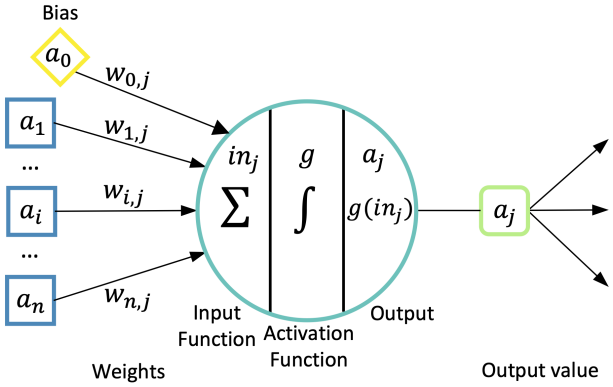

Figure 2.4: Graphical representation of a neuron.

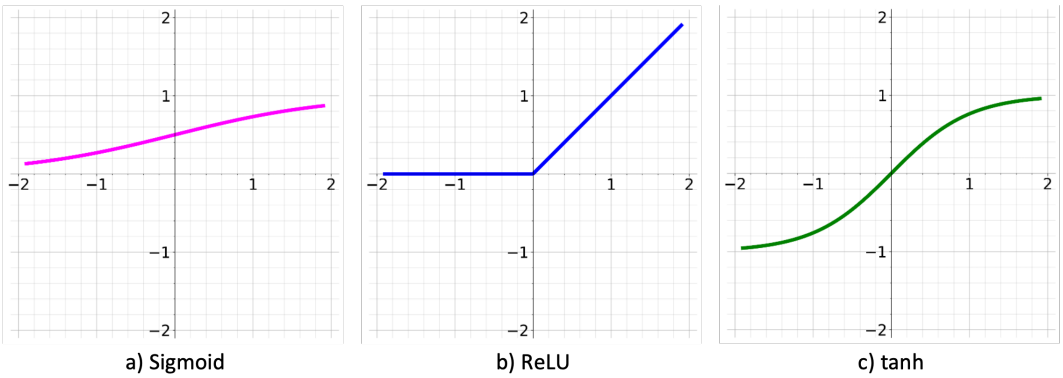

Figure 2.5: Graphical representation of three common activation functions: a) Sigmoid, b) ReLU, and c) tanh.

to a given input relies on its initial state, which, in turn, might depend on previous inputs.

Typically, to learn the weights, an ANN uses Gradient Descent (GD) (Ruder, 2016) methods (e.g., Stochastic Gradient Descent (SGD) (Nemirovski and Yudin, 1978), that adjust the direction of each weight to improve the overall network performance according to the minimization, or maximization, of an objective function. Several variants and improvements over GD have been proposed, such as Adam (Kingma and Ba, 2014), an algorithm for optimizing stochastic objective functions using first-order gradients, which relies on computing adaptive learning rates for each parameter. In supervised learning, a typical function used as a *loss* function L and computed over labeled training data is the expected error, i.e., the expected difference between the observed output and predicted output of the neural network. A crucial characteristic required for the loss function is the differentiability over the network parameters. To provide an insight into how GD works, let us consider Mean Squared Error (MSE) as the loss function to compute the distance be-

tween the output y and the desired output $\hat{y}$

$$L(f_\theta(x), \hat{y}) = \frac{1}{2}\sum_{i=1}^{n}(y^i - \hat{y}^i)^2 \tag{2.13}$$

GD aims to minimize the function L by iteratively updating network parameters θ in the opposite direction of the gradient of L. Formally, at step k, the algorithm updates the parameters as follows

$$\theta_{k+1} = \theta_k - \alpha \nabla_{\theta_k} L(f_k(x), \hat{y}) \tag{2.14}$$

In Chapter 5, we use Multi-Layer Perceptron (MLP) networks, i.e., FNNs organized in layers to represent dynamic models in the context of MCTS. An MLP has several layers of input nodes connected unidirectionally, such that each node receives input only from nodes belonging to the immediately preceding layer (see Figure 2.6 for a high-level representation of the structure). In the proposed methodology, we use one of the common activation functions, namely ReLU.

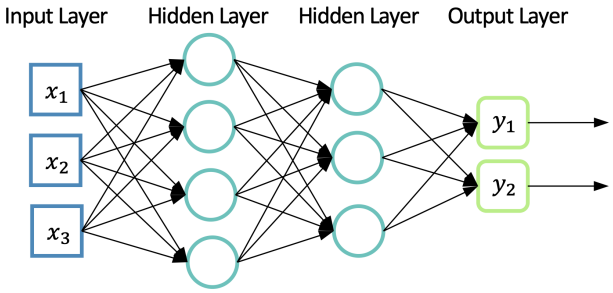

Figure 2.6: High-level representation of a FNN with three inputs, two hidden layer and 2 outputs. The biases are not indicated in this simplified illustration.

2.7 Benchmark domains

This section describes the state-of-the-art benchmark domains used to evaluate the proposed methods in the experiments discussed in Chapter 4.

2.7.1 Rocksample

Rocksample (Smith and Simmons, 2004) is a benchmark domain for POMDP solvers inspired by robotic planet exploration. In rocksample an agent moves through a grid containing valuable and valueless rocks placed in fixed position with the aim to maximize the discounted reward collecting rock values. We perform our tests on rocksample(5,8), consisting of a 5×5 grid in which we pose 8 rocks (see Figure 2.7a). The rock value configuration changes at each episode and it is decided a priori in order to reflect specific constraints. Notation (i, j) identifies the cell in column i and row j on the grid while for

rocks we use indices from 1 to 8. The agent (light blue circle in Figure 2.7a) knows the rock locations but it cannot observe rock values (which is the hidden part of the state). These values can only be inferred using observations returned by the environment. The correct result of rock observations, however, is inversely proportional to the distance between the agent position and the rock. At each step, the agent performs one action among $moving$ (up, down, left, right), $sensing$ a rock (i.e., checking its value) or $sampling$ a rock (i.e., collecting its value). The reward obtained by moving and sensing is 0, while sampling a rock gives a reward of 10 if the rock is valuable, -10 if it is valueless. This problem can be formalized as a POMDP.

- In the $state$ space $\mathcal{S}$, each state is characterized by i) the agent position on the grid, ii) the rocks configuration (hidden), and iii) a flag indicating rocks already sampled.
- $\mathcal{A} = \{up, down, left, right, check_{1,\dots,8}, sample\}$. The set of $actions$ is composed by the four moving actions, the sample action and a sensing action for each rock. Since we aim at maximizing the information learned about state-variable relationships, we prevent the agent from exiting the grid.
- $\mathcal{O} = \{1, 2, 3\}$. $Observations$ have three possible values, namely: 1 for *valuable* and 2 for *valueless* rock observation returned by sensing actions, 3 for *null* observations returned by moving actions.
- The $transition$ function $\mathcal{T}$ is deterministic for moving and sampling actions, while sensing a rock does not change the state.
- The $reward$ function $\mathcal{R}$ returns 0 for moving and sensing actions, 10 for sampling a valuable rock, and -10 for sampling a valueless rock. If the agent tries to exit the grid, $\mathcal{R}$ returns -100.
- The $discount factor$ used is $\gamma = 0.95$.

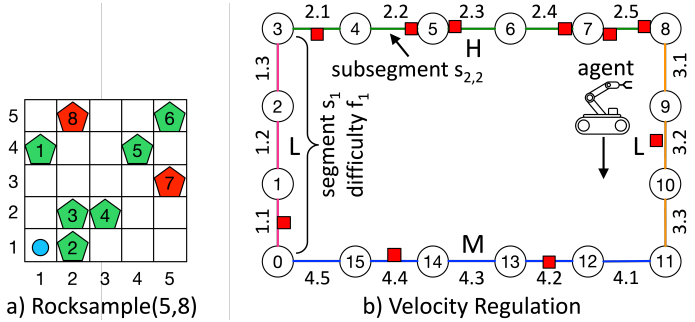

Figure 2.7: Instances of benchamark domains used in the experiments discussed in Chapter 4.

2.7.2 Velocity regulation

In the velocity regulation problem (Castellini et al., 2021b, 2020c) a mobile robot traverses a pre-defined path (see Figure 2.7b) divided into segments z_i and subsegments $z_{i,j}$. Notation (i, j) identifies the position of the robot in the path, where i is the index of the segment and j the index of the subsegment. More precisely, with (i, j) we mean that the agent is at the beginning of subsegment $z_{i,j}$. Each segment is characterized by a difficulty f_i that depends on the obstacle density in the segment. The robot has to traverse the entire path in the shortest possible time, tuning its speed v to avoid collisions with obstacles. Each time the robot collides a time penalty is given. The robot does not know in advance the real difficulty of the segments (which is the hidden part of the state) and it can only infer their values from the readings of a sensor (see Figure 2.7b). This problem can be formalized as a POMDP.

- In the $state$ space $\mathcal{S}$, each state is characterized by i) the position of the robot in the path $p = (i, j)$, where i and j are the indexes of the segment and the subsegment, ii) the (hidden) true configuration of segment difficulties $(f_1, \ldots, f_m)$, where $f_j \in \{L, M, H\}$, L represents low difficulty, M medium difficulty, H high difficulty, iii) t is the time elapsed from the beginning of the path.

- $\mathcal{A} = \{S, I, F\}$. The set of *actions* is composed by the three possible speed values of the robot in a subsegment, namely, slow (S), intermediate (I) or fast (F).

- $\mathcal{O} = \{0, 1\}$. *Observations* are related to subsegment occupancy. The *occupancy model* $p(oc|f)$ probabilistically relates segment difficulties to subsegment occupancy, $oc = 0$ means that no obstacles are detected in the next subsegment, on the contrary, $oc = 1$ means that some obstacles are detected. The observation model provides the probability of observations given segment difficulties, namely $p(o|f)$. We refer to the original work on the velocity regulation problem for more details about specific parameters (Castellini et al., 2021b).

- The *transition* function $\mathcal{T}$ is deterministic as the agent always moves to the next subsegment.

- The *reward* function here is $\mathcal{R} = -(t_1 + t_2)$, where t_1 is the time required to traverse a subsegment and t_2 is the penalty due to collisions (in our tests $t_2 = 10$). The time required to traverse a subsegment depends on the action that the agent performs and the time penalty it receives. Namely, the agent needs 1 time unit if the action is F (fast speed), 2 time units if the action is I and 3 time units if the action is S. The *collision model* $p(cl|f, a)$ regulates the collision probability, more precisely $cl = 0$ means no collision and $cl = 1$ that a collision occurs.

- The *discount factor* we used is $\gamma = 0.95$.

An example of parameters are summarized in Table 2.1.

f	$p(o = 1 \mid f)$
L	0.0
M	0.5
H	1.0

a)

f	a	$p(cl = 1 \mid f, a)$
L	S	0.0
L	I	0.0
L	F	0.0
M	S	0.0
M	I	0.5
M	F	0.9
H	S	0.0
H	I	1.0
H	F	1.0

b)

Table 2.1: Example parameters for the velocity regulation domain. a) Observation model $p(o|f)$: probability of observing subsegment occupancy given segment difficulty. b) Collision model $p(cl|f, a)$: collision probability given segment difficulty and action.

Chapter 3

Related Work

In this Chapter, we discuss the main research areas related to the methodologies presented in the thesis. RL (Sutton and Barto, 2018) can be categorized in model-free (Kaelbling et al., 1996) and model-based (Moerland et al., 2023; Luo et al., 2022). First, in Section 3.1, an overview of model-free RL approaches is provided, then in Section 3.2, we dive deep into model-based RL techniques, which are the main focus of this work. Section 3.3 outlines connections between probabilistic planning and learning focusing on Monte Carlo methods, while Section 3.4 discusses model learning in MCTS-based planning and Section 3.5 focuses on the application of MCTS-based planning methods to robotic platforms. Finally, Section 3.6 analyses the state-of-the-art about planning under uncertainty, 3.7 focuses on Continual Learning (CL), and Section 3.8 presents the surveys in the literature about the application of RL to environmental sustainability.

3.1 Model-free RL methods

In model-free RL, an agent looks for an optimal policy through direct interaction with the environment and improves its performance based on the information obtained from the explored samples (Dong et al., 2020). These algorithms do not learn a model of the environment (i.e., transition and reward models) but they directly learn the state-action relationship (i.e., the policy). These methods can be applied to domains in which the model is totally unknown. Referring to Figure 3.1, we can categorize model-free methods into *value-based* and *policy-based* depending on the adopted policy optimization strategy.

3.1.1 Value-based

Value-based approaches aim to optimize the value function and then they derive an optimal policy from it. An example of a value-based technique is the well-known Q-learning (Watkins and Dayan, 1992; Watkins, 1989) algorithm. The Q-Learning tabular data structure allows for only a limited number of states to be stored, therefore the algorithm presents limitations related

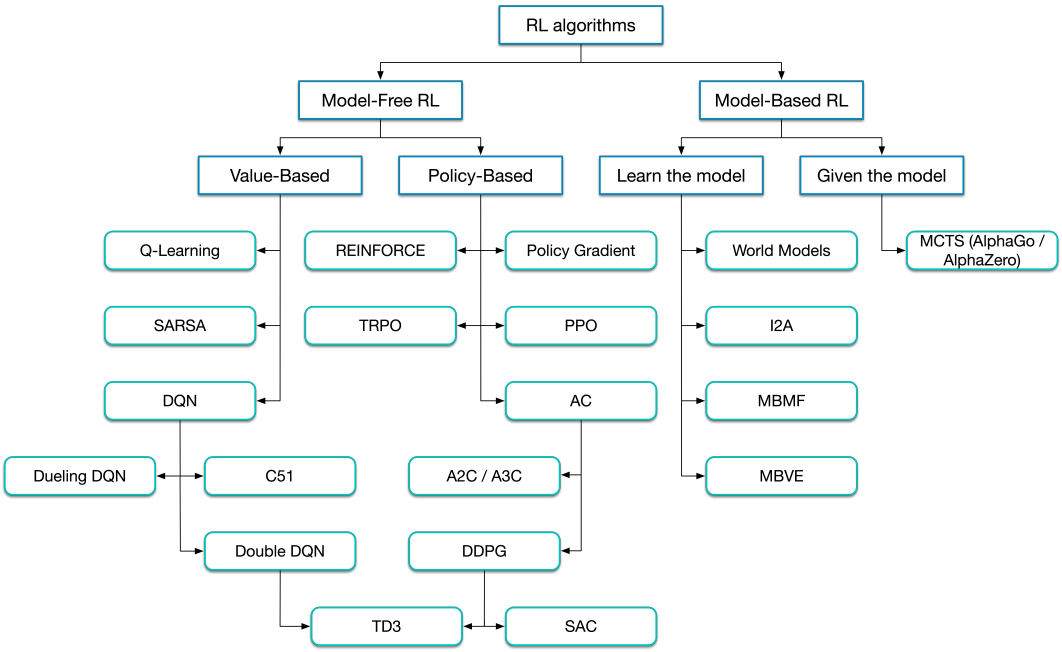

Figure 3.1: A non-exhaustive taxonomy of RL algorithms inspired from Dong et al. (2020). Rectangles denote different categories while squircles denote specific algorithms.

to scalability and continuous action spaces (Castellini et al., 2022b). Another example of a value-based algorithm is Sarsa (Sutton and Barto, 2018), which differs from Q-Learning in the Q-function update process. Deep Q-Network (DQN) (Mnih et al., 2013) belongs to this category of model-free approaches too. This algorithm extends Q-Learning and overcomes its limitations by exploiting a deep neural network to approximate the Q-function, which makes DQN capable of handling complex, high-dimensional environments. Despite its success, this algorithm presents some limitations and has been improved in several ways, such as Double DQN (Hasselt et al., 2016), Dueling DQN (Wang et al., 2016), and Rainbow (Hessel et al., 2018). Value-based approaches are characterized by high efficiency, small variance in the estimation of the value function, and low probability of falling into a local optimum. However, such methods present difficulties in handling continuous action spaces, and a crucial problem relates to determinism at inference time. Moreover, the use of the epsilon greedy strategy and the max operator may lead to overestimation, like in DQN.

3.1.2 Policy-based

Policy-based approaches aim to learn a parameterized policy directly. To this purpose, this class of methods learns the policy's parameter vector θ, such as weights and biases of a neural network, based on the gradient of a scalar performance measure (e.g. the maximum cumulative reward) with respect to θ. REINFORCE (Williams, 1992) can be considered the prototype of Pol-

icy Gradient (PG) algorithms. It consists of a class of methods to update parameters descending the gradient along the direction of the expected reward for maximizing the immediate reward. Several extensions of REINFORCE have been developed to improve the performance of this algorithm, such as REINFORCE-with-baseline (Sutton and Barto, 2018) and Reward-To-Go (Tamar et al., 2016). Policy-based methods can easily apply to continuous action space problems and learn stochastic policies. However, these techniques suffer from low data efficiency and high variance (Wang et al., 2020a).

The *actor-critic* framework has been proposed to overcome these limitations by combining the benefits of both value-based and policy-based approaches, therefore learning the value function (i.e., the *critic*) in addition to the policy (i.e., the *actor*). To better understand the advantage of using an additional neural network, and thus using the actor-critic architecture, it is necessary to introduce the distinction between on-policy and off-policy methods. On-policy methods, like Sarsa, aim to evaluate or improve the policy used for decision-making. This means the agent must interact with the environment using the same policy that it is trying to improve. On the other hand, off-policy approaches, like Q-Learning, evaluate or improve a different policy from the one used to generate data. The off-policy setting can be more efficient, as the agent can improve the policy from the experiences of others. One of the key strengths of actor-critic architectures is that they can combine the advantages of *on-policy* and *off-policy* learning. Specifically, the critic can be trained off-policy, which means that it can learn from any dataset, regardless of how the data was collected, leading to a significant improvement in sample efficiency. On the other hand, the actor can be trained using the policy gradient theorem, which is an on-policy strategy. This allows the agent to learn from its own experience, which can be very effective, as the actor learns directly from the policy that it is trying to improve. Two of the most prominent actor-critic approaches are Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) and Proximal Policy Optimization (PPO) (Schulman et al., 2017). TRPO presents the trust region search technique to ensure that the new policy is not too far from the current policy, limiting the exploration to the safe zone to improve the training performance. To this aim, "KL-Divergence" is used to measure the distance between old and new policies. However, this approach introduces a second-order optimization and significantly increases training overhead. PPO improves such algorithm transforming the method into a first-order optimization by always clipping the ratio between the policies in small intervals.

3.2 Model-based RL methods

Model-based RL combines planning and learning, hence it can be defined as a class of MDP-based approaches that *i)* use a model, namely a form of reversible access to the dynamics of the MDP, *ii)* use learning to approximate a global solution, i.e., a policy or value function (Moerland et al., 2023).

Typically, model-based RL approaches perform two steps: the first consists of learning the dynamic model, then the second uses the learned model to perform for generating the policy. We can categorize model-based RL methods into two groups (see Figure 3.1): those that *learn the model* (and then use it) and those that only use a *given model* (considering it as known). It is important to note that, in model-based RL, the word "learning" has a twofold meaning, as it refers to both learning the model of the dynamics and learning a policy.

3.2.1 Learn the model

In model-based RL, an agent learns the model of the environment by taking actions and observing their outcomes (e.g., the next state and the immediate reward). Learning a model is akin to supervised learning (Jordan and Rumelhart, 1992) and requires establishing the type of model it aims to learn, the approximation method to use, and the region where the model should be valid.

Type of model

Although the model includes both reward and transition functions, we focus on models for transition probabilities between states since it is one of the focuses of this thesis. Given a batch of transition data, the main types of one-step dynamic functions are the following (Moerland et al., 2023). The *forward model* specifies the successor state given the current one and the action to perform. It is commonly used for lookahead planning and it corresponds to the model type we employ in the methodologies presented in Chapters 4 and 5. The *backward/reverse model* defines instead the precursor state and the action that led into a specific given state, thus enabling planning in the backward direction, as in prioritized sweeping (Moore and Atkeson, 1993). Finally, the *inverse model* determines the action to perform for transitioning from the current state to the subsequent one, like in Rapidly-exploring Random Tree (RRT) planning (LaValle, 1998).

Approximation method

Approximation methods can be classified as parametric or non-parametric. Using *parametric approaches* the number of parameters does not rely on the observed dataset size. Exact methods leverage tabular models to store possible transitions, but they are impractical for high-dimensional problems. On the other hand, approximation techniques necessitate a reduced number of parameters and enable generalization, becoming typically favored in large state space problems. Several examples of function approximation methods are linear regression (Parr et al., 2008; Sutton et al., 2008), Dynamic Bayesian Networks (DBNs) (Hester and Stone, 2012), and neural networks (Oh et al., 2015; Narendra and Parthasarathy, 1990), being popular due to their scalability and ability to handle non-linear functions. In the

methodology presented in Chapter 5, we leverage neural networks as an approximation method to learn the dynamics model of MDPs.

Non-parametric approaches directly store and leverage data to represent the model. Such methods are usually not applied to large state space problems due to high data necessity and computational complexity. Exact techniques include Replay buffers (Lin, 1992), a non-parametric version of the tabular transition model, while approximated approaches like Gaussian processes (Deisenroth and Rasmussen, 2011) succeed in generalizing information to similar states.

Model validity region

Concerning model validity region, there are two possibilities: global and local. A *global validity* means the model approximates the dynamics function over the entire space of states. This approach is commonly used in model learning, as in the methodology presented in Chapter 5. On the other hand, *local validity* entails having a local approximate model of the dynamics, discarded after planning over it. This approach narrows the range of approximation functions and may reduce instability but requires continuous estimation of new models from all previous data, which can be infeasible to store entirely.

Challenges

Learning a model entails several challenges, some of which are particularly relevant to the methodologies presented in Chapters 4 and 5.

Stochasticity. The transition function of stochastic MDPs defines a probability distribution over all the possible successor states rather than a single next state. This is challenging, for instance, when training deterministic neural networks on an MSE loss (Oh et al., 2015), as they tend to learn the conditional mean of the next state distribution (Moerland et al., 2017) instead of entire distributions. Approaches to address this issue include descriptive models (e.g., tabular models and Gaussian models (Deisenroth and Rasmussen, 2011)) and generative models (like variational inference (Buesing et al., 2018; Moerland et al., 2017) and Generative Adversarial Networks (GANs) (Yu et al., 2017)).

Uncertainty. Uncertainty due to limited data (namely epistemic uncertainty) is a significant challenge in model-based learning, which can be reduced by collecting more data, while aleatoric uncertainty, (i.e., stochasticity) cannot. Estimating and managing uncertainty is crucial for reliable planning. Frequentist approaches, such as statistical bootstrapping, and Bayesian RL methods, like Gaussian Processes (GPs), have been applied to model estimation (Chua et al., 2018; Fröhlich et al., 2014; Deisenroth and Rasmussen, 2011), with a recent interest in Bayesian methods for neural network approximation of dynamics (Depeweg et al., 2016; Gal et al., 2016).

Partial observability. Partial observability in MDPs occurs when the current observation does not provide complete information about the true state

of the environment. This challenge can be addressed by incorporating information from previous observations by using windowing (Lin and Mitchell, 1992), belief states (Karl et al., 2016; Silver and Veness, 2010; Spaan and Vlassis, 2004), RNNs, and external memory systems like Neural Turing Machines (NTMs) (Graves et al., 2014).

Non-stationarity. Non-stationarity arises within a MDP when the transition and/or reward functions change over time. If an agent relies on its previous model without detecting these changes, its performance can deteriorate rapidly. The primary approach to address non-stationarity is through the use of partial models (Doya et al., 2002), which are ensembles of stationary models, designed to help the agent detect regime switches and switch between models accordingly. Another approach dealing with non-stationarity is given by CL (Lesort et al., 2020). It relies on contexts in which data are not available at once, and they need to be integrated without forgetting existing knowledge.

Another relevant paradigm closely related to non-stationarity is learning under concept drifts (Widmer and Kubat, 1996). Concept drift refers to the phenomenon where the distribution underlying streaming data changes over time in arbitrary ways, i.e., the statistical properties of the target variable the model aims to predict unexpectedly change over time. Detecting, understanding, and adapting to concept drift constitute a framework for learning within environments characterized by concept drifts (Lu et al., 2019a). Stream Learning (SL) aims to learn effectively from continuous streams of data by processing one data point at a time. The system must be able to make predictions at any point in time, dynamically adapt to changes in the data distribution, and remain computationally efficient during both learning and prediction tasks (Bifet et al., 2018). On the other hand, online learning represents a specific case of CL (Lesort et al., 2020; Käding et al., 2017), in which predictive or decision-making tasks are addressed by learning sequentially from individual experiences as they arrive, i.e., one data point at a time.

3.2.2 Given the model

Once the model is known (or estimated), the integration of planning and learning requires addressing the following key concerns.

Determining the initial planning state

There are several approaches by which the initial state of planning can be determined. The first method involves selecting initial states *uniformly* across the entire state space, as seen in Dynamic Programming (Bellman, 1966). However, it does not scale to high-dimensional problems. The second approach chooses the starting state among previously *visited* ones, ensuring planning only in reachable states. Dyna (Sutton, 1990) employs this technique. The third method leverages a *prioritization* over reachable states based on their relevance for planning updates, as in prioritized sweeping

(Moore and Atkeson, 1993). Finally, the use as the initial planning state of the *current* state of the real environment, like in AlphaGo Zero (Silver et al., 2017), emphasizes finding better solutions or gaining more information in the region where the agent is currently active. In the methodologies presented in Chapters 4 and 5, we use the current state of the real environment as the initial state of the planning algorithm.

Deciding the computational resources allocation

Two important aspects of model-based methods include when to start planning and how much planning effort to allocate. The first issue consists in deciding the number of steps to perform in the real environment before starting a new planning process. Several methods perform multiple planning steps after each step in the real environment, such as Dyna, while others collect more data before planning, like Probabilistic Inference for Learning Control (PILCO) (Deisenroth and Rasmussen, 2011), which gathers data over complete episodes and then replans the entire solution once a new set of real transitions has been collected. Similar to MCTS (Browne et al., 2012), the methodologies we present in Chapters 4 and 5 perform a new MCTS-based planning process after each step performed in the real environment. The allocation of planning effort depends on the number of planning iterations to perform, the computational budget each iteration requires, and it also has to consider the computational demands of the entire planning algorithm itself.

Establishing a planning strategy

In deciding how to plan, some key points need to be established. The first two considerations relate to the planning method type and its direction. Discrete planning involves discrete back-ups, stored in trees or tables, for improving value or policy functions. Discrete planning methods, such as one-step search, MCTS, and minimax-search, do not require model differentiability. Instead, the differential planning approach relies on gradient-based methods and requires a differentiable model, such as neural networks, used by PILCO. Then, it has to be determined the planning direction. While forward planning is the default approach in most cases, backward planning, particularly through prioritized sweeping (Moore and Atkeson, 1993), can offer advantages in terms of information propagation over the state space.

The third point is to determine the breadth and depth of planning. Methods with a breadth of one sample single transitions or individual traces from the model and update them by leveraging model-free approaches, such as in Dyna. Planning methods like MCTS adaptively scale the breadth. Finally, full-breadth approaches explore the entire action space before considering a deeper level, as in dynamic programming. On the other hand, when setting the depth to one, planning methods stop after one level of depth (e.g., Dyna). Adaptive depth approaches vary the depth for different parts of the plan (like MCTS) instead. Finally, full-depth techniques sample traces until

the episode ends (such as PILCO). As a planning strategy for the approaches proposed in Chapters 4 and 5, we choose POMCP and MCTS, respectively.

Planning on a learned model involves consideration of model uncertainty. To deal with this issue, an approach consists of Data-Close Planning, which ensures that planning iterations remain within regions where data has been observed as Dyna and Guided Policy Search (GPS) (Levine and Abbeel, 2014) do. Another method is the uncertainty propagation. Explicitly estimating and propagating model uncertainty allows for robust planning over extended horizons. When departing too far from observed data, model uncertainty increases, predictions spread across the state space, and the learning signal diminishes. To propagate uncertainty, one can use an analytical method that fits a parametric distribution to uncertainty at each timestep and analytically propagates such distribution through the model, like in PILCO, or a sample-based approach which instead tracks uncertainty distributions by propagating a set of particles forward, representing predicted distributions at various steps, such as in Chua et al. (2018). The method we propose in Chapter 5 considers the uncertainty of the learned model, hence updates it periodically based on a predefined number of observations collected in the real environment.

Defining the integration of planning within the learning and action cycle

The integration of planning in the learning and acting loop includes considering three key aspects. First of all, directing new planning iterations from learned knowledge. The learned value or policy functions store valuable information about the current environment, which can guide the next planning iteration. Value priors are commonly incorporated through bootstrapping, where the current state or state-action value prediction is used to reduce the search depth of the search in planning, like in Silver et al. (2017) and Moerland et al. (2018). Policy priors can also influence planning, as in GPS and Guo et al. (2014). In particular, AlphaGo Zero (Silver et al., 2017) uses action probabilities as a prior in MCTS planning, enhancing exploration for high-probability actions.

In model-based RL, the ultimate goal is finding a global approximation of the optimal value or policy function. To this aim, the update of such global approximations can be performed by constructing training targets based on the results obtained from the planning process and defining loss functions. Most model-based approaches typically create training targets at the root of the search tree, but they can also be constructed at deeper levels within the tree, providing more information from the planning process. Some approaches combine planned and real data to update value or policy functions, while others exclusively rely on planning for these updates (e.g., in Deisenroth and Rasmussen (2011)), using real data solely for the dynamics model training.

Model-based reinforcement learning can directly choose actions to perform in the real environment from planning results. Several methods use planning solely for selecting actions (e.g. Silver et al. (2008)), while others

combine it with value or policy update process (e.g., Moerland et al. (2018); Silver et al. (2017)). There exist several methods to perform real-world action selection, including greedy selection based on learned models (e.g., in Model Predictive Control (Chua et al., 2018; Nagabandi et al., 2018)) and introducing exploration noise (Silver et al., 2017), and incorporating exploration criteria into the planning process (Dearden et al., 1998). In the methodologies we present in Chapters 4 and 5, we leverage the planning algorithm to compute an optimal policy.

3.3 Connections between probabilistic planning and learning focusing on Monte Carlo methods

Several studies have shown strong connections between probabilistic planning and learning. Both paradigms address similar problems using comparable approaches: estimating the same value functions and incrementally updating these estimates. The main difference lies in the source of experience: learning relies on real interactions with the environment, while probabilistic planning uses simulated interactions, hence a simulator of the environment is assumed to be available. MCTS, which has emerged as a powerful framework for search and planning, presents search mechanics resembling those of sample-based RL approaches, and an overall framework similar to RL in its application to planning problems. While Silver (2009) initially suggested a connection between MCTS and RL, a lack of shared terminology impeded a deeper understanding (Vodopivec et al., 2017).

3.3.1 Similarities between MCTS and RL

This section outlines the similarities between MCTS and sample-based RL techniques by establishing connections between the terminologies used in both fields. In addition, we contextualize the mechanics of the MCTS iteration phases within RL (Vodopivec et al., 2017).

- From an RL perspective, Monte Carlo learning techniques gather experience by episodically sampling the state space. Having a model of the task, simulated experiences can be generated, making these episodes similar to MCTS simulations. Additionally, a trajectory in RL, representing the path taken during an episode, corresponds to the path from the root to a terminal node in MCTS. This gathered experience, including rewards, improves value estimates. RL methods track rewards throughout a trajectory and update the value of visited states based on their total return. In MDP-based methods, credit is assigned to a state or state-action based on the rewards received after visiting that state.

- Updating the value function with new feedback (i.e., backup) is analogous to MCTS backpropagation. Online algorithms perform backups after each time step, while offline algorithms do so at the end of episodes. This mirrors MCTS's multiple backpropagations per iteration or a single

one after the rollout phase. In guiding exploration, RL utilizes a control policy to decide which actions to take and states to visit, mirroring MCTS's tree and default policies. Policies can either maintain action selection probabilities or compute them as required, which is efficient and widely used in both RL and MCTS. RL methods that employ Monte Carlo sampling as a learning approach are considered Monte Carlo control methods, which include MCTS techniques.

- In RL, Generalized Policy Iteration (GPI) (Sutton and Barto, 2018) involves iterating policy evaluation and policy improvement. MCTS operates similarly by making decisions based on previous estimates (policy improvement) under the tree policy during the selection phase and updating these estimates with current feedback during backpropagation (policy evaluation). This allows anytime retrieval of the current policy and value function, similar to GPI, and increasing GPI iterations generally improves solution optimality, as observed in MCTS. Policy improvement involves comparing actions to determine the better ones, either by directly estimating state-actions (Q-values) or by evaluating states (V-values) and assigning actions the value of the resulting states. Both approaches can be implemented in MCTS.

- In RL, the value of a node might be updated at every visit or only at its first visit in an episode. Typically, MCTS algorithms do not track the first visit of a state. A first-visit approach in MCTS would update only the occurrence of the node closest to the root, ignoring other instances. Updating values for every visit can lead to slower convergence, especially in tasks with cycles. RL agents can operate under on-policy control, where they simultaneously evaluate and improve the policy they are following, or under off-policy control, where they adhere to the behavior policy while assessing the target policy. These concepts directly relate to the backpropagation phase of MCTS. In on-policy MCTS, exact feedback values are propagated and an average is maintained for each node, while off-policy MCTS propagates alternative values, such as the maximum value among a node's children.

- TD control methods, which evaluate and improve a policy by bootstrapping TD-learning, are linked to MCTS methods. In MCTS, evaluating nodes by averaging outcomes results in backups equivalent to those of on-policy TD control algorithms, and the four MCTS phases resemble an iteration of Sarsa(1), an on-policy TD control approach. Conversely, backing up the maximum value of child nodes (Coulom, 2007) is similar to off-policy TD methods such as Q-learning (Watkins, 1989).

3.3.2 Differences between MCTS and RL

While many of the mechanics of MCTS can be explained using RL theory, key differences arise in the concepts of rollout and expansion in MCTS (Vodopivec et al., 2017).

- For large-scale tasks where storing the entire state space is impractical, two main approaches are used: approximation and partial memorization. RL often employs value function approximation, although high-quality domain-specific features are challenging to obtain. Domain-specific features can improve performance, although acquiring high-quality features can be challenging. Tabular RL algorithms, which store each value in a single table entry without approximation, are typically inefficient for large-scale tasks. In contrast, search methods like MCTS use tabular storage for only the most relevant parts of the state space, as determined by heuristics or guided by the selection policy. This selective memorization is managed during the MCTS expansion phase.

- MCTS methods adaptively change the state-space representation online, expanding a direct-lookup table (tree or graph) with each iteration. This approach, akin to incremental decision trees (Utgoff, 1989) and adaptive state aggregation (Bertsekas and Castanon, 1989), has been a focus in Machine Learning (ML) for decades. Adaptive representations have a longstanding history in RL and gained prominence with successes like those in Go (Silver et al., 2016). However, RL methods generally do not employ explicit tabular adaptive representations like MCTS methods, which typically memorize and update only a subset of visited states, distinguishing between memorized and non-memorized parts of an episode. As a result, RL does not acknowledge the rollout phase or the use of distinct policies, like the default policy in MCTS.

3.4 Model learning in MCTS-based planning

MCTS planning (Browne et al., 2012) and UCT (Kocsis and Szepesvári, 2006) are approaches that compute the policy online and locally (i.e., only for the states actually visited by the agent). Hence they allow to mitigate the scaling problem using sampling based strategies.

3.4.1 Methods for MDPs

A recent popular result obtained with these approaches is that of AlphaGo (Silver et al., 2017, 2016), a program which defeated a world champion in the game of Go. There exists also extensions of MCTS-based planning to partially observable environments (Zuccotto et al., 2022a; Mazzi et al., 2021b; Castellini et al., 2019b; Silver and Veness, 2010) in which the uncertainty about the state is explicitly represented. To the best of our knowledge, the only attempt to perform model learning in MCTS-based planning on observable environment is that of Bayesian Adaptive Monte Carlo Planning (BAMCP) (Guez et al., 2013). This approach performs Bayesian learning that leads to a significant computational complexity since it considers all possible models according to a belief (i.e., a probability distribution) over their parameters. The technique proposed in Chapter 5 instead performs

standard learning based on a neural network model, providing a simple and sample efficient alternative to BAMCP.

3.4.2 Methods for POMDPs

Since the goal of the methodology we present in Chapter 4 is to learn some information about the environment and to introduce it in POMCP to improve its performance, we also analyzed related works on merging learning and planning, with specific focus on POMDPs and POMCP. Our work is also related, for instance, to Bayesian adaptive learning in POMDPs (Ross et al., 2011). Katt et al. (2017) present an elegant method to learn the transition and reward models in which authors extend the POMCP algorithm to the Bayes-Adaptive case, proposing the Bayes-Adaptive Partially Observable Monte Carlo Planning (BA-POMCP) approach that, however, learns the parameters of the transition model.

The method we propose in Chapter 4, instead, learns probabilities of pairs of state-variables to have equal values in the hidden part of single states (i.e., we do not consider any information about how the state changes over time). We assume that the hidden part of the state can change only from one episode to another and each state has a probability to occur that depends on some (unknown) state-variable probabilistic relationships. We notice that this setting is very common in practice but it cannot be naturally encoded in the transition model. The information encoded in our MRF is instead used to initialize and update the belief. For the same reason, our approach also differentiates from Factored BA-POMDP (Katt et al., 2019), which learns a compact model of the dynamics by exploiting the underlying structure of a POMDP, allowing to better scale to large problems. Even this approach deals with knowledge about the transition from one state to another across the steps of an execution and it cannot learn the probability distribution of states considering probabilistic state-variable relationships, as our MRF does. We remark that we do not factorize the POMDP to learn a compact model of dynamics. We are interested in learning probabilistic relationships between state-variable values, which is an information affecting the initial belief and its update over time.

For instance, the traffic level in two aisles of a warehouse can be highly correlated, hence in an episode the two aisles may have both high traffic levels, and in another episode, they may have both low traffic levels, but the probability that the two aisles have different traffic level in an episode is low. This prior knowledge about the state of the environment, represented by the initial belief into POMDPs, can be naturally integrated in POMCP using the MRF, a generative model that directly represents state-variable relationships. Using the MRF we push the belief probabilities towards states that agree with this knowledge.

In the literature, some works propose approaches to learn arbitrary MRF structures (Vuffray et al., 2020; Pletscher et al., 2009; Salakhutdinov, 2009; Abbeel et al., 2006; Besag, 1977) mainly in the field of computer vision. Due to their generality, these approaches have a higher complexity than the

approach we propose, which is specialized on pairwise MRF for representing state-variable relationships inside POMDPs. In Shah et al. (2021) authors also focus on pairwise MRF but the methodology they propose focuses on learning continuous pairwise MRF. The MRFs that we use in our approach are discrete.

Methodologies for optimally updating POMDP beliefs to reduce uncertainty on the true state have been proposed in Fischer and Taş (2020); Thomas et al. (2020); Ognibene et al. (2019); Spaan et al. (2015); Veiga (2015); Araya et al. (2010); Stachniss et al. (2005). However, these methods mainly focus on introducing the belief into the reward function in order to allow the definition of information gain goals, otherwise not definable, in the context of POMDP. To deal with large environments in practical problems, hierarchical models (Friston, 2008) have been used to extend the POMDP framework (Doshi-Velez, 2009; Sridharan et al., 2008; Theocharous et al., 2004; Pineau et al., 2001; Theocharous et al., 2001). These approaches take advantage of the structure of the problem to decompose the state or the action space, introducing different levels of abstraction, with the aim of learning much larger models. Moreover, in these works the computation of optimal policies is performed considering only a subset of the models or an action subset, since it is intractable to compute optimal policies for the original problem. However, in our approach, we do not decompose the original problem into sub-tasks and we compute policies considering the entire problem domain. Finally, within the research topic of learning for planning in robotic platforms Atrash and Pineau (2010) propose a methodology for learning a model of the user in applications where untrained humans interact and control the robot. Also in this case the goal is to learn a model of the environment.

3.5 Application of MCTS-based methods to robotic platforms

Regarding the application of POMCP to robotic platforms, we noticed that the planning algorithm has been recently applied to different robotic problems. In Goldhoorn et al. (2014) authors propose two extensions of POMCP to find-and-follow people that work in the continuous space and plan actions in real-time. The Adaptive Highest Belief Continuous Real-Time POMCP Follower presented in that paper is aimed to avoid unnecessary turns of the robot in reaching the goal. Our method and ROS-based architecture presented in Chapter 4, instead, aim to learn state-variable relationships and use them in POMCP to improve planning performance. In Giuliari et al. (2021); Wang et al. (2020b) POMCP is used in the context of Active Visual Search. The authors propose a method in which the agent starts acting in an unknown environment (i.e., with no information about the area map). Moreover, they present a new belief reinvigoration approach that deals with dynamically growing state space. In Lauri and Ritala (2016) POMCP has been used to control a mobile robot to explore a partially known environ-

ment. POMCP has been previously integrated with ROS in Wertheim et al. (2020) where a robotic planning platform called ROS-POMDP is presented. It generates the POMDP model of the problem using Performance Level Profiles (PLP) (Brafman et al., 2016) and Relational Dynamic Influence Diagram Language (RDDL) (Sanner, 2010). A two-layer control architecture is instead proposed in Castellini et al. (2021b), where the upper layer uses an extension of POMCP to tune the velocity of a mobile robot and the lower layer uses a standard engine controller to deal with path planning. As explained above, the ROS-architecture that we propose has a completely different goal, namely, allowing to show that the integration of MRF learning and POMCP is possible in real-world robotic applications.

3.6 Planning under uncertainty

Planning under uncertainty is a crucial task for autonomous and intelligent agents. The first works on POMDP-based planning date back to the seventies (Sondik, 1978). Since then several methods have been proposed to solve POMDPs (Walraven and Spaan, 2019; Sunberg and Kochenderfer, 2018; Amato and Oliehoek, 2015; Veiga et al., 2014; Silver and Veness, 2010; Ross et al., 2008). Recent works highlight the benefits of introducing prior knowledge in problems formalized as POMDPs and solved by POMCP. For instance, logic rules have been used to guide POMCP during the online generation of the policy with the aim of improving its performance (Mazzi et al., 2023b), interpreting the POMCP policy (Mazzi et al., 2022; Meli et al., 2022; Mazzi et al., 2020), detecting anomalous actions (Mazzi et al., 2021a), reducing the risk of unsafe actions (Mazzi et al., 2023a), shielding these actions (Mazzi et al., 2021b,c). Castellini et al. (2019b) have shown that the introduction of prior knowledge about state-variable relationships yields performance improvement. In particular, constraints expressed as MRFs (Murphy, 2012) and Constraint Networks (CNs) (Dechter, 2003) have been used. In subsequent work, Castellini et al. (2021b) have shown how mobile robots can exploit prior knowledge about task similarities to improve their navigation performance in an obstacle avoidance context. The main limitation of these works regards the requirement to have a full specification of the prior knowledge in advance, but this is not always feasible in practice, especially in complex application domains such as robotic ones. The methodology from Castellini et al. (2021b, 2019b) differs from ours in that we aim to learn the relationships among state-variables on real robots while acting in the environment and to adapt the related MRF while it is used.

Some other works deal with the problem of adding constraints to planning for improving the performance or scaling to large environments. Lee et al. (2018) use MCTS to generate policies for constrained POMDPs, and Amato and Oliehoek (2015) explore the multi-agent structure of some specific problems to decompose the value function. Instead, in Chapter 4, we constrain the state space on the basis of state-variable relationships to refine the belief during execution. Specifically, we exploit the learned MRF whose po-

tentials express probabilistic constraints between state-variable values. Other related works in the field of planning under uncertainty concern factored POMDPs and their applications (Williams and Young, 2007; McAllester and Singh, 1999). However, our approach is substantially different since the performance improvement does not derive from a factorization of the POMDP but from the introduction in POMDP of prior knowledge on the domain, represented as an MRF learned from previously collected data.

3.7 Continual Learning

CL (Lesort et al., 2020) is an ML paradigm where the model is designed to adapt to changing data distributions and evolving learning objectives over time without having access to all the data at once. A key challenge in this approach is to avoid catastrophic forgetting (French, 1999), meaning the model must continuously acquire new skills while retaining previously learned knowledge. CL strategies can be classified into four categories (Lesort et al., 2020): *i)* dynamic architecture techniques (Mallya and Lazebnik, 2018; Li and Hoiem, 2017; Rusu et al., 2016), involve modifying model architecture to acquire new skills, either by explicitly adding, cloning, or storing model parameters, or implicitly through methods like weight freezing; *ii)* regularization methods (Kirkpatrick et al., 2017; Zenke et al., 2017) utilize additional information to prevent overfitting; *iii)* rehearsal approaches (Hayes et al., 2019; Rebuffi et al., 2017) maintain memory of past tasks by storing raw data; and *iv)* generative replay (Caselles-Dupré et al., 2021; Shin et al., 2017) involves training generative models on data distributions to facilitate learning new tasks while retaining knowledge of previous ones. Replay strategies (Hurtado et al., 2023; Sangermano et al., 2022) include rehearsal and generative replay approaches as sub-categories (Lesort et al., 2020).

Among dynamic architecture techniques, Progressive Neural Networks (PNNs) (Rusu et al., 2016) incrementally add new neural networks, called columns, for each new task while keeping the parameters of previously learned columns fixed. This architecture involves lateral connections between new and existing columns, enabling knowledge transfer from old tasks to new ones. By preserving the old columns intact, PNN effectively prevents catastrophic forgetting, ensuring previously learned representations remain unaltered. On the other hand, Learning Without Forgetting (LWF) (Li and Hoiem, 2017) retains the outputs of the original model on the old task data to preserve the original abilities when a new task is introduced and adds new nodes to the output layer with randomly initialized parameters. During training on the new task, the old task outputs are included in the loss function to ensure the performance does not deviate significantly from its previous state. Finally, PackNet (Mallya and Lazebnik, 2018) trains a neural network on an initial task and then prunes some of the weights that are considered less important, freeing up parameters for subsequent tasks. For each new task, the pruned network is retrained, allowing it to learn the new task while preserving knowledge of the previous tasks through the retained important weights.

Concerning regularization, Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) addresses catastrophic forgetting by incorporating a regularization term in the loss function, that penalizes significant changes in parameters crucial for previously learned tasks. The importance of each parameter is quantified using the Fisher information matrix, which indicates how significant each parameter is to the performance of old tasks. By discouraging modifications to important parameters during new task training, EWC helps maintain the model's performance on previous tasks. Synaptic Intelligence (SI) (Zenke et al., 2017) is similar to EWC but uses a different metric to measure parameter importance. SI accumulates the importance of each parameter based on how sensitive the loss function is to changes in that parameter online during training. This accumulated importance is used to regularize parameter updates when learning new tasks, ensuring that significant parameters are preserved, thereby reducing the risk of catastrophic forgetting.

An example of a rehearsal strategy is Incremental Classifier and Representation Learning (iCaRL) (Rebuffi et al., 2017), which combines classification and representation learning to handle CL scenarios. iCaRL maintains a memory of previous task exemplars, that are used along with new task data during training. It uses a nearest-mean-of-exemplars classifier that computes class prototypes to classify new instances. This method ensures that the model performs well on both old and new tasks by balancing exemplar memory and representation learning. Instead, Exemplar Streaming (ExStream) (Hayes et al., 2019) is an online learning algorithm designed to handle streams of data in real-time. It incrementally updates the model by clustering new data points and merging them with existing clusters. ExStream maintains a fixed number of clusters and dynamically adapts them as new data arrives, ensuring that knowledge from previous tasks is efficiently preserved.

Finally, Generative Replay (GR) (Shin et al., 2017) leverages generative models within the GANs framework to generate synthetic data that, when combined with previous task solver responses, emulate previously learned tasks. When a new task is introduced, the replayed synthetic data generated by the generator-solver pair is integrated with new task data to update both the generator and solver networks. Self-TRIGgered GEnerative Replay (S-TRIGGER) (Caselles-Dupré et al., 2021) uses Generative Replay to retain information about past environments and incorporates an automatic detection mechanism for environment changes based on the error distribution of Variational Autoencoders (VAEs). When faced with a new environment, S-TRIGGER enables the VAE to autonomously initiate Generative Replay without requiring explicit specification of the environment change.

CL shares some similar concepts with our work since training data are not available at once and need to be integrated by keeping the most representative data, as rehearsal approaches do (Lesort et al., 2019; Rebuffi et al., 2017). However, CL algorithms aim at learning new tasks preserving the knowledge of previously learned tasks, considering that old tasks were correctly learned. The model-learning technique for MCTS presented

in Chapter 5, instead starts with an expert's model of the environment which is approximated and partial, and it updates the model in the parts that do not fit the observations taken from the real environment. Furthermore, in our approach model adaptation/update is inserted in a process of policy improvement typical of RL methods. As explained in Chapter 7, in future work, we could compare the performance of some of these approaches with that of the proposed technique and also integrate some CL strategies in it.

3.8 Application of RL to environmental sustainability

The literature provides already some surveys on the application of RL to problems related to environmental sustainability, but all these works focus only on specific aspects of environmental sustainability or they consider also AI methods different from RL. For instance, Ma et al. (2020) focus on Energy-Harvesting Internet of Things (IoT) devices, offering insights into recent advancements addressing challenges in commercialization, standards development, context sensing, intermittent computing, and communication strategies. Charef et al. (2023) conduct a study considering various AI techniques, including RL, to enhance energy sustainability within IoT networks. They categorize studies based on the challenges they address, establishing connections between challenges and AI-based solutions while delineating the performance metrics used for evaluation. Within the domain of Architecture, Engineering, Construction, and Operation, Rampini and Re Cecconi (2022) concentrate on the application of AI techniques, including RL, in Asset Management. Their work reviews studies related to several aspects such as energy management, condition assessment, operations, risk, and project management, identifying key points for future development in this context. Alanne and Sierla (2022) shift their focus to smart buildings, discussing the learning capabilities of intelligent buildings and categorizing learning application domains based on objectives. They also survey the application of RL and Deep Reinforcement Learning (DRL) in decision-making and energy management, encompassing aspects like control of heating and cooling systems and lighting systems. Within the context of smart buildings and smart grids, Mabina et al. (2021) examine the utilization of ML, including RL, for optimizing energy consumption and electric water heater scheduling, emphasizing the advantages of these approaches in Demand Response (DR) due to their interaction with the environment. Himeur et al. (2022) investigate the integration of AI-big data analytics into various tasks such as load forecasting, water management, and indoor environmental quality monitoring, focusing on the role of RL and DRL in optimizing occupant comfort and energy consumption. Yang et al. (2020) focus on the application of RL and DRL techniques to sustainable energy and electric systems, addressing issues such as optimization, control, energy markets, cyber security, and electric vehicle management.

In the realm of transportation systems, Li et al. (2023) explore various topics, including cooperative Mobility-on-Demand systems, Driver Assistance

Systems, Autonomous Vehicles (AVs), and Electric Vehicles (EVs). Sabet and Farooq (2022) study the state-of-the-art in the context of Green Vehicle Routing Problems, which involve reducing greenhouse gas (GHG) emissions and addressing issues like charging activities, pickup and delivery operations, and energy consumption. Moreover, the authors note that most of the works leverage metaheuristics while using RL methods is uncommon. Chen et al. (2019) tackle sustainability concerns within the Internet of Vehicles, leveraging 5th Generation Mobile Network (5G) technology, Mobile Edge Computing architecture, and DRL to optimize energy consumption and resource utilization. Rangel-Martinez et al. (2021) assess the application of ML techniques, including RL, in manufacturing, with a focus on energy-related fields impacting environmental sustainability. Sivamayil et al. (2023) explore a wide range of RL applications (e.g., Natural Language Processing, health care, etc.) emphasizing Energy Management Systems with an environmental sustainability perspective. Mischos et al. (2023) investigate Intelligent Energy Management Systems across diverse building environments, considering control types and optimization approaches, including ML, Deep Learning (DL), and DRL. Yao et al. (2023) discuss the application of Agent-Based Modeling and Multi-Agent System modeling in the transition to Multi-Energy Systems, highlighting RL and suggesting future research directions in Multi-Agent Reinforcement Learning (MARL) for energy systems.

While these works address specific aspects of environmental sustainability using RL methods, the review we conduct in Chapter 6 takes a comprehensive approach, analyzing all contexts in which RL techniques have recently contributed to enhancing environmental sustainability. Our goal is to provide practitioners with insights into state-of-the-art methods for addressing environmental sustainability challenges across various application domains, including energy and water resource management and traffic management. In summary, the survey we conduct in Chapter 6 provides an overview of RL application domains within the context of environmental sustainability.

Chapter 4

Learning State-Variable Relationships in POMCP

In this chapter, we present three methodologies to learn state-variable relationships as the agent acts in the environment to improve the performance of POMCP. In Section 4.1, we outline the motivation and scenarios; Section 4.2 describes an extension of POMCP that introduces prior knowledge in the algorithm by using MRFs; in Section 4.3, we present three MRF learning techniques to compute MRF parameters from agent action outcomes and the belief of the agent; Section 4.4 answers the question "When can the MRF be trusted?" proposing two stop learning criteria, namely a confidence interval-based and a convergence-based stop learning criterion; Section 4.5 describes a method to learn state-variable relationships on a robotic platform; finally, Section 4.6 outlines an approach that adapts online the MRF if a mismatch is detected between the MRF and the true state. Section 4.7 introduces the metrics used to evaluate the performance of the learning approaches; in Section 4.8 we compare the techniques to determine the best MRF learning method and analyze the dependence of the performance on the number of training episodes; Section 4.9 presents the performance of the best learning approach combined with the confidence interval-based stop learning criteria; Section 4.10 reports experimental results of the best learning approach combined with the stopping criterion based on the convergence of MRF potentials on robotic platforms, finally, Section 4.11 concludes the chapter with the performance of the MRF adaptation approach.

Publications on this topic:

- M. Zuccotto, A. Castellini, M. Piccinelli, E. Marchesini, and A. Farinelli. Learning environment properties in partially observable Monte Carlo planning. In Proceedings of the 8th Italian Workshop on Artificial Intelligence and Robotics - A workshop of the 20th International Conference of the Italian Association for Artificial Intelligence (AIRO@AI*IA), volume 3162, pages 50–57. CEUR-WS.org, 2021.

- M. Zuccotto, A. Castellini, and A. Farinelli. Learning state-variable relationships for improving POMCP performance. In Proceedings of the

37th ACM/SIGAPP Symposium on Applied Computing (SAC), pages 739–747. Association for Computing Machinery (ACM), 2022a. doi: 10.1145/3477314.3507049.

- M. Zuccotto, M. Piccinelli, A. Castellini, E. Marchesini, and A. Farinelli. Learning state-variable relationships in POMCP: A framework for mobile robots. Frontiers in Robotics and AI, 9:663–704, 2022b. doi: 10.3389/FROBT.2022.819107

4.1 Motivation and scenarios

Despite recent works propose approximate (Hauskrecht, 2000) and online (Silver and Veness, 2010; Ross et al., 2008) planning approaches, only few (Castellini et al., 2019b; Lee et al., 2018; Amato and Oliehoek, 2015) make use of prior knowledge to improve planning performance. These approaches differ in the way they introduce prior knowledge. In Amato and Oliehoek (2015) authors explore the multi-agent structure of the problem to decompose the value function, while in Lee et al. (2018) cost-constraints are used to solve multi-objective problems and MCTS is used to generate policies for constrained POMDPs. Finally, in Castellini et al. (2019b) authors constrain the state space on the basis of state-variable relationships to refine the belief during execution. The introduction of such prior knowledge has a twofold positive effect, on performance improvement and on a more rapid convergence of the belief state distribution on the real state of the environment (Castellini et al., 2020c, 2019c). These methods, however, need to be provided with the prior knowledge to use, in advance. In our work, we propose a method to overcome this limitation and, given the structure of the problem, learn state-variable constraints expressed by a MRF (Bishop, 2006), as in Castellini et al. (2019b), starting from a non-informative one and refining it while the agent acts.

4.2 POMCP with state-variable relationships

The methodology we use to represent prior knowledge in POMCP has been introduced in Castellini et al. (2019b), and we refer to it as Extended POMCP in the following. It allows to define probabilistic equality relationships among pairs of state-variables through MRFs (defined in Section 2.5). The use of the MRF allows to factorize the joint probability function of state-variable configurations and this probability is used to constrain the state space. Indeed, the MRF defines a probability distribution over states of the POMDP. For instance, in the rocksample domain the state space is the set of all possible rock value configurations and the constraints introduced by the MRF allow to (probabilistically) reduce the possibility to explore states that have small probability to be the true state. The integration of prior knowledge in POMCP is mainly developed in the particle filter initialization and in the

reinvigoration phase (Castellini et al., 2019b), where the probabilistic constraints stored in the MRF are used to optimize the management of the particle filter representing the agent belief. As we can see in Figure 4.1a, given a pair of state-variables $(X_i, X_j)|(i,j) \in E$ representing two rocks in rocksample, the value 0.9 on the edge between X_1 and X_2 represents a state-variable relationship about rock value equality. Then, see Figure 4.1.b, a potential could be $\psi_{X_1,X_2}(0,0) = 0.45$, which indicates a compatibility of 0.45 to have value 0 in both rocks X_1 and X_2, or $\psi_{X_1,X_2}(0,1) = 0.05$, which indicates a compatibility of 0.05 to have value 0 in rock X_1 and 1 in rock X_2. In the following, when we refer to an MRF we mean a set of potentials representing compatibilities of different variable assignments

$$\psi_{X_i,X_j}(l,h),\ (i,j) \in E,\ l,h \in \{1,\dots,k\} \tag{4.1}$$

where k is the number of possible values of each variable.

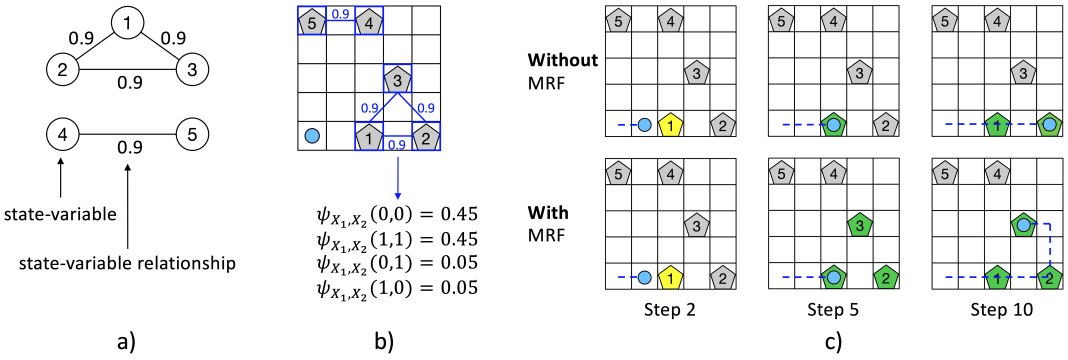

Figure 4.1: Example of MRF and intuition of the advantage given by its usage on performance in the rocksample environment. a) Example of MRF. b) Example of usage of the MRF in the rocksample environment. The MRF topology is depicted using rocks as nodes and equality probability constraints are specified on blue edges between rocks. c) Effect of the action performed by the agent at steps 2, 5, and 10 using standard POMCP (first row) and using the information stored in the MRF (second row). A yellow colored rock means that the rock is checked by the agent, while a green colored rock represents a rock observed to be valuable by the agent. We consider rocks 1, 2, and 3 to be valuable in this example.

To intuitively understand the advantage introduced by the MRF, consider the rocksample environment depicted in Figure 4.1.b, in which the knowledge introduced by the MRF is represented by blue edges between rocks on the grid. The prior knowledge about state-variable relationships is here information about equality relationships among the value of different rocks (e.g., with probability 0.9 rocks 4 and 5 have the same value). We use this knowledge to "push" the belief probabilities towards states that agree with this information during particle filter initialization. The rationale is that if the agent knows that two rocks (i.e., two state-variables) have the same value with high probability (0.9 in Figure 4.1.b), then it can improve its planning

performance because once it has observed the value of one rock, it has also acquired some knowledge about the value of the other rock and it can plan accordingly. In the first row of Figure 4.1.c we show a hypothetical sequence of actions performed by the agent having no knowledge about rock values relationships (i.e. standard POMCP), while in the second row we show a hypothetical sequence of actions performed exploiting such knowledge. In both cases, at step 2 the agent performs a sensing action to check the value of rock 1 (yellow colored). In the hypothesis that the agent observes that rock 1 is valuable (green pentagon in the second column), in the first case (i.e., without MRF) it has no information about rocks 2 and 3, while in the second case (i.e., with MRF) the agent has some information also about rocks 2 and 3, which are considered valuable with high probability (green pentagons). At step 10, exploiting the acquired knowledge about rock values relationships the agent with MRF has already sampled rocks from the three rocks, while the agent without any knowledge has only sampled rocks 1 and 2, hence the agent with the MRF moves faster. We remark that the knowledge in the MRF does not affect the transition model, but only the probability distribution over POMDP states. The knowledge stored in an MRF is used to initialize the particle filter (representing the belief) of POMCP and to update the particle filter (i.e., the belief) during reinvigoration, a procedure used by POMCP to introduce new particles upon depletion.

4.3 MRF learning techniques

We present three different methods to learn the MRF during the execution of standard POMCP. The first, named *Sample-based MRF Learning (SL)* is based on knowledge gathered by the agent when it gets the true value of the state-variables, i.e., using sampling actions in rocksample. In this method we assume that the agent can somehow (e.g., from the reward or a specific action) get the true value of (hidden) state-variables; for instance, in rocksample we assume the agent can see the true value of the rock only when it collects it. We use this information only in the MRF learning algorithm, not to update the belief. The second and third methods, called *Maximum-likelihood Belief-based MRF Learning (MBL)* and *Weighted-likelihood Belief-based MRF Learning (WBL)* are instead based on the information present in the belief at the end of each learning episode. The two methods differ from each other in the way they use the information in the belief to compute the MRF: MBL uses only the state having maximum likelihood, while WBL uses all states in the belief, weighted by their likelihood. In all our tests we assume to learn the MRF in NE episodes, where each episode e is composed of a fixed number of steps. The methodology can simply adapt to the case of episodes with a different number of steps. We initialize the MRF with uninformative priors and then update it at the end of each episode. Beyond the differences between the approaches, it is important to note that they all perform MRF learning offline, updating the MRF after each episode. Details about the three learning methods and common data structures are reported in the following.

4.3.1 Data structures used in the MRF-learning algorithms

Learning the MRF means learning the potentials of pairwise MRF representing state-variable relationships. Given two variables X_i and X_j connected by a pairwise relationship in the MRF and having k possible values each, we learn the potential $\psi_{X_i,X_j}(l,h)$ for each pair (l,h) with $l \in \{1,\dots,k\}$ and $h \in \{1,\dots,k\}$, where variable equality occurs when $l = h$. We keep track of intermediate quantities using three data structures:

- A *vector of state-variable values* $\mathcal{V}_e(i)$, $i = 1,\dots,n$, for each episode e, where n is the number of state-variables and the value in cell i, namely, $\mathcal{V}_e(i) \in \{1,\dots,k\}$ is the value of state-variable X_i observed or extracted from the final belief in episode e. This vector is initialized to $\mathcal{V}_e(i) = 0, \forall i \in \{1,\dots n\}$, and then each value $\mathcal{V}_e(i)$ is updated when the value of variable X_i in episode e becomes available.

- A four-dimensional array of *counts of state-variables equalities and inequalities*, for each episode e, $\mathcal{M}_e(i,j,l,h)$, where $(i,j) \in E$ and $l,h, \in \{1,\dots,k\}$. The value $\mathcal{M}_e(i,j,l,h)$ is the number of times variable X_i had value l and variable X_j had value h in the previous e episodes, where $e \in \mathbb{N}$. Array $\mathcal{M}_e$ is updated at the end of each episode e using the values in $\mathcal{V}_e(i)$, and the MRF potentials $\psi_{X_i,X_j}(l,h)$ are directly computed using values in $\mathcal{M}_e(i,j,l,h)$ (see Equation 4.5), hence the MRF can be updated using values in $\mathcal{M}_e$.

- A matrix of *probabilities of state-variables equalities*, $\mathcal{P}_e(i,j)$, where $(i,j) \in E$. The value $\mathcal{P}_e(i,j)$ is the probability that state-variables X_i and X_j had equal values until episode e (see Equation 4.6).

In summary, our pipeline at each episode e first computes $\mathcal{V}_e$, then $\mathcal{M}_e$, afterwards ψ^e, and finally $\mathcal{P}_e$. What differentiates the three MRF-learning strategies here proposed is how they populate $\mathcal{V}_e$ and how they update $\mathcal{M}_e$ after each episode. In the next sections, we present the three proposed learning algorithms and the related strategies for populating $\mathcal{V}_e$ and updating $\mathcal{M}_e$.

4.3.2 Learning method 1: Sample-based MRF Learning (SL)

In this approach $\mathcal{V}_e(i)$ is the value observed for state-variable X_i when the variable is sampled in episode e. We call this $\mathcal{V}_e^{SL}$. If in rocksample the i-th rock has been sampled and it was valuable, then $\mathcal{V}_e^{SL}(i) = 1$, if it was valueless then $\mathcal{V}_e^{SL}(i) = 2$, and if it was not sampled then $\mathcal{V}_e^{SL}(i) = 0$. $\mathcal{M}^{SL}$ is initialized to $\mathcal{M}^{SL}(i,j,l,h) = 0$ for each $(i,j) \in E$, and $l,h \in \{1,\dots k\}$. After each episode $\mathcal{M}_e^{SL}$ is updated using vector $\mathcal{V}_e^{SL}$ as:

$$\mathcal{M}_{e+1}^{SL}(i,j,l,h) = \begin{cases} \mathcal{M}_e^{SL}(i,j,l,h) + 1, \text{ if } \mathcal{V}_e^{SL}(i) = l \wedge \mathcal{V}_e^{SL}(j) = h \\ \mathcal{M}_e^{SL}(i,j,l,h), \text{ otherwise} \end{cases} \tag{4.2}$$

4.3.3 Learning method 2: Maximum-Likelihood Belief-based MRF Learning (MBL)

In this approach $\mathcal{V}_e(i)$ is populated with the value of state-variable X_i in the state having maximum likelihood in the belief of the agent at the end of episode e. We call this $\mathcal{V}_e^{MBL}$. $\mathcal{M}^{MBL}$ is initialized to $\mathcal{M}^{MBL}(i,j,l,h) = 0$ $\forall (i,j) \in E$, $\forall l,h \in \{1,\dots k\}$. At the end of each episode the array $\mathcal{M}_e^{MBL}$ is updated using vector $\mathcal{V}_e^{MBL}$ as:

$$\mathcal{M}_{e+1}^{MBL}(i,j,l,h) = \begin{cases} \mathcal{M}_e^{MBL}(i,j,l,h) + 1, \text{ if } \mathcal{V}_e^{MBL}(i) = l \wedge \mathcal{V}_e^{MBL}(j) = h \\ \mathcal{M}_e^{MBL}(i,j,l,h), \text{ otherwise} \end{cases} \tag{4.3}$$

4.3.4 Learning method 3: Weighted-likelihood Belief-based MRF Learning (WBL)

In this approach we consider all the states in the belief at the end of episode e. Each state s is considered with a weight depending on its probability $b(s)$ in the belief. Therefore, we compute $|S|$ vectors, $\mathcal{V}_{e,s}^{WBL}, s = 1,\dots,|S|$, and we update $\mathcal{M}_e^{WBL}$ for each $s \in S$ and for each $(i,j) \in E$ as:

$$\mathcal{M}_{e+1}^{WBL}(i,j,l,h) = = \begin{cases} \mathcal{M}_e^{WBL}(i,j,l,h) + \mathcal{X}_e(i,j,l,h), \text{ if } \mathcal{V}_e^{WBL}(i) = l \wedge \mathcal{V}_e^{WBL}(j) = h \\ \mathcal{M}_e^{WBL}(i,j,l,h), \text{ otherwise} \end{cases} \tag{4.4}$$

where $\mathcal{X}_e(i,j,l,h) = \sum_{s \in S} b(s) \cdot \mathbb{1}_{(X_i=l, X_j=h)}$, and $\mathbb{1}$ is the Kronecker delta.

4.3.5 Computation of potentials ψ from $\mathcal{M}$

We compute MRF potentials ψ from multi-dimensional array $\mathcal{M}$ at each episode e by normalizing each cell using the following formula

$$\psi_{X_i,X_j}^e(l,h) = \frac{\mathcal{M}_e(i,j,l,h)}{\sum_{w=1}^{k}\sum_{y=1}^{k}\mathcal{M}_e(i,j,w,y)} \tag{4.5}$$

where $(i,j) \in E$, namely we consider only pairs of nodes connected by an edge. For instance, given a pair of state-variables $(X_i, X_j)|(i,j) \in E$ assuming values in $\{0,1\}$, the potential $\psi_{X_i,X_j}^e(0,0) = 0.6$ corresponds to the ratio between the number of times $X_i = X_j = 0$ and the number of times each possible assignment for X_i and X_j has been observed. Namely, given $\mathcal{M}_e(i,j,0,0) = 6$, $\mathcal{M}_e(i,j,0,1) = 1$, $\mathcal{M}_e(i,j,1,0) = 1$, $\mathcal{M}_e(i,j,1,1) = 2$, we compute $\psi_{X_i,X_j}^e(0,0) = \frac{6}{6+1+1+2} = 0.6$

4.3.6 Computation of state-variables equality probabilities $\mathcal{P}$ from ψ

These probabilities are finally computed for each $(i,j) \in E$

$$\mathcal{P}_e(i,j) = \sum_{l=1}^{k} \psi_{X_i,X_j}^e(l,l) \tag{4.6}$$

In other words, $\mathcal{P}_e(i,j)$ is the sum of potentials corresponding to equal values of variables X_i and X_j. For instance, given the pair of state-variables (X_i, X_j) and the potentials $\psi^e_{X_i,X_j}(0,0) = 0.6$, $\psi^e_{X_i,X_j}(0,1) = 0.1$, $\psi^e_{X_i,X_j}(1,0) = 0.1$, $\psi^e_{X_i,X_j}(1,1) = 0.2$, we compute $\mathcal{P}_e(i,j) = 0.8$.

4.4 When can the MRF be trusted?

Since the MRF learning strategies defined above update the MRF after each episode, an important question that arises using these methods is "after how many episodes can the MRF be used by POMCP to achieve a performance improvement?". If we trust the MRF after a single episode, for instance, the relationships that it represents are specific for the state of that episode. Hence, only a set of episodes together can reveal the true relationships among state-variables. Our goal is to learn the true relationships in the minimal number of episodes to start using the MRF as soon as possible and, consequently, improving the POMCP performance as soon as possible. If the total number of episodes to perform is fixed, ending earlier the learning phase could leave more episodes to exploit the MRF, but also increases the risk of using an imprecise MRF. In the following sections, we present two methods for determining when the MRF is informative enough to be trusted and to stop the offline learning process, one based on confidence intervals and one based on the convergence of MRF potentials. After stopping the learning process, the learned MRF is used only at the beginning of each episode to initialize belief pushing the belief probabilities toward states that agree with the joint probability it represents. During the episode, the belief is updated according to the original implementation of POMCP, i.e., without using the information stored in the MRF.

4.4.1 Confidence interval-based stop learning criterion

The methodology here presented aims to stop the learning process when the MRF is informative enough to bring an improvement in planning performance. We use a statistical approach based on confidence intervals (CI) of state-variable equality probabilities $\mathcal{P}_e(i,j),\ (i,j) \in E$. Algorithm 3 formalizes the approach. It receives the matrix of equality probabilities $\mathcal{P}_e(i,j)$, the significance level α, and the episode index e. It returns true if, for every edge, the sample size is large enough and the CI of the probability does not include value 0.5. According to a known rule of thumb (Montgomery and Runger, 2010; Aczel and Sounderpandian, 2008), when a probability p is sampled, the sample size N is large enough if $(N \cdot p > 5) \wedge (N \cdot (1-p) > 5)$. This condition is checked in line 2 of Algorithm 3. The condition on the CI is instead checked in line 5. The lower and upper bounds, respectively, $L^e_{i,j}$ and $U^e_{i,j}$, are computed using the formula of the CI for population proportion (Aczel and Sounderpandian, 2008) $p^e_{i,j} \pm Z_{\alpha/2}\sqrt{\frac{p^e_{i,j}(1-p^e_{i,j})}{e}}$, in which $Z_{\alpha/2}$ represents the Z value that cuts off a right-tail area of $\alpha/2$ under the standard normal curve (Aczel and Sounderpandian, 2008).

The rationale of this algorithm is that a MRF must provide quality information for all state-variable relationships in order to be informative for planning (i.e., to improve planning performance). We translate the concept of quality into a statistical form using CIs. Namely, given a confidence level $(1-\alpha)100\%$, our approach guarantees that every *equality constraint* (i.e., edge with equality probability greater than 0.5) has only a small probability α to refer to an inequality relationship (i.e., an edge with equality probability less than 0.5). In the other way around, every *inequality constraint* has only a small probability α to actually refer to an equality relationship[1].

Algorithm 3: Confidence interval-based stop learning criterion (Zuccotto et al., 2022a)

Data: $\mathcal{P}_e(i,j)$: equality probabilities at episode e;
α: significance level; e: episode index
Result: True if learning should stop, False otherwise
for $i,j \mid (i,j) \in E$ **do** `// Iterate on all edges (i,j)`
  if $(e \cdot \mathcal{P}_e(i,j) > 5) \wedge (e \cdot (1-\mathcal{P}_e(i,j)) > 5)$ **then** `// Sz`
    $L^e_{i,j} \leftarrow \mathcal{P}_e(i,j) - Z_{\alpha/2}\sqrt{\frac{\mathcal{P}_e(i,j)(1-\mathcal{P}_e(i,j))}{e}}$ `// Lower`
    $U^e_{i,j} \leftarrow \mathcal{P}_e(i,j) + Z_{\alpha/2}\sqrt{\frac{\mathcal{P}_e(i,j)(1-\mathcal{P}_e(i,j))}{e}}$ `// Upper`
    if $(L^e_{i,j} < 0.5) \wedge (U^e_{i,j} > 0.5)$ **then** `// Chk CI`
      return False
  else
    return False
end
return True

The proposed confidence interval-based stopping criterion links to model calibration and conformal learning. All three approaches use statistical techniques to guarantee the reliability of their probabilistic estimates, the MRF equality constraints in the case of our approach, and model prediction for model calibration and conformal learning. Moreover, all three approaches use statistical techniques to achieve their goals. In particular, model calibration leverages methods like Platt scaling (Platt, 1999) or isotonic regression (Zadrozny and Elkan, 2002) to tune the calibration. On the other hand, both conformal prediction and the proposed stop-learning criterion use confidence levels to control the probability of error. In the former, the confidence level determines the width of the prediction regions, ensuring that the regions contain the true outcome with a certain confidence (Vovk et al., 2005), while in our approach it is used to compute confidence intervals for the equality probabilities stored in the learned MRF and determine whether these intervals exclude the value 0.5. Specifically, while model calibration focuses on ensuring predicting probabilities representative of the

[1]This has an interesting parallelism with hypothesis testing conducted using Student's t-test on linear regression coefficients, in which a coefficient is said to be not significant if its confidence interval contains the zero value.

true likelihood of correctness, conformal learning generates prediction intervals that contain the true outcome with a specified probability (Shafer and Vovk, 2008). In contrast, our approach focuses on determining whether the learned state-variable equality probabilities are informative and reliable enough to improve planning performance. However, the proposed stop-learning criterion differs from model calibration and conformal prediction in its specific objective and application. Model calibration is generally applied to improve the performance of predictive models, ensuring the model's output probabilities are trustworthy (Guo et al., 2017), while conformal learning generates prediction regions for new data with a specified confidence level, ensuring the reliability of predictions (Shafer and Vovk, 2008). Our approach, on the other hand, is tailored to the context of MRF learning and focuses on stopping the learning process of an MRF when state-variable relationships stored in it are informative enough to improve planning performance.

Algorithm 4: Convergence-based stop learning criterion (Zuccotto et al., 2022b)

Input: $\mathcal{P}_e(i,j)$: equality probabilities at episode e; $\mathcal{P}_{e-1}(i,j)$: equality probabilities at episode $e-1$, η: convergence threshold; ce threshold on number of consecutive episodes in which changes must be small; ct: consecutive episodes satisfying the convergence condition

Output: *stop*: stop learning flag, $\overline{ct}$: updated number of consecutive episodes satisfying the convergence condition

```
stop ← True
for i, j | (i, j) ∈ E do // Iterate on all edges (i,j)
    if |P_e(i, j) − P_{e−1}(i, j)| > η then
        stop ← False
        ct̄ ← 0
        return stop, ct̄
end
ct̄ ← ct + 1
if ct̄ < ce then
    stop ← False
return stop, ct̄
```

4.4.2 Convergence-based stop learning criterion

The methodology we propose analyzes the equality probabilities in the MRF and stops the learning phase when these probabilities converge, namely, when their values have little changes for a few consecutive episodes. More precisely, at the end of each episode e we check if each equality probability $\mathcal{P}_e(i,j),\ (i,j) \in E$, differs less than a threshold η from the same equality

probability at the end of the previous episode $e-1$. If this condition is satisfied for ce consecutive episodes then we stop the MRF learning process. Algorithm 4 formalizes the approach. It receives the matrices of equality probabilities at episodes e and $e-1$, namely, $\mathcal{P}_e(i,j)$ and $\mathcal{P}_{e-1}(i,j)$, the convergence threshold η, the threshold ce on the number of consecutive episodes, and the number ct of consecutive episodes that satisfied the condition on the convergence threshold until the current episode e. It returns the stop learning flag $stop$, and the updated number of consecutive episodes that satisfy the convergence condition $\overline{ct}$. The value of $stop$ is true if, for every edge, the difference between the value at episode e and at episode $e-1$ is below the threshold η (line 3) for at least three consecutive episodes (line 9), false otherwise. The value of $\overline{ct}$ is used for checking the stopping condition at the next episode. Determining the convergence threshold η and the threshold ce on the number of consecutive episodes with small changes must be manually fine-tuned. Automatic methods to perform this task could be an interesting focus of future research.

4.5 Application of the learning algorithms to robotic platforms

We developed a light and straightforward framework that integrates POMCP with ROS, targeting mobile robots. The architecture can also be exploited to execute the MRF learning algorithm and subsequently to run the extended POMCP that leverages the constraints in the learned MRF. Optionally, the extended POMCP can be run with MRF adaptation. The architecture can be used with all mobile robotic platforms that support the ROS Navigation Stack (Marder-Eppstein et al., 2010) and with POMDPs defined following the original POMCP implementation. Additionally, it can be executed in simulation using Gazebo. Since the architecture relies on the ROS network to communicate, the POMCP algorithm is not directly run on the machine mounted on the robotic platform but on an external one, which has more computational power. This results in a faster execution of POMCP, with lower power consumption for the mobile robot. Notice that the proposed ROS architecture aims to show the feasibility of integrating MRF learning with POMCP in real-world robotic applications using a realistic simulator (i.e., Gazebo) rather than deploying the method on a physical robot. The structure of the architecture is illustrated in Figure 4.2. It contains three main components, namely, the *environment*, the *planner*, and the *agent*, all implemented in C++. In the following paragraphs, each component is described in detail.

Environment. The environment is a discretization of the real world that exploits a task-specific representation, such as a grid for the rocksample domain.

Planner. The role of the planner is manifold. First, it runs POMCP, from the standard to the extended version. Second, it manages the whole learning process, handling the learning algorithm and keeping track of learned

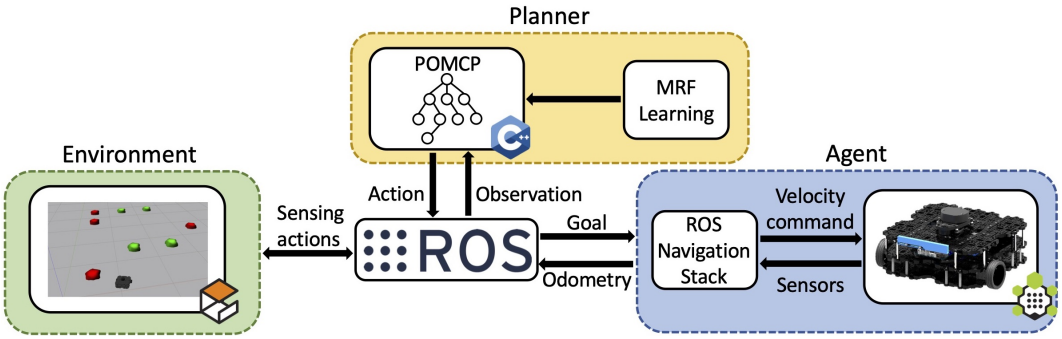

Figure 4.2: ROS architecture for running POMCP on mobile robotic platforms. The three main components are identified by the colored boxes, and they are connected to the same ROS network. The planner supports standard POMCP, extended POMCP, and our proposed approach with MRF adaptation, and besides the MRF learning algorithm (Zuccotto et al., 2022b).

relations to eventually trigger the stopping the criterion. When performing the MRF learning process, the node keeps track of the belief after that each action is performed by the agent. Then, at the end of each episode, the MRF is updated accordingly.

The planner communicates with ROS network during the *Step* function call, hence when applying the transition and the observation model of the POMDP. Right after producing the best desired action, a command is dispatched to the agent, and the planner pauses until the agent feeds back the result.

Agent. The agent node is the interface with the robotic platform. It holds information about the robot position through odometry, and it is responsible of moving the mobile platform to the desired position whenever the planner produces a goal command, which corresponds to a 3D pose in the environment. This is done exploiting the ROS Navigation Stack, which takes the pose as input, and gives a series of target velocities as output. On the other hand, if the planner produces a sensing action, the agent will directly interface with the environment, or with sensors mounted on the robotic platform.

4.6 MRF adaptation

In the previous sections we analyzed the effects of the MRF introduction. Here, instead, we deal with cases in which we use the learned MRF in episodes having unlikely state-variable configurations with respect to the joint probability defined by the MRF itself.

The MRF is learned on several episodes and it contains probabilistic information about state-variable relationships. For instance, a probability 0.9 between state-variables X_1 and X_2, i.e., $\mathcal{P}(1,2) = 0.9$, in the rocksample domain means that in 90% of the learning episodes, the most probable state-variable configuration had equal values in rocks X_1 and X_2. When the MRF is used in a new episode, however, the values of the rocks in that specif-

ic episode can be equal to or different from each other (e.g., in a specific episode X_1 could be valuable and X_2 valueless although this configuration has only probability 0.1 to occur). The MRF is used in POMCP to "push" the belief probabilities towards states that agree with the joint probability it represents. In other words, using the constraints among state-variable values introduced by the MRF, we probabilistically reduce the possibility to have in the particle filter a large amount of particles corresponding to states that have a small probability to be the true state. At each episode, we initialize the belief leveraging the information present in the MRF, peaking the probability distribution on states that reflect the equality relationships expressed in the MRF. In our example the states having the same value of X_1 and X_2 will be initialized with a higher probability. This is beneficial if the values in the true state of the current episode are actually equal to each other (which happens with probability 0.9 in our example) and harmful if the values in the true state of the current episode are actually different from each other (which happens with probability 0.1). In this second case, the belief is peaked on wrong states and even several observations could be not enough to "correct" the probability distribution over states, leading to a performance decrease with respect to the standard POMCP. Thus, the idea of the algorithm presented in Zuccotto et al. (2022b) is to adapt the probabilities in the MRF during the usage of the MRF as new evidence is gathered about the true values of the state-variables in the specific episode and there is a mismatch (i.e., *discrepancy*) between these true values and the information in the MRF. Then, the adapted MRF is used to re-initialize the belief to change the agent strategy. Let's consider, for instance, an episode of rocksample in which rock X_1 is valuable and rock X_2 is valueless. When we use the MRF, the states having different rock values are penalized, but if the agent collects the rocks, then their true values are available, hence we can detect the discrepancy between the MRF probabilities $\mathcal{P}(1, 2) = 0.9$ and the true rock values, and update the MRF probabilities accordingly, to avoid penalizing good states in the following steps of the same episode. This is the idea behind the MRF adaptation algorithm formalized in Algorithm 5 and explained in detail in the following. Notice that, given an episode e, the adaptation of the MRF has effect only in the steps after a discrepancy is detected in that episode. However, the learned MRF is restored in the next episode $e + 1$ because each episode is characterized by a different true state. We remark that the proposed algorithm does not learn a new MRF, as it adapts the information stored in the learned MRF when a discrepancy is detected during an episode and it uses the adapted MRF to re-initialize the particle filter. At the beginning of the subsequent episode, we restore the learned MRF (with no adaptation) and we use it to initialize the particle filter. On the other hand, during the learning process, we update the MRF leveraging the information given by the state with the highest probability in the belief, and we do not introduce the MRF in POMCP.

Using the MRF adapted during episode e in episode $e + 1$ involves initializing the particle filter based on the specific knowledge acquired in episode

e, i.e., peaking the probability distribution on states in which X_1 and X_2 have different values. However, the configuration of state-variable values changes with each episode, with a probability of 0.9 that X_1 and X_2 will have the same value. Note that we refer to the MRF learned over multiple episodes as the "learned MRF" and that we call the "adapted MRF" a copy of the learned MRF, where the state-variable equality probabilities are adjusted during its use when new evidence about the true values of the state variables is gathered in a specific episode and a discrepancy between these values and the information stored in the MRF is detected. As a result, the adapted MRF defines a probability distribution over unlikely state-variable configurations. In contrast, the learned MRF defines a more accurate probability distribution over states initializing, with high probability, states in which rocks X_1 and X_2 have the same value, reflecting $\mathcal{P}(1,2) = 0.9$[2]. Consequently, using an adapted MRF to initialize the particle filter might peak the probability distribution on unlikely states that may not correspond to the true state-variable values configuration. On the other hand, the learned MRF contains probabilistic information about state-variable relationships that is more accurate with respect to the probability distribution from which episodes state-variable configurations are sampled. Therefore, it allows for a more appropriate initialization of beliefs.

The inputs of the main function of the algorithm (Function `Adapt`) are: the step q of the episode, the MRF $\mathcal{P}(i,j), i = 1,\ldots,n, j = 1,\ldots,n$ updated until the current step of the current episode; the index i of the state-variable of which we have observed the true value in the current step; the vector $TV[i]$ of state-variables observed until the current step, where $TV[i] = na$ if the true value of the variable has not been observed and $TV[i] = v_i$ if the true value of the variable has been observed; $\langle a_0, o_0, ..., a_q, o_q\rangle$ the sequence of actions and observations (history) obtained up to the current execution step q. The output is the adapted MRF and the new belief b' (returned by Function `Belief_recomputation`) if a discrepancy has been detected, otherwise the function `Adapt` ends returning the received MRF and an empty belief to notify that no discrepancies have been detected.

Every time the true value of a state-variable X_i is gathered, the algorithm checks if X_i is connected to another state-variable X_j by an edge in the MRF and if both variables have been observed (line 6). In this case, the algorithm checks the value of $\mathcal{P}(i,j)$ and the values of the observed variables v_i and v_j to detect a discrepancy (line 7). In particular, a discrepancy occurs if the equality probability in the MRF is discordant with the true values of X_i and X_j. In such case, the MRF must be updated. If $\mathcal{P}(i,j) > 0.5$ and $v_i \neq v_j$ then $\mathcal{P}(i,j)$ is set to 0 (see line 10). This is because we are sure that the two variables have different values in the current episode, and the MRF is

[2]One might consider storing one or more adapted versions alongside the learned MRF, but this would offer no advantage. This is because determining which version of the adapted MRF to use for initializing the particle filter requires knowing the true values of the state variables in the current episode. However, this information is not available in advance in a partially observable environment.

Algorithm 5: MRF adaptation algorithm (Zuccotto et al., 2022b)

Function `Adapt`$(q, \mathcal{P}_e(i,j),\ i,\ TV_e,\ \langle a_1, o_1, \dots a_t, o_t\rangle)$**:**
  Input: q: step of the episode; $\mathcal{P}_e(i,j)$: equality probabilities up to step q of episode e;
  i: index of the observed state-variable X_i at step q;
  TV_e: observed state-variable up to step q of episode e
  $\langle a_1, o_1, \dots a_q, o_q\rangle$: history of actions and observations up to step q of episode e
  Output: $\mathcal{P'}_e(i,j)$: (adapted) equality probabilities,
  b': the new belief if a discrepancy has been detected, empty otherwise
  discrepancy $\leftarrow$ False
  $\mathcal{P'}_e(i,j) \leftarrow \mathcal{P}_e(i,j)$
  $b' \leftarrow b$
  for $j \leftarrow 1, \dots, n$ **do**
    if $(i,j) \in E \wedge (TV_e[j] \neq 0)$ **then** `// Check the edge (i,j)`
      `// check discrepancy`
      if $(\mathcal{P}_e(i,j) > 0.5 \wedge TV_e[i] \neq TV_e[j]) \vee (\mathcal{P}_e(i,j) < 0.5 \wedge TV_e[i] == TV_e[j])$ **then**
        discrepancy $\leftarrow$ True
        if $\mathcal{P}_e(i,j) > 0.5$ **then**
          $\mathcal{P'}_e(i,j) \leftarrow 0$ `// Set inequality on edge (i,j)`
        else
          $\mathcal{P'}_e(i,j) \leftarrow 1$`// Set equality on edge (i,j)`
        end
  end
  if *discrepancy* **then**
    $b' \leftarrow$ Belief_recomputation$(q, \mathcal{P'}_e(i,j), \langle a_1, o_1, \dots a_q, o_q\rangle)$
return $\mathcal{P'}_e(i,j), b'$

Function `Belief_recomputation`$(q, \mathcal{P}_e(i,j),\ \langle a_1, o_1, \dots a_q, o_q\rangle)$**:**
  Input: q: step of the episode; $\mathcal{P}_e(i,j)$: equality probabilities; $\langle a_1, o_1, \dots a_t, o_t\rangle$: history of actions and observations
  Output: b: new belief
  $b \leftarrow$ empty
  for *np* $\leftarrow 1 \dots NP$ **do** `// Initialize b according to the new MRF`
    $s \sim \mathcal{P}_e(i,j)$
    $b \leftarrow b \cup s$
  end
  for *m* $\leftarrow 1 \dots q$ **do** `// Replay history`
    $b' \leftarrow$ empty
    foreach *s in b* **do**
      $(s', o, r) \sim \mathcal{G}(s, a_m)$
      if $o_m == o$ **then**
        $b' \leftarrow b' \cup s'$
    end
    $b \leftarrow b'$
  end
return b

updated accordingly. If $\mathcal{P}(i,j) < 0.5$ and $v_i = v_j$ then the algorithm sets $\mathcal{P}(i,j)$ to 1 (see line 12). In this case we are sure that the two variables have the same values, and the MRF is again updated accordingly. In the example above, if rock X_1 is valuable and rock X_2 is valueless but $\mathcal{P}(1,2) = 0.9$, then this probability is set to $\mathcal{P}(1,2) = 0$ in the adapted MRF.

If a discrepancy has been detected and the MRF updated, the current belief at step q must be also updated considering the new specific knowledge acquired on the current episode. Function `Belief_recomputation` performs this task. Its inputs are the step q of the episode, the adapted MRF $\mathcal{P}(i,j)$, the history of actions and observations $\langle a_0, o_0, ..., a_q, o_q \rangle$ and its output is the updated belief b. The new belief b is first initialized (line 21) sampling NP states according to the distribution defined by the adapted MRF (we set *NP* to the number of POMCP simulations), and then updated using POMCP belief update following the current history $\langle a_1, o_1, \ldots a_q, o_q \rangle$ (see line 25). As in Silver and Veness (2010), a simulator $\mathcal{G}$ is used as a generative model of the POMDP. The updated belief b is used in the next step instead of the current belief.

4.7 Empirical evaluation: performance measures

We introduce here three measures to evaluate our MRF-learning strategy. One for evaluating planning performance, one for the goodness of the learned MRF, and one for the goodness of the belief.

- **Discounted return.** The discounted return of an episode e, called ρ_e, is the sum of the discounted rewards collected in all steps of that episode. The difference between the discounted return obtained using the learned MRF and the discounted return obtained by the standard POMCP on episode e is called $\Delta\rho_e$ and the average of this difference over all episodes of all runs is called $\overline{\Delta\rho_e}$.

- **MRF distance.** Assuming to know the true matrix of probabilities of state-variables equalities $\mathcal{P}^*(i,j), (i,j) \in E$ and that computed after learning episode e by our method $\mathcal{P}_e(i,j),\ (i,j) \in E$, the MRF distance d_M^e is computed as the Euclidean distance normalized by the number of edges in the MRF $\|\mathcal{P}^* - \mathcal{P}_e\|_2/|E|$ between the two matrices. Usually, we compute this measure after the last learning episode and we call it d_M. The average of this difference over all learning stages of different runs is called $\overline{d_M}$.

- **Belief-state distance.** This measure, introduced in Castellini et al. (2020c), allows to quantify the discrepancy between the agent's belief about the true state and the true state itself. The prior knowledge introduced by the learned MRF is expected to improve the planning performance (i.e., discounted return) by improving the belief, hence it is expected to reduce the belief-state distance. This distance, called

d_{SB} in the following, is the weighted averaged Manhattan distance between the state-variable configuration in the true hidden state and the state-variable configuration of all states $s \in \mathcal{S}$ weighted by their belief probability $b(s)$. Mathematically, if we define the configuration of rocks in the true state as $\phi_S = (\phi_1, \ldots, \phi_n)$ where $\phi_j \in \{1, \ldots, k\}$ and n is the number of rocks, and we define the m configurations of rocks in the belief as $f_B^i = (\phi_1^i, \ldots, \phi_n^i)$, $i \in \{1, \ldots, m\}$, $\phi_j \in \{1, \ldots, k\}$ where the probability of each rock configuration ϕ_B^i in the belief is p_B^i, then the belief-state distance is

$$d_{SB} = \sum_{i=1}^{m} (p_B^i \cdot \sum_{j=1}^{n} |\phi_j - \phi_j^i|). \tag{4.7}$$

In our tests, we compute the difference of the belief-state distance reached at each step by POMCP with the MRF and the standard POMCP, and we name it Δd_{SB}. Then we average these differences over all steps of all episodes of all runs, and we name this $\overline{\Delta d_{SB}}$.

- **Time complexity.** The learning strategies together with the stopping criterion have a complexity $\mathcal{O}((|\mathcal{S}|+|E|)\cdot NE)$, where $|\mathcal{S}|$ is the number of states (to scan the belief), $|E|$ is the number of edges (to update $\mathcal{M}$) and NE is the number of learning episodes.

4.8 Comparison of learning methods

We perform two tests on rocksample(5,8), presented in Section 2.7.1 (see Figure 4.3a). In the first test (called *test 1* in the following) we compare the performance of the three learning methods presented in Section 4.3. In the second test (called *test 2*) we investigate the dependence of the MRF performance on the number of learning episodes.

4.8.1 Experimental setting

In both tests we perform *NR=5 runs*. Each *run* is composed of *NE=100 episodes* of rocksample, and in each episode the agent performs *NS=100 steps*. In each episode rock values change but they always satisfy the constraints of the MRF of Figure 4.3b. POMCP always uses 100000 particles and performs the same number of simulations. For each run, the learned MRF is then tested by performing the same 100 episodes. Performance are evaluated in terms of discounted return and distance between the true MRF and the learned MRF. Notice that during the learning phase the information gathered in the MRF is not used in POMCP.

4.8.2 Test 1: Identification of the best MRF learning method

In this test, for each run we learn the MRF first with SL, then with MBL, and finally with WBL. Afterwards, we insert each learned MRF in POMCP

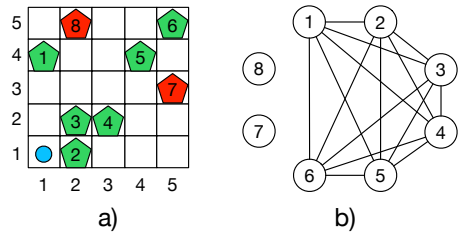

Figure 4.3: Running example on rocksample (Zuccotto et al., 2022a). *a)* Board with valuable (green) and valueless (red) rocks. *b)* MRF topology.

and compute its performance. Looking at Figure 4.4 we observe that the three MRF-learning methods do not lead to the same learning result and their effect on the performance is different (we report the average results with the corresponding standard deviations). The distance (d_M), between the exact MRF and the MRF obtained with SL, MBL and WBL, respectively is depicted in Figure 4.4a. To clarify the meaning of this measure, we visualize the probabilities of each MRFs as heatmaps in Figure 4.4b. The color of each cell in the heatmap goes from blue to red expressing the probability of the equality constraint from 0 to 1, respectively. As we can see, the best learning performance is obtained with MBL, followed by WBL. Both compute MRF of good quality. In Figure 4.4c we show the cumulative discounted returns ρ obtained applying the MRFs computed by SL, MBL and WBL, compared with those computed by standard POMCP (named STD in the following) and the exact MRF (called EXCT in the following). The worst performing method is SL, due to the misleading information in its MRF, while the cumulated discounted return of the two belief-based learning methods falls between that of STD and EXCT. To clarify the difference in terms of performance achieved by the three learning methods, in Figure 4.5 we show the difference $\Delta\rho_e$ between the discounted return obtained by STD and that obtained using the MRF learned by the three methods in all the runs (i.e, each box plot represents data of 5 runs). The worst performance is achieved by SL, with a median $\Delta\rho_e$ of -1.11. This method reaches worse performance than STD because the MRF is computed using only information from collected rocks (not from observed rocks), and POMCP tends to collect more valuable rocks than valueless rocks, hence the information is partial and the MRF is not accurate. This is clear also analyzing the final distance between the true MRF and the learned MRF, namely, $\overline{d_M} = 0.11$. MBL and WBL instead achieve a performance improvement, with median $\Delta\rho_e$ 3.92 and 2.19, respectively. This is because these methods use the belief as a source of information, hence they consider the information coming from observations (i.e., rock sampling) that are performed on both valuable and valueless rocks. MBL reaches an average distance to the true MRF of 0.01 and WBL of 0.03. This analysis clearly shows that MBL reaches the best performance. Figure 4.5 finally shows that the median $\Delta\rho_e$ achieved by using the true MRF is 4.10, higher than all the other methods, as expected.

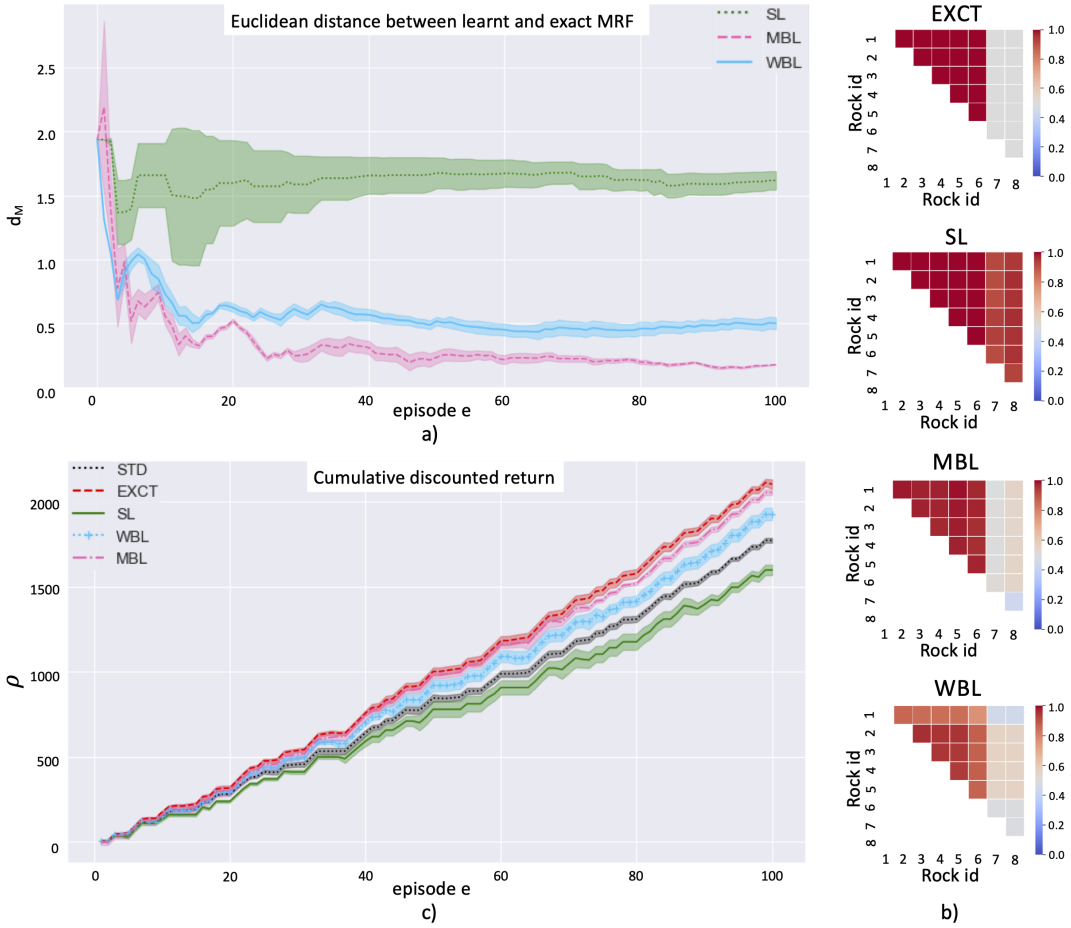

Figure 4.4: a) Euclidean distance between learned MRFs and exact MRF. b) Heatmaps of the exact MRF and the MRFs learned with SL, MBL, and WBL, respectively. c) Comparison between cumulative discounted returns obtained applying the MRFs learned by SL, MBL, WBL and the ones obtained by STD and EXCT.

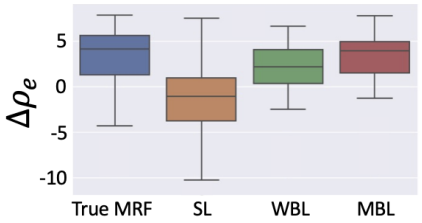

Figure 4.5: Density of difference in discounted return from STD (Zuccotto et al., 2022a).

4.8.3 Test 2: Dependence of the performance on the number of training episodes

In test 2 we focus on MBL, since it is the learning method with the lowest final d_M and highest ρ, and we test the effect of the number of training episodes

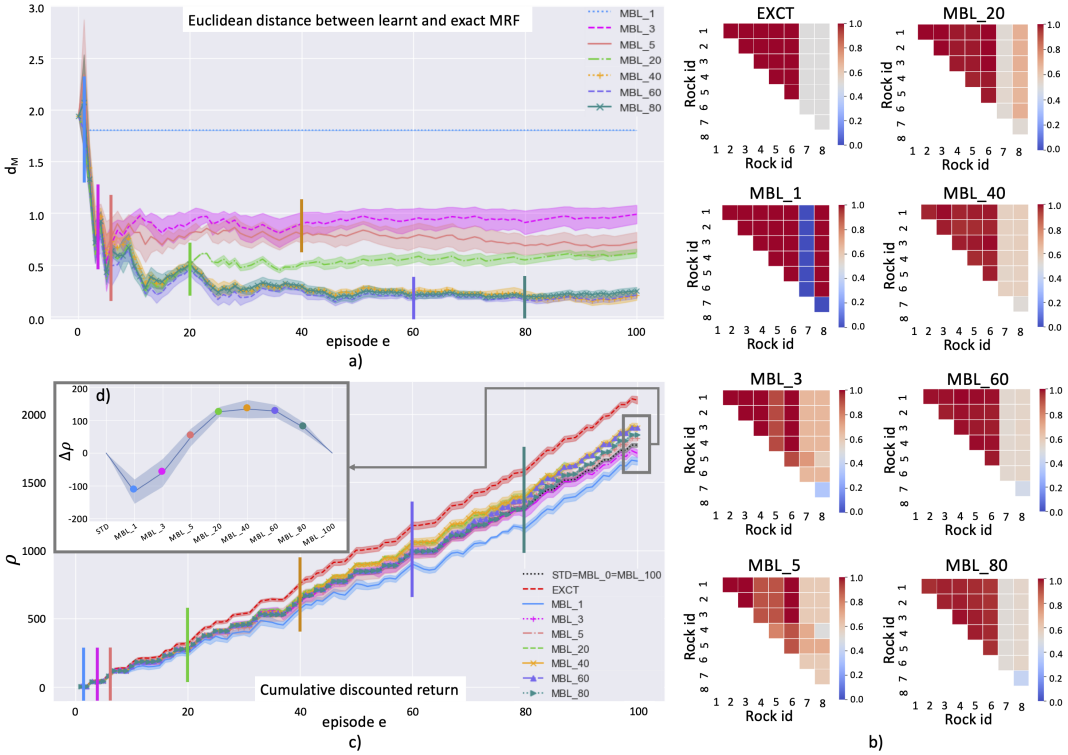

Figure 4.6: a) Euclidean distance between learned MRF and exact MRF. b) Heatmaps of the exact MRF and the MRFs learned with MBL_e, where $e \in \{1, 3, 5, 20, 40, 60, 80\}$. c) Comparison between average discounted returns obtained with MBL_e and the ones accumulated by STD (that is equivalent to MBL_0 and MBL_{100}) and EXCT d) Difference in discounted return from STD.

on planning performance. We learn the MRF during the first e episodes, with $e \in \{1, 3, 5, 20, 40, 60, 80\}$, and then we use the MRF in the remaining episodes. Performance are measured after the 100th episode. We identify each version of these tests with symbol MBL_e (MBL_e in the figures).

In Figure 4.6a we show the d_M distances between the learned MRFs as the episodes are performed, highlighting with a vertical line the episode at which MRF learning phase is ended and the agent starts to use the learned MRF. The image shows that MBL_1, namely the test performed with the MRF trained for a single episode, has the highest $\overline{d_M}$ value because the information collected in a single episode is definitely not enough to learn the correct state-variable relationships, hence the MRF is very different from the exact one. MBL_{40}, MBL_{60} and MBL_{80} have the best learning results in terms of $\overline{d_M}$ and their MRFs are close to the MRF obtained after 100 training episodes (see Figure 4.6b and Figure 4.4b). MBL_3, MBL_5 and MBL_{20} learning performance fall between MBL_1 and MBL_{40} and the MRF computed in a single episode (i.e., MBL_1) has the worst d_M. To better understand these differences, in Figure 4.6b we visualize MRFs as heatmaps and show that they differentiate on constraints involving rocks 7 and 8. As we can see the MRF obtained with MBL_1 expresses hard equality constraints (i.e., probabilities

are 1) between rocks 1-6 and 8, and hard inequality constraints (i.e., probabilities are 0) between rocks 1-6 and rock 7. These extreme values of probability correspond to a state-variables assignment consistent with the exact MRF but it is specialized on the state of episode 1, that is $\langle 0, 0, 0, 0, 0, 0, 1, 0\rangle$. As a consequence, the belief is always initialized only with states that satisfy this constraint $\langle 1, 1, 1, 1, 1, 1, 0, 1\rangle$ and $\langle 0, 0, 0, 0, 0, 0, 1, 0\rangle$.

In Figure 4.6c we show the impact of the training duration on planning performance ρ, highlighting with a vertical line the episode at which MRF learning is ended and the agent starts to use the learned MRF. The effect of the low quality MRF learned in MBL_1 is clearly visible: its discounted return ρ is the lowest, falling under the line of STD, hence the planning performance are worse than not using prior knowledge at all. Regarding MBL_3, we see that the line is mostly superposed or below that of STD, while the line of MBL_5 is mostly superposed or above that of STD, thus 5 learning episodes seem to be the minimum to obtain a (small) performance improvement with respect to STD. On the other hand, the effect of the longer training in MBL_{20}, MBL_{40}, MBL_{60}, MBL_{80} makes performance grow over that of STD and approaching that of EXCT. Despite the high quality MRF obtained with MBL_{80}, the use of the MRF starts too late and there is no way to regain the loss cumulated while learning the MRF. For what concerns MBL_{60}, its earlier introduction of the MRF balances the lack of information on constraints involving rocks 7 and 8, making this test one of the best performing. The highest value of final average cumulative discounted return $\overline{\rho}$ is obtained with MBL_{40} despite a more probable equality relationship between rocks from 1 to 7 and rocks 7 and 8. To better understand the performance improvement we compute the mean difference between the final cumulative discounted return obtained by STD and other methods, $\Delta\rho$ (see Figure 4.6d). Interestingly, the MRF obtained using one and three episodes, labels MBL_1 and MBL_3 in the chart, have lower performance than STD. This is because they are specialized only on the rock configurations of these episodes but they do not generalize enough on the true probability distribution of rock configurations. In fact, the distances of these two MRFs to the true one are high, i.e., $\overline{d_M} = 0.12$ and $\overline{d_M} = 0.05$, respectively. Figure 4.5b shows instead that the performance obtained with the MRFs learned with 5 and more episodes are higher than that of STD, with a maximum obtained with 40 episodes (see MBL_40). Notice that in our test the performance decreases after 40 episodes not because the MRF gets worse but because it has fewer and fewer episodes to exploit the acquired knowledge. Notice that, this test is designed specifically to evaluate the effect of the number of training episodes on planning performance. The stop-learning problem we will discuss in subsequent sections focuses only on uncertainty on the probabilities stored in the MRF, not on the number of episodes used for learning. Generalizing about the number of episodes is beyond our scope. Moreover, including the number of learning episodes in the stopping criterion would transform it into a planning problem. This is because one would need to account for future episodes, significantly increasing the complexity of the problem.

4.9 Performance of MBL with stopping criterion

In these experiments, we perform tests using MBL with the stopping criterion based on confidence intervals described in Section 4.4.1 on rocksample(5,8) outlined in Section 2.7.1 (see *test 3* in the following) and velocity regulation introduced in Section 2.7.2 (see *test 4* in the following).

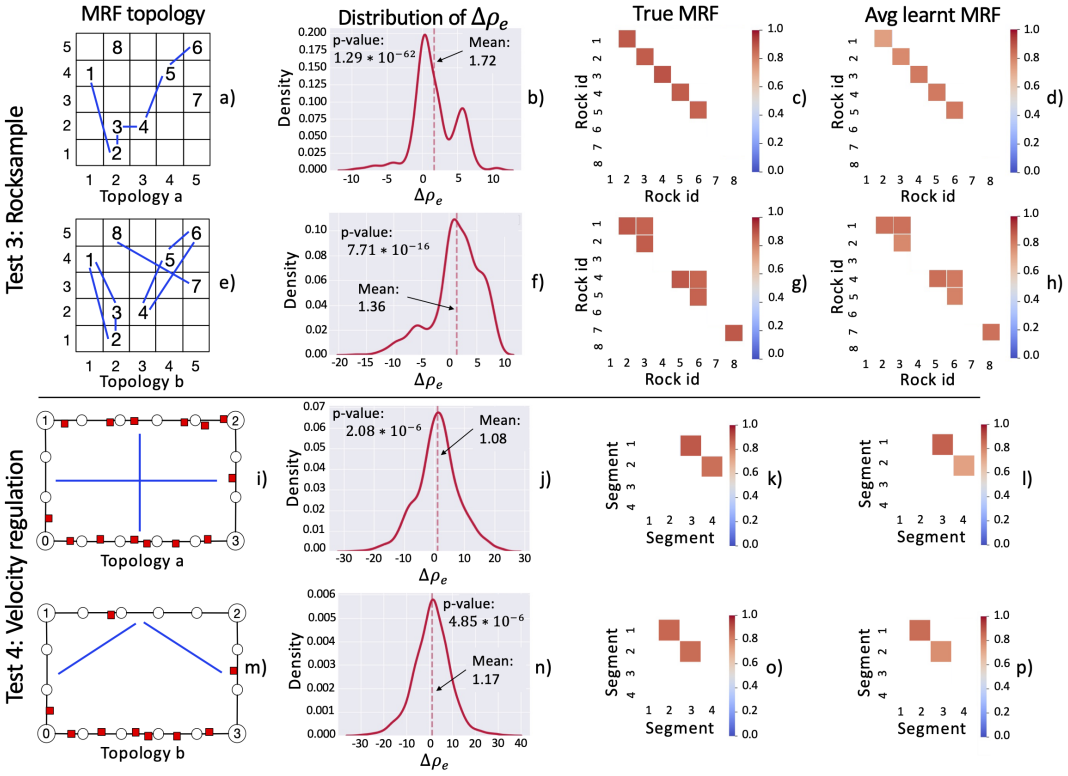

Figure 4.7: Test on rocksample (Test 3) and on Velocity regulation (Test 4) (Zuccotto et al., 2022a). *a)*, *e)*, *i)* and *m)* MRF topologies. *b)*, *f)*, *j)* and *n)* Density of difference in discounted return from STD. *c)*, *g)*, *k)* and *o)* Heatmap of the true MRF to learn. *d)*, *h)*, *l)* and *p)* Average of the learned MRFs during experiments 1 and 2.

4.9.1 Experimental setting

In test 3 we first select a true MRF (i.e., a set of relationships among rock values, as those shown in Figures 4.7a,e). Edge probabilities are always set to 0.9 in the true MRF. We perform $NR=10$ runs. In each run we start preparing an empty MRF having the same topology (i.e., set of edges) of the true one (notice that our current method does not learn the topology of the MRF but only the potentials of a MRF with pre-defined topology). We learn the MRF potentials for a number of episodes determined by the stopping criterion in which the information in the MRF is not used in POMCP. The configuration of rock values changes with each episode satisfying the distribution defined by the true MRF. Then we evaluate the performance of the MRF by introducing

it in POMCP and performing *NE=100* episodes with and without the MRF, comparing the discounted return of each episode and averaging it over all the runs. In each episode the agent performs *NS=70* steps. POMCP always uses 100000 particles and performs the same number of simulations. Test 4 is performed in the same way but on the velocity regulation domain. The MRF topologies used are displayed in Figures 4.7.i,m. Edge probabilities are again set to 0.9. The number of steps per episode is in this case *NS=16*, namely, the number of subsegments in the path. To summarize the flow of test 3 and test 4, we show in Figure 4.9 the MRF learning and usage for MBL with stopping criterion.

The confidence level for our tests was determined empirically by evaluating three values: 99% ($\alpha = 0.01$), 95% ($\alpha = 0.05$), and 90% ($\alpha = 0.1$). Figure 4.8.a illustrates the topology of the MRF and the state-variable equality relationships we aim to learn, while Figures 4.8.b and 4.8.c depict the variation of the quantity $Z_{\alpha/2}\sqrt{\frac{p^e_{i,j}(1-p^e_{i,j})}{e}}$ while learning the equality relations between state variables X_1 and X_2, and between X_6 and X_7. The trends of the three curves are similar. We chose the 95% confidence level ($\alpha = 0.05$) to find a balance between the confidence level and the interval width.

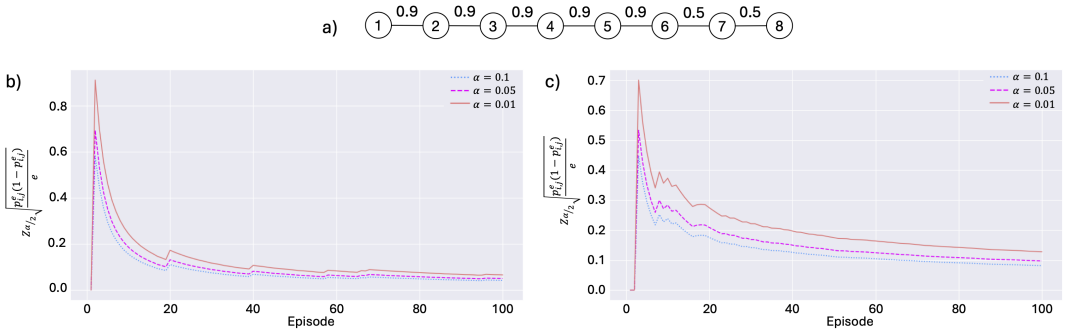

Figure 4.8: *a)* MRF topology with the equality probability constraints on its edges. Variation of $Z_{\alpha/2}\sqrt{\frac{p^e_{i,j}(1-p^e_{i,j})}{e}}$ computed with $\alpha = 0.01$, $\alpha = 0.05$, and $\alpha = 0.1$ for *b)* edge $(1, 2)$ and *c)* edge $(6, 7)$.

To prove that the introduction of the learned MRF provides a statistically significant improvement with respect to the standard POMCP, we show that the average difference $\overline{\Delta\rho_e}$ between the discounted return obtained with the learned MRF and the discounted return obtained with STD is significantly larger than zero. Notice that the difference is computed episode by episode to reduce the randomness and the average is computed across all the NE=100 episodes of each run (i.e., over 1000 episodes in total). More precisely, at episode e, we compute the difference of discounted return ρ_e as $\Delta\rho_e = \rho_e^{MBL} - \rho_e^{STD}$. Then we compute the average of these values over all the episodes of all the runs *average discounted return* $\overline{\Delta\rho_e}$. Similarly, we compute the *average belief-state distance* $\overline{\Delta d_{SB}}$. This is however averaged over all steps of each episode of all runs, which are $70 \cdot 100 \cdot 10 = 70000$ in rocksample and $16 \cdot 100 \cdot 10 = 16000$ in velocity regulation.

Table 4.1: Performance improvement obtained by the learned MRF compared to STD in tests 3 and 4.

Test	MRF topol.	$\overline{\Delta\rho_e}(\overline{\Delta\rho_e\%})$	p-val	$\overline{d_M}$	$\overline{\Delta d_{SB}}$
3	a	**1.72(8.35%)**	$1.29 \cdot 10^{-62}$	0.04	-0.16
3	b	**1.36 (7.16%)**	$7.71 \cdot 10^{-16}$	0.03	-0.14
4	a	**1.08 (3.52%)**	$2.08 \cdot 10^{-6}$	0.04	-0.52
4	b	**1.17 (3.92%)**	$4.85 \cdot 10^{-6}$	0.06	-0.33

4.9.2 Test 3: results on rocksample

The results on rocksample are summarized in the first two rows of Table 4.1 and the first two rows of Figure 4.7. The main result is represented by the average difference in discounted return $\overline{\Delta\rho_e}$ achieved using the learned MRF compared to not using it, which is 1.72 (i.e, 8.35%) with the MRF topology *a* displayed in Figure 4.7a, and 1.36 (i.e, 7.16%) with the MRF topology *b* displayed in Figure 4.7e. These average differences are computed over 100 episodes and 10 runs. The distributions of these differences are displayed in Figures 4.7b,f. We verified that these average differences are statistically different from zero using the Student's t-test. In both cases the p-values are lower than 0.05, confirming that the learned MRF produces on average a statistically significant performance improvement.

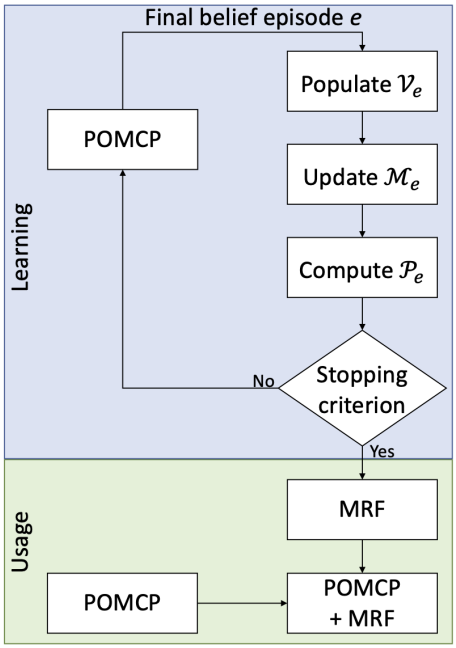

Figure 4.9: Main steps of MRF learning and usage with MBL with stopping criterion. The blue background highlights the MRF learning phase, while the green one emphasizes the MRF usage process (Zuccotto et al., 2022a).

To explain the motivation of this improvement we show in Figures 4.7c,d and in Figures 4.7g,h, the differences between the true MRF and the learned MRF, respectively, for the first and second MRF topology. The similarity is very high, in fact the average MRF distance is $\overline{d_M} = 0.04$ for topology *a* and $\overline{d_M} = 0.03$ for topology *b* (see Table 4.1). This shows that MBL with the proposed stopping criterion manages to generate an accurate MRF. Furthermore, we display in the fifth and sixth column of Table 4.1 that the usage of the learned MRF produces a statistically significant decrease in the *average belief-state distance* $\overline{\Delta d_{SB}}$. The negative values of these differences, respectively -0.16 and -0.14, mean that the average belief-state distance obtained using the MRF is smaller than that obtained without using it. This provides insight into the mechanism that generates the improvement in the discounted return. Namely, using the learned MRF the belief tends to converge faster to the true state (i.e., the true rock value configuration) allowing the agent to collect larger rewards than standard POMCP. Finally, in Figure 4.10 we show the trends of the lower bounds $L^e_{i,j}$ of probabilities $\mathcal{P}_e(i,j),\ (i,j) \in E$ of the MRF learned in the first run of test 3 (topology *a*, shown in Figure 4.7a). Episodes e are displayed on the x-axis and lower bound values $L^e_{i,j}$ on the y-axis. Since the equality probabilities in the true MRF are all 0.9, the stopping criterion (Algorithm 3) stops the learning phase (line 5) when all lower bounds are higher than 0.5 (black horizontal line in Figure 4.10) and the condition on the sample size (line 2) is satisfied. This happens at episode 28 in the chart. We highlight that the lower bounds tend to be low in the first episodes, when the counts of state-variables equalities/inequalities $\mathcal{M}^{MBL}_{e+1}(i,j,l,h)$ are low, and they tend to converge to the true equality probability as the number of episodes increases.

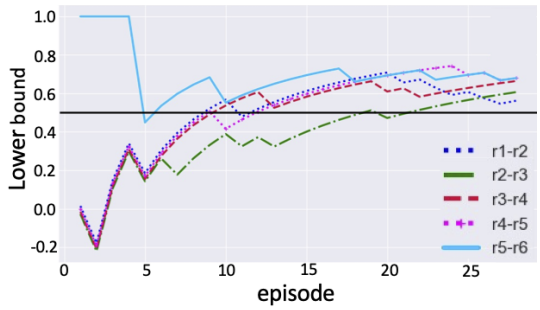

Figure 4.10: Lower bound of MRF equality probability for the first run of test 3/a, black line represents value 0.5 (Zuccotto et al., 2022a).

4.9.3 Test 4: results on velocity regulation

The experiments on the velocity regulation domain confirm the positive results achieved with rocksample. The average increase of discounted return obtained using the MRF is in this case of 3.52% with the MRF topology *a* displayed in Figure 4.7i, and of 3.92% with the MRF topology *b* displayed in Figure 4.7m. The numerical results are summarized in the third and fourth

rows of Table 4.1. Also in this case the p-values of the $\overline{\Delta\rho_e}$, which are lower than 0.05, confirm the statistical significance of the improvement. The motivation of the improvement can be found, again, in the very good approximations of the true MRFs (see Figures 4.7l,p,k,o and Table 4.1) generated by MBL learning with stopping criterion. These good approximations yield a statistically significant reduction of the *average belief-state distance* $\overline{\Delta d_{SB}}$, of -0.52 and -0.33, respectively, in the first and second MRF topology.

4.10 Experimental results on robotic platforms

In these experiments, we perform tests using MBL with the stopping criterion based on the convergence of MRF potentials described in Section 4.4.2. Then, we test the ROS architecture for MRF learning with the MBL approach and the convergence-based stopping criterion (see Section 4.4.2). Both tests are performed on rocksample(5,8) outlined in Section 2.7.1.

4.10.1 Test on MRF learning

Experimental setting

We perform tests using the MRF learning algorithm (see Section 4.3.3) with the stopping criterion based on the convergence of MRF potentials (see Section 4.4.2) to learn the MRF. Experiments are performed on the rocksample domain described in Section 2.7.1 using a *C++* simulator.

In this test we first select a true MRF (i.e., a set of relationships among rock values, see Figure 4.11a). Edge probabilities are always set to 0.9 in the true MRF. We perform *NR=10* runs, hence we compute 10 MRF. In each run we start preparing an empty MRF having the same topology (i.e., set of edges) as the true one (notice that our current method does not learn the topology of the MRF but only the potentials of an MRF with pre-defined topology). We learn the MRF potentials for a number of episodes determined by the stopping criterion with threshold $\eta = 0.01$ and $ce = 3$. The configuration of rock values changes with each episode satisfying the distribution defined by the true MRF. Then we evaluate the performance of the learned MRF performing *NE=100* episodes with EXCT and STD algorithms, comparing the discounted return of each episode and averaging it over all the runs. In each episode the agent performs *NS=60* steps. The POMCP always uses 100000 particles and performs the same number of simulations.

To prove that the introduction of the learned MRF provides a statistically significant improvement with respect to STD, we show that the average difference $\overline{\Delta\rho_e}$ between the discounted return obtained with EXCT and the discounted return obtained with STD is significantly larger than zero. Notice that the difference is computed across all the NE=100 episodes of each run (i.e., over 1000 episodes in total). More precisely, at episode e, we compute the difference of discounted return ρ_e as $\Delta\rho_e = \rho_e^{EXCT} - \rho_e^{STD}$. Then we compute the average of these values over all the episodes of all the runs *average discounted return* $\overline{\Delta\rho_e}$.

Results on Rocksample

The results we obtained using the C++ simulator are summarized in Figure 4.11. The main result is represented by the average difference in discounted return, $\overline{\Delta\rho_e}$, achieved using the learned MRF in EXCT with respect to STD that does not use any kind of prior knowledge. The value of $\overline{\Delta\rho_e}$ is 1.15 and corresponds to a performance improvement of 5.99% (see Figure 4.11d). The distribution and the corresponding average difference are computed over 100 episodes and 10 runs. To verify that $\overline{\Delta\rho_e}$ is statistically different from zero, we perform the Student's t-test that confirms the statistical significance of the result since the p-value is lower than 0.05.

To explain the motivation of this improvement in Figure 4.11b we compare the true and the learned MRF. On the x-axis we display the edges of the MRF topology, while on the y-axis we show edge probability values. With pink dots we represent the values on the true MRF edges (i.e., that from which we sampled the state-variable configurations of the learning episodes), while blue dots and lines represent, respectively, the average values of the learned MRF and their standard deviations (where the average is computed over the 10 runs performed in the learning process). The picture shows that the similarity between learned MRFs and the true one is very high. Moreover, Figure 4.11c depicts the trend of difference in probability values of all edges during a run of the learning process until it is stopped by the proposed criterion. In episode 25, in the x-axis, the difference in equality probabilities of all edges starts to be lower than 0.01, the threshold used in the stopping criterion. Since this condition persists in the next 3 episodes, the stopping criterion ends the learning phase at episode 27. Similar results with a different stopping criterion have been presented in Zuccotto et al. (2022a).

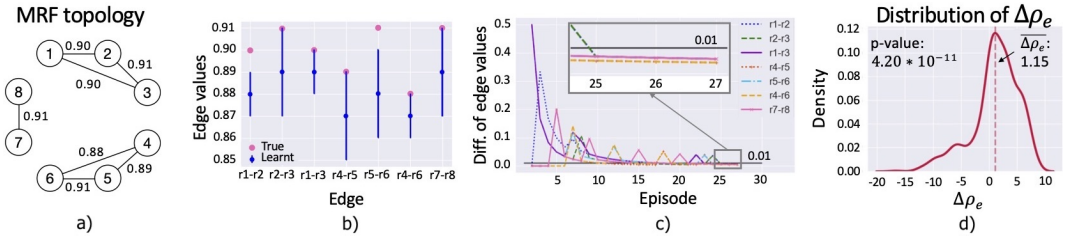

Figure 4.11: Test on MRF learning using the C++ simulator of rocksample (Zuccotto et al., 2022b). a) True MRF topology with the equality probability constraints on its edges. b) True MRF and average of the learned MRFs. Pink dots represent the values on the edges of the true MRF, while blue dots and lines correspond to the average edge values of the learned MRF and their standard deviations, respectively. c) Difference of edge probability values during an execution of the learning process until the convergence is reached at episode 27. The black line represents the convergence threshold. d) Density of difference in discounted return from STD.

4.10.2 Test on the ROS architecture for MRF learning

Experimental setting

We perform tests using the MRF learning algorithm (see Section 4.3.3) with the stopping criterion (see Section 4.4.2) to learn the MRF on the ROS-architecture proposed in Section 4.5. We perform our tests on the open-source multi-robot simulator Gazebo (Koenig and Howard, 2004), in which a TurtleBot3 acts in the rocksample domain described in Section 2.7.1.

In this test we first select a true MRF (see Figure 4.12a). Edge probabilities are always set to the values on the edges of the true MRF topology. We perform *NR=10* runs, in each run we start preparing an empty MRF having the same topology of the true one. We learn the MRF potentials on the Gazebo environment running the learning algorithm for a number of episodes determined by the stopping criterion with threshold $\eta = 0.01$ and $ce = 3$. The configuration of rock values changes in each episode satisfying the distribution defined by the true MRF shown in Figure 4.12a. Then, we test the performance of the learned MRF performing *NE = 100* episodes with EXCT and STD, comparing the discounted return of each episode and averaging it over all the runs. The MRF we used is the average of the 10 MRFs obtained with the learning process. In each episode the agent performs *NS = 60* steps. The POMCP always uses *NP = 100000* particles and performs the same number of simulations.

To prove that the introduction of the learned MRF provides a statistically significant improvement with respect to STD, we show that the average difference $\overline{\Delta\rho_e}$ between the discounted return obtained with the MRF learned with the ROS-architecture on Gazebo and the discounted return obtained with STD on the same framework is significantly larger than zero. Notice that the difference is computed episode by episode and the average is computed across all the NE=100 episodes of each run (i.e., over 1000 episodes in total). More precisely, at episode e, we compute the difference of discounted return ρ_e as $\Delta\rho_e = \rho_e^{EXCT} - \rho_e^{STD}$. Then we compute the average of these values over all the episodes of all the runs *average discounted return* $\overline{\Delta\rho_e}$.

Results on Rocksample

The results we obtained using the Gazebo simulator are summarized in Figure 4.12. The main result consists in the average difference of discounted return, $\overline{\Delta\rho_e}$, achieved using the learned MRF in EXCT with respect to STD that does not use any kind of prior knowledge. The value of $\overline{\Delta\rho_e}$ is 1.28 and corresponds to a performance improvement of 5.88% (see Figure 4.12d). The distribution and corresponding average difference is computed over 100 episodes and 10 runs. To verify that $\overline{\Delta\rho_e}$ is statistically different from zero, we perform the Student's t-test that confirms the statistical significance of the result since the p-value is lower than 0.05.

What allows this improvement is visible in Figure 4.12b in which we compare the true and the learned MRF. On the x-axis we display the edges of the

MRF topology, while on the y-axis we show edge probability values. Pink dots represent the values on the true MRF edges (i.e., that from which we sampled the state-variable configurations of the learning episodes), while blue dots and lines represent, respectively, the average values of the learned MRF and their standard deviations (where the average is computed over the 10 runs performed in the learning process). The picture shows that the learned MRFs are very similar to the true one. Moreover, this also shows that using the proposed learning approach implemented in the ROS architecture allows us to learn accurate MRFs. Figure 4.12c depicts the trend of difference in probability values of all edges during a run of the learning process until it is stopped by the proposed criterion. At episode 20, in the x-axis, the difference in equality probabilities of all edges starts to be lower than 0.01, the threshold used in the stopping criterion. Since this condition persists in the next 3 episodes, the stopping criterion ends the learning phase at episode 23.

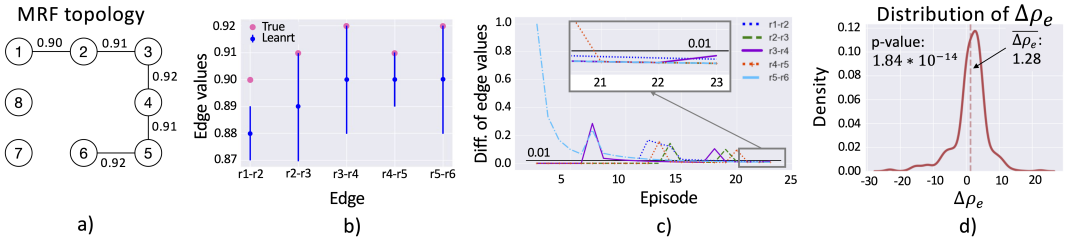

Figure 4.12: Test on the ROS architecture for MRF learning using the Gazebo simulator of rocksample (Zuccotto et al., 2022b). a) True MRF topology with the equality probability constraints on its edges. b) True MRF and average of the learned MRFs. Pink dots represent the values on edges of the true MRF, while blue dots and lines correspond to the average edge values of the learned MRF and their standard deviations, respectively. c) Difference of edge probability values during an execution of the learning process until the convergence is reached at episode 23. The black line represents the convergence threshold. d) Density of difference in discounted return from STD.

To further clarify the learning process performed on the ROS-based architecture, in the supplementary material, we provide a video showing four learning episodes performed by a TurtleBot in the Gazebo simulator of the rocksample domain [3]. Figure 4.13 shows a snapshot of the video. The mobile robot acting in the Gazebo environment is shown in Figure 4.13a. When it performs a sensing action on a rock a question mark appears in the cell containing the rock. This cell becomes green or red if the outcome of the sensing action identifies the rock as valuable or valueless. When the agent performs a sampling action on a rock, the cell in which the rock is posed turns blue to specify that the rock has been collected. Figure 4.13b shows the true rock value configuration of the episode that satisfies the distribution

[3] https://www.frontiersin.org/articles/file/downloadfile/819107_supplementary-materials_videos_1_mp4/octet-stream/Video%201.MP4/1/819107?isPublishedV2=False

defined by the true MRF. In Figure 4.13c,d we show the edge probability values of the learned MRF updated at the end of episode 23 and the ones of the true MRF we aim at learning. In Figure 4.13e we show the evolution of the edge probability values in the learned MRF and when all the values reach the convergence in Figure 4.13e, the learning process ends.

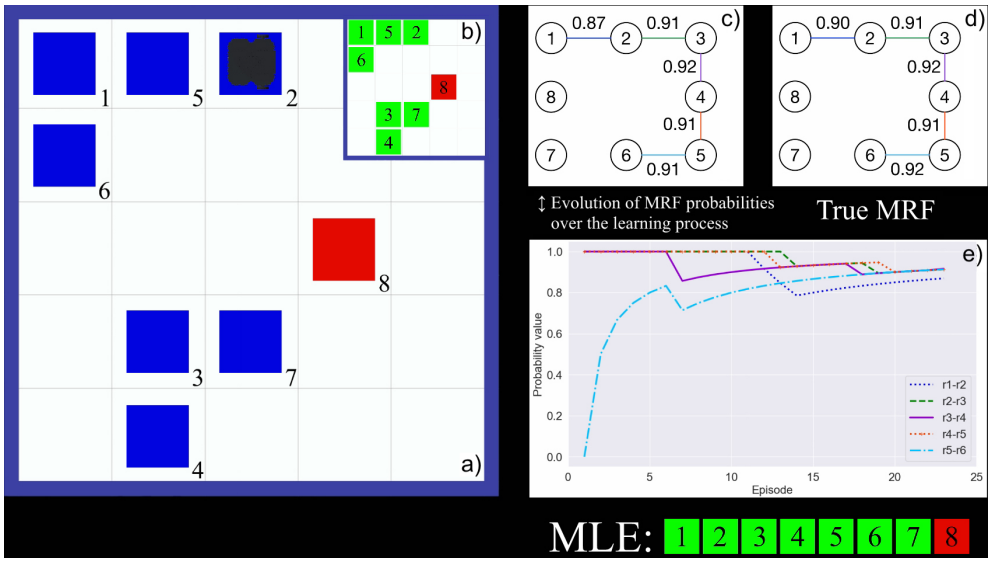

Figure 4.13: Snapshot of the video showing four learning episodes performed by a Turtlebot in the Gazebo simulator of the rocksample domain (Zuccotto et al., 2022b). a) Instance of the rocksample environment in which the TurtleBot acts during episode 23. b) True rock values in episode 23. c) MRF learned after 23 episodes. d) True MRF, the one to be learned. e) Evolution of edge probability values at the end of episode 23. When they converge the learning process is ended.

4.11 Experimental results on MRF adaptation

In these experiments, we perform tests on MRF adaptation described in Section 4.6 on rocksample(5,8) outlined in Section 2.7.1 and on velocity regulation presented in Section 2.7.2 to which the angular velocity model is introduced as described in Section 4.11.1.

4.11.1 Experimental setting

We perform two tests to evaluate the ADA method described in Section 4.6. One is performed on rocksample(5,8) and the second on velocity regulation. For both domains, a C++ simulator of the environment was used to avoid the slowdown introduced by Gazebo, since the physics of the environment is not critical for evaluating this algorithm. The goal of our tests consists in highlighting that using ADA we can, on average, improve the performance of the planner over both STD and EXCT. This improvement is achieved by

limiting the performance decrease generated when the learned, or given by expert, MRF is used on episodes characterized by unlikely state-variable configurations. In both tests we perform $NR = 10$ runs using an MRF that reflects probabilistic equality constraints among state-variable values learned using the MRF learning method of Section 4.3.3. To evaluate the performance of ADA we perform $NE = 100$ episodes using the MRF adaptation approach every time a discrepancy is detected during an episode and $NE = 100$ episodes using the EXCT algorithm. Then we compare the discounted return of the two methods considering only the episodes in which the MRF adaptation approach has been used and average it over all the runs. In each episode the agent performs $NS = 60$ steps in the rocksample domain, while in the velocity regulation environment it performs for $NS = 32$ steps, namely, the number of subsegments in the path. The configuration of rock values (for the test on rocksample) and of segment difficulties (for the test on velocity regulation) changes with each episode satisfying the distribution defined by the true MRF depicted in Figure 4.14b. The POMCP always uses $NP = 100000$ particles and performs the same number of simulations. We summarize the parameters used in our tests in Table 4.2.

Table 4.2: Parameters of tests on ADA.

Environment	NR	NE	NS	NP
Rocksample	10	100	60	100000
Velocity regulation	10	100	32	100000

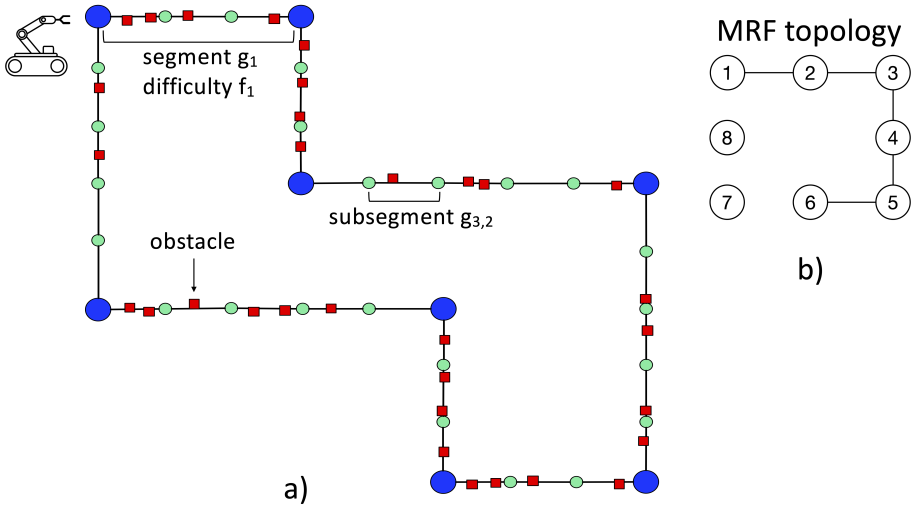

Figure 4.14: a) Instance of velocity regulation. b) True MRF topology used for the domain (Zuccotto et al., 2022b).

f	$p(oc = 1 \mid f)$
L	0.600
M	0.690
H	0.940

a)

f	$p(av = 1 \mid f)$
L	0.170
M	0.240
H	0.530

b)

f	a	$p(cl = 1 \mid f, a)$
L	S	0.000
L	I	0.033
L	F	0.033
M	S	0.000
M	I	0.033
M	F	0.067
H	S	0.000
H	I	0.067
H	F	0.100

c)

Table 4.3: Main elements of the POMDP model for the velocity regulation domain (Zuccotto et al., 2022b). a) Occupancy model $p(oc|f)$: probability of subsegment occupancy given segment difficulty. b) Angular velocity model $p(av|f)$. c) Collision model $p(cl|f, a)$: collision probability given segment difficulty and action.

In the tests on the velocity regulation domain, the *Observations* are not only related to subsegment occupancy but also to angular velocity. While the *occupancy model* $p(oc|f)$ probabilistically relates segment difficulties to subsegment occupancy, the *angular velocity model* provides the probability of angular velocity given segment difficulties, namely $p(av|f)$. More precisely, $av = 0$ means that the robot performs few curves in the subsegment while $av = 1$ that it performs several curves. In a realistic application on a mobile robot, oc is computed by averaging the values of the laser in front of the robot and applying a threshold to obtain the two binary values. Moreover, we count the actions corresponding to robot turns with angular velocity $\geq 45°$ deg/s, and threshold such count to obtain the binary signal for av. The final observation is a coding of both variables oc and av computed as $o = av+2\cdot oc$. Namely $o = 0$ if $av = 0$ and $oc = 0$, $o = 1$ if $av = 1$ and $oc = 0$, $o = 2$ if $av = 0$ and $oc = 1$, and $o = 3$ if $av = 1$ and $oc = 1$. The parameters used in our tests are summarized in Table 4.3, and Figure 4.14a shows an instance of the domain. Figure 4.14b illustrates the true MRF that we used to constrain segment difficulties. It presents five edges with the following probability values: $\mathcal{P}(1, 2) = 0.90$, $\mathcal{P}(2, 3) = 0.91$, $\mathcal{P}(3, 4) = 0.92$, $\mathcal{P}(4, 5) = 0.91$, $\mathcal{P}(5, 6) = 0.91$. Thus, admissible configurations of segment difficulties have, with high probability, all these segments with the same value while the values of segments 7 and 8 can be randomly assigned since there are no constraints on their values in the MRF.

To prove that the use of the MRF adaptation method in POMCP provides a statistically significant improvement with respect to the use of the MRF without adaptation, we show that the average difference $\overline{\Delta\rho_e}$ between the discounted return obtained with ADA and the discounted return obtained with EXCT is significantly larger than zero. Notice that the difference is computed episode by episode and the average is computed across all the

episodes of each run in which ADA is used. More precisely, at episode e, we compute the difference of discounted return as $\Delta\rho_e = \rho_e^{ADA} - \rho_e^{EXCT}$. Then we compute the average of these values over all the episodes of all the runs *average discounted return* $\overline{\Delta\rho_e}$.

4.11.2 Results on Rocksample

Figure 4.15a shows the distribution of the differences between the discounted returns obtained using ADA and those obtained using EXCT. The distribution is computed considering the 162 episodes (out of 1000, i.e., 100 episodes for 10 runs) in which the adaptation mechanism was activated (i.e., at least a discrepancy between the learned MRF and the true state has been detected). The average distance is $\overline{\Delta\rho_e} = 1.35$, which corresponds to a 6.54% improvement (see the first line of Table 4.4). The p-value of the Student's t-test guarantees that this average is significantly different from zero (Table 4.4). Figure 4.15b shows the distribution of the differences between the discounted returns obtained using ADA and those obtained using STD. This distribution is computed on 1000 values (i.e., 100 episodes for 10 runs). The average is $\overline{\Delta\rho_e} = 1.62$, which corresponds to a 7.46% improvement (see the second line of Table 4.4). Also in this case the p-value of the Student's t-test guarantees that this average is significantly different from zero (Table 4.4). Therefore, we can state that the improvement is, on average, statistically significant.

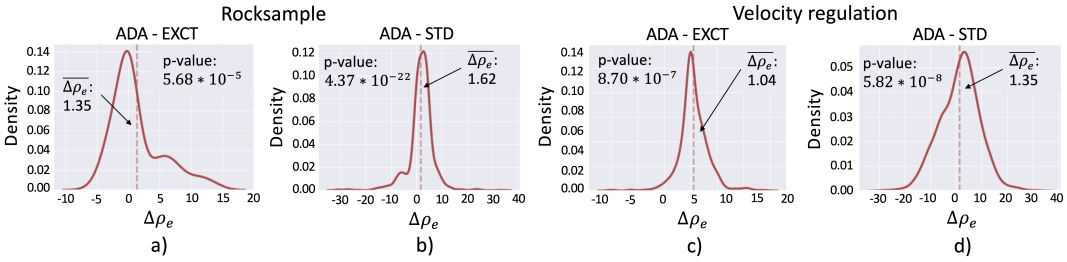

Figure 4.15: Density of difference in discounted return between a) ADA and EXCT on rocksample. b) ADA and STD on rocksample. c) ADA and EXCT on velocity regulation. d) ADA and STD on velocity regulation (Zuccotto et al., 2022b).

4.11.3 Results on Velocity Regulation

The experiments performed on the velocity regulation domain confirm the positive results obtained on rocksample. Figure 4.15c shows the distribution of the differences between the discounted returns obtained using ADA and the ones obtained using EXCT. The distribution is computed considering 714 episodes (out of 1000), i.e., the number of episodes in which the adaptation mechanism was activated. The average difference is $\overline{\Delta\rho_e} = 1.04$, corresponding to a 3.51% performance improvement (third line of Table 4.4). This average is significantly different from zero since the p-value of the Student's

Table 4.4: Performance of ADA.

Environment	Comparison	$\overline{\Delta\rho_e}(\overline{\Delta\rho_e}\%)$	p-val
Rocksample	ADA - EXCT	**1.35(6.54%)**	$5.68 \cdot 10^{-5}$
	ADA - STD	**1.62 (7.46%)**	$4.37 \cdot 10^{-22}$
Velocity regulation	ADA - EXCT	**1.04 (3.51%)**	$8.70 \cdot 10^{-7}$
	ADA - STD	**1.35 (3.34%)**	$5.82 \cdot 10^{-8}$

t-test is lower than 0.05 (see Table 4.4). Figure 4.15d shows the distribution of the differences between the discounted return obtained with ADA and the ones obtained with STD. In this case the distribution is computed on 1000 values (i.e. all the 100 episodes for all the 10 runs). The average $\overline{\Delta\rho_e} = 1.35$ and it corresponds to an improvement of 3.34% (fourth line of Table 4.4). Also in this case the p-value of the Student's t-test guarantees that the value of $\overline{\Delta\rho_e}$ is statistically different from zero (Table 4.4). Thus, the performance improvement obtained are on average statistically significant.

4.11.4 Different behavior of ADA compared to EXCT

Finally, to highlight the different behavior of ADA compared to EXCT, in Figure 4.16a we show the behavior of ADA (on the left) and EXCT (on the right) in a specific episode in which ADA is used and gives a performance improvement. Instead, in Figure 4.16d we show the behavior of a specific episode in which the use of ADA yields to a decrease in performance. In each figure, we represent on the left grid the actions performed by the agent using ADA, while on the right grid its action using the learned MRF. To denote the presence of a rock in a specific cell we use its id (from 1 to 8). The agent starting position is represented by the light blue circle and blue arrows indicate the path traveled by the agent. In pink-bordered boxes we indicate the id of the rock that the agent senses from a cell. With green boxes and red triangles, we respectively represent the fact that the agent samples a valuable or valueless rock in the corresponding cell. Finally, the orange lightning means that a discrepancy is detected and that the adaptation approach is used as previously described in Section 4.6.

Figure 4.16a shows a lighting in cell (2,2) of the left grid. The learned MRF (see Figure 4.16b) expresses a high equality probability between rock 4 and rock 3 ($\mathcal{P}(3,4) = 0.90$), thus after the valuable rock 4 has been collected the agent is encouraged to sample also the rock 3. In the true state-variable configuration, instead, rock 3 is valueless thus a discrepancy with the learned MRF is detected. Then the probability value on the edge $\mathcal{P}(3,4)$ is set to 0 because the assignments of rock 3 and 4 are different (see Figure 4.16c). Afterwards, the particle filter is re-initialized according to Algorithm 5 and the belief re-computed. The positive effect of the proposed method is clearly

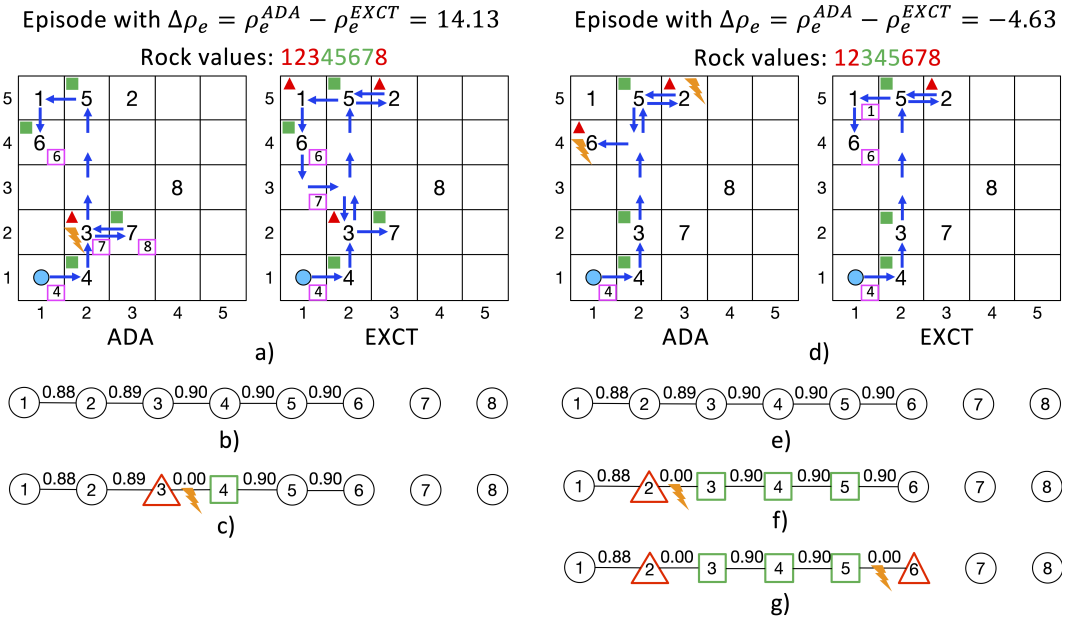

Figure 4.16: Behavior of ADA and EXCT (Zuccotto et al., 2022b). a) Relevant actions of the execution traces of an episode with a *positive* $\Delta\rho_e$ between ADA (on the left) and EXCT (on the right). b) Initial MRFs (learned). c) Adapted MRF, $\mathcal{P}(3,4) = 0$. d) Relevant actions of the execution traces of an episode with a *negative* $\Delta\rho_e$ between ADA (on the left) and the use of EXCT (on the right). e) Initial MRFs (learned). f) MRF after the first adaptation, $\mathcal{P}(2,3) = 0$. g) MRF after the second adaptation, $\mathcal{P}(5,6) = 0$.

visible since the agent does not sample rock 1 and 2. Rock 2, in fact, is related to the (valueless) rock 3 by an equality probability of 0.89 (on average) in the learned MRF so the agent is not encouraged to sample rock 2 (see Figure 4.16c). Rock 1, in turn, is related to rock 2 by $\mathcal{P}(1,2) = 0.88$ thus the agent does not sample rock 1. For the same reason rocks 5 and 6 are sampled due to the equality probability that relates the assignment of rock 5 to the valuable rock 4 and the one that relates the assignment of rock 5 to the value of rock 6 (both probabilities are 0.9 on average). On the right grid of Figure 4.16a, instead, we see what happens when the learned MRF, despite its correctness in probabilistic terms, does not reflect the state-variable configuration in the specific episode at hand. The agent samples the valueless rock 3 then, since its knowledge about the environment does not change, the agent samples also rocks 1 and 2, both valueless. In this episode, ADA allows to limit the negative effect of a misleading MRF obtaining a $\Delta\rho_e$ of 14.13.

In Figure 4.16d, instead, we depict the most relevant agent actions of an episode in which ADA performs worse than EXCT. On the left grid, we show that two discrepancies with the learned MRF (see Figure 4.16e) are detected, respectively in cell (2,5) and (1,4), thus ADA is used twice in this episode. The effect of the first usage of ADA (Figure 4.16f) consists in discouraging the agent to sample rock 1, while the second (Figure 4.16g) does not influ-

ence any other sampling action due to the fact that rock 5 has already been sampled and no other state-variable has equality relationships with rock 6. In the right grid, instead, the agent performs a sensing action on rock 6 that returns a negative response discouraging the agent to sample the rock. The different behavior of the agents about rock 6 gives a negative value for $\Delta\rho_e$, that is -4.63.

Chapter 5

Learning the Environment Model in MCTS-based planning

In this chapter, we introduce a Monte Carlo Tree Search-based planning approach in which the transition model is learned online starting from an approximated model provided by an expert. This method periodically refines the model as new information is collected from the environment. Section 5.1 outlines the motivation and scenarios; Section 5.2 introduces three model learning techniques used to integrate data acquired from the environment with prior knowledge provided by the expert; Section 5.3 introduces a real-world application domain related to air quality and thermal comfort control in smart buildings developed in the context of a project (Capuzzo et al., 2022); Section 5.4 presents the experimental setting; Section 5.5 reports experimental results.

Publication on this topic:

- M. Zuccotto, E. Fusa, A. Castellini, and A. Farinelli. Online model adaptation in Monte Carlo tree search planning. Optimization and Engineering - Special Issue on Machine Learning and Inverse Problems, 2024. doi: 10.1007/s11081-024-09896-2.

5.1 Motivation and scenarios

In real-world applications, it is not trivial to have a complete knowledge of the model of the environment. Model-free RL algorithm, such as TRPO (Schulman et al., 2015) and PPO (Schulman et al., 2017), can be applied to domains in which the environment is completely unknown. However, in several real-world applications the environment is known or partially known, thus using a model of it can significantly improve sample efficiency of RL approaches (i.e. reduce the interactions with the environment) and yield more accurate policies quicker. A known model-based algorithm is Dyna (Sutton and Barto, 2018; Sutton, 1991, 1990) which does not scale to large domains since it represents the Q-function as a table, hence it cannot be applied to real-world domains with large and continuous state and action spaces. MCTS

planning (Browne et al., 2012) and UCT (Kocsis and Szepesvári, 2006) approaches mitigate the scaling issue using sampling based strategies. To the best of our knowledge, the only attempt to perform model learning in MCTS-based planning on observable environments is that of Bayesian Adaptive Monte Carlo Planning (Guez et al., 2014, 2013, 2012), but it leads to a significant, often prohibitive, computational complexity, since it considers all possible models maintaining a belief over their parameters. The technique we propose instead performs standard learning based on a neural network model, hence providing a simpler and more sample-efficient alternative. Other works in this context concern the introduction of uncertain transition models in MDP (Adulyasak et al., 2015; Delgado et al., 2011; Bagnell et al., 2001; White and Eldeib, 1994; Satia and Lave, 1973), however their goal is not to learn the transition model.

Another approach to tackle the problem of dynamic environments can be found in Continual Learning (Lesort et al., 2020), which shares some similarities with our work. Both methods do not rely on context data to be available at once as it needs to be integrated without forgetting existing knowledge. In this context, several methods have been recently proposed to address catastrophic forgetting (French, 1999), such as Memory Outlier Elimination (MOE) (Hurtado et al., 2023) or Gated Incremental Memory (GIM) (Cossu et al., 2020). MOE identifies and removes outliers from the memory buffer by sampling from subpopulations with homogeneous labels. On the other hand, GIM incrementally adds new modules to RNNs to detect drifts in the input distribution and uses autoencoders to identify input distributions and determine the appropriate module for sequence processing. Regarding multi-agent scenarios, Agnostic Consolidation (DAC) (Carta et al., 2022) splits each learning phase into two distinct phases: adaptation, which focuses on task optimization, and consolidation, which prevents forgetting by sharing model parameters among independently trained agents.

5.2 Model learning techniques for MCTS

The methodology here proposed is composed of two main steps (points 1 and 2 in Figure 5.1). In the first step, we assume an expert provides a mathematical model to approximate the dynamics of the real environment. This model is a black box function (we do not need to have any knowledge about its implementation) that receives a state and an action and returns the next state. We perform random sampling on the overall state and action spaces, and for each state-action pair we provide it to the model and store the related next state returned by the model. In this way, we generate a dataset $\widehat{\mathcal{D}}$ containing a large amount of triplets state-action-next-state for different environment conditions (i.e., states) and actions. Then, we generate a ANN-copy $\widehat{\mathcal{T}}^{ANN}$ of the approximated model by training the ANN on dataset $\widehat{\mathcal{D}}$. The function approximation capabilities of ANN allow us to work with any type of expert's model. This first step, described in detail in the next section and shown in the horizontal rectangle on top of Figure 5.1, is performed only once. In the sec-

ond step, described in detail in the subsequent section, we use the network $\widehat{\mathcal{T}}^{ANN}$ inside MCTS to perform simulations and we update the network as new observations are collected from the environment. Three versions of this step are proposed (see the three vertical rectangles in Figure 5.1) that merge in different ways the prior knowledge in the ANN and the information in the collected data, as described in Section 5.2.2. An extension of the proposed approach is to consider a non-stationary environment. In addition to continuously refining an approximate model of the environment, the integration of continuous learning techniques could be explored to learn new information without forgetting previously acquired knowledge, thus mitigating the risk of catastrophic forgetting, as mentioned in Section 7.2.

5.2.1 Generation of ANN-copy of the expert's model

We use the expert's model $\widehat{\mathcal{T}}$ to generate the dataset $\widehat{\mathcal{D}}$ of simulated data (see the cylinder on top of Figure 5.1), i.e., a collection of triplets (s, a, s') where s and a, respectively, represent a state of the environment and an action performed by the agent, and s' is the state of system after performing the action. More precisely, we randomly choose an initial state and run the expert's model performing actions randomly sampled over the action space to generate the (s, a, s') triplets for that day. Then, we train a neural network $\widehat{\mathcal{T}}^{ANN}$ on $\widehat{\mathcal{D}}$ obtaining a copy of the expert's model. Technical details about the architecture of the ANN and the amount of data needed to learn it depends on the properties of the expert's model (examples are given in Section 5.4).

5.2.2 Transition model update

The approximated transition model $\widehat{\mathcal{T}}^{ANN}$ can be updated in several ways as the agent acts in the environment and collects information about its dynamics and reward. The three approaches here proposed, named MCP_R, MCP_S and MCP_M, integrate in three different ways the knowledge in the expert's model with the information gathered step-by-step by the agent as it observes true values of state-variables (i.e., sensor readings in real-world applications). The three model adaptation strategies are explained in the following and a flow chart for each of them is depicted at the bottom of Figure 5.1.

MCP_Real (MCP_R): update of $\widehat{\mathcal{T}}^{ANN}$ using only data from the environment.

This version stores in a dataset $\dot{\mathcal{D}}$ the triplets $(\dot{s}, \dot{a}, \dot{s}')$ of state, action and next state observed by the agent in the real-world environment. Every n time-steps MCP_R performs a re-training of the neural network $\widehat{\mathcal{T}}^{ANN}$ updating the current weights of the network using data in $\dot{\mathcal{D}}$ as a training set. The updated network is used in MCTS simulations for the next n time-steps (i.e., until the next update). This update strategy has the advantage of being

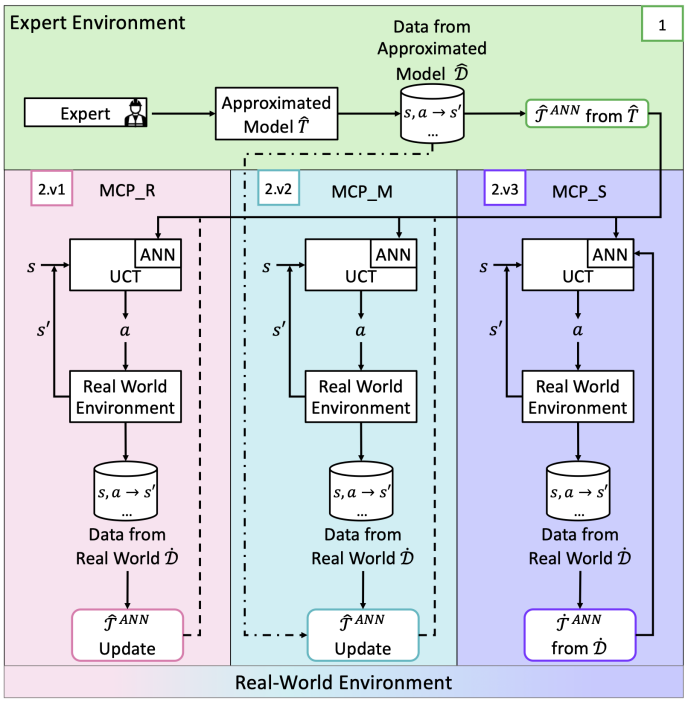

Figure 5.1: Two main steps of the proposed methodology (Zuccotto et al., 2024b). The green horizontal rectangle shows the generation of an approximated transition model (the expert's model); the pink, light blue, and purple vertical rectangles show the three versions of the model update procedure.

simple but it suffers the risk of catastrophic forgetting since the re-training is based only on data from the environment, hence the knowledge in the expert's model, stored in the weights of the initial network, can be quickly lost as weights are updated. This issue could be particularly harsh when observations from the real-world environment are very concentrated on small parts of the state-action space, as in the first update stages when $\dot{\mathcal{D}}$ contains a few observations. In these cases, the expert's knowledge about unobserved parts of the state-action space could be lost by updating the weights only considering data from the observed part of the space. The update strategies presented in the following aim to mitigate this problem, preserving expert's knowledge in parts of the state-action domains not already visited by the agent.

MCP_Mix (MCP_M): update of $\widehat{\mathcal{T}}^{ANN}$ using data from both the environment and the original expert's model.

This version stores in dataset $\dot{\mathcal{D}}$ the triplets $(\dot{s}, \dot{a}, \dot{s}')$ collected from the real-world environment during the last n steps. Then, every n time-steps, it inserts into $\widehat{\mathcal{D}}$ (i.e., the dataset generated by the original expert's model) the triplets in $\dot{\mathcal{D}}$ and removes from $\widehat{\mathcal{D}}$ the triplets that are "close enough" to the triplets just inserted. Finally, MCP_M re-trains $\widehat{\mathcal{T}}^{ANN}$ on the updated $\widehat{\mathcal{D}}$ to adapt the network weights according to both triplets observed from the en-

vironment (in parts of the state-action space where they are available) and triplets generated by the original expert's model (in parts of the state-action space where observations from the real-world environment are still not available). After the update $\widehat{\mathcal{T}}^{ANN}$ is used in MCTS simulations for the next n time-steps (i.e., until the next update).

The set of triplets in $\widehat{\mathcal{D}}$ "close enough" to a state-action pair $(\dot{s}, \dot{a}) \in \dot{\mathcal{D}}$ is defined, given the set of thresholds $\mathcal{X} = \{x_0, \ldots, x_v\}$ on state-variables (i.e., a threshold for each state-variable), as

$$\begin{aligned}&\widehat{\mathcal{U}}((\dot{s}, \dot{a}, \dot{s}'), \mathcal{X}) = \\ &\quad = \{(\widehat{s}, \widehat{a}, \widehat{s}') \in \widehat{\mathcal{D}} | (\dot{a} = \widehat{a}) \wedge (\dot{s}_0 \in [\widehat{s}_0 \pm x_0]) \wedge \cdots \wedge (\dot{s}_v \in [\widehat{s}_v \pm x_v])\}. \quad (5.1)\end{aligned}$$

Notice that we do not consider $\dot{s}'$ and $\widehat{s}'$ to compute the set of "neighbors" of a triplet since they depend on the specific transition models. Given the dataset $\dot{\mathcal{D}}$ of the last n triplets observed from the real-world environment we therefore introduce them in $\widehat{\mathcal{D}}$ and remove from $\widehat{\mathcal{D}}$ the set

$$\widehat{\mathcal{U}} = \bigcup_{(\dot{s}, \dot{a}, \dot{s}') \in \dot{\mathcal{D}}} \widehat{\mathcal{U}}((\dot{s}, \dot{a}, \dot{s}'), \mathcal{X}). \quad (5.2)$$

MCP_Select (MCP_S): update of $\widehat{\mathcal{T}}^{ANN}$ using only data from the environment and usage of the original $\widehat{\mathcal{T}}^{ANN}$ for simulated state-action pairs far from observed state-action pairs.

This version works as MCP_R in the first two steps. Namely, it stores in $\dot{\mathcal{D}}$ the triplets $(\dot{s}, \dot{a}, \dot{s}')$ observed in the real-world environment, and every n time-steps performs a re-training of $\widehat{\mathcal{T}}^{ANN}$ using only data in $\dot{\mathcal{D}}$. Afterwards, it uses two different transition models in the MCTS simulations of the subsequent n time-steps. Namely, it uses the expert model $\widehat{\mathcal{T}}^{ANN}$ or the re-trained model $\dot{\mathcal{T}}^{ANN}$, depending on the closeness of the current state-action pair in the simulation $(\tilde{s}, \tilde{a})$ to the state-action pairs of the triplets in $\dot{\mathcal{D}}$. Therefore, given a state-action pair $(\tilde{s}, \tilde{a})$ from a simulation and a set of thresholds $\mathcal{X} = \{x_0, \ldots, x_v\}$ on state-variables, we compute the set of triplets in $\dot{\mathcal{D}}$ close enough to $(\tilde{s}, \tilde{a})$ as

$$\begin{aligned}&\dot{\mathcal{U}}((\tilde{s}, \tilde{a}), \mathcal{X}) = \\ &\quad = \{(\dot{s}, \dot{a}, \dot{s}') \in \dot{\mathcal{D}} | (\tilde{a} = \dot{a}) \wedge (\tilde{s}_0 \in [\dot{s}_0 \pm x_0]) \wedge \cdots \wedge (\tilde{s}_v \in [\dot{s}_v \pm x_v])\}. \quad (5.3)\end{aligned}$$

Notice that, also in this case, we do not consider $\dot{s}'$ and $\widehat{s}'$ to compute the set of "neighbors" of a triplet since they depend on the specific transition models. If $\dot{\mathcal{U}}(\tilde{s}, \tilde{a})$ is not empty, we use the network-copy of the expert's model $\dot{\mathcal{T}}^{ANN}$ as a transition model, otherwise we use the network computed from observed triplets $\widehat{\mathcal{T}}^{ANN}$. This selection of the transition model is repeated at each step of the MCTS simulation process, apart in the first n time-steps when only the network-copy of the expert's model $\widehat{\mathcal{T}}^{ANN}$ is available.

5.3 Empirical evaluation: MCTS planning for air quality control in smart buildings

We evaluate the proposed approach on a simulated version of a real-world problem concerning air quality and thermal comfort control in smart buildings. Let's consider a room (e.g. a meeting room in a company or a classroom in a school/university) provided with sensors measuring *carbon dioxide (CO_2)* and *VOC* concentration, indoor and outdoor temperature. Ventilation and air sanification devices are used to manage air renewal and sanitization over time to avoid exceeding threshold limits. In particular, the environment has actuators for opening/closing windows or turning on/off vents and sanitizers. We also assume to have in advance the current daily occupation schedule of the room (i.e. number of persons present in the room at each hour of the day). Some examples of environments with these features are university lecture halls, in which students attend pre-scheduled lectures during the day, or meeting rooms, that must be booked in advance with the number of participants. Taking inspiration from Teleszewski and Gładyszewska-Fiedoruk (2019); Tang et al. (2016) we developed a simulator of the environment described above to test our framework in silico. The problem can be formalized as an MDP whose main elements are listed in the following.

5.3.1 State space

The *state* contains: *i*) the number of persons currently present in the room, *ii)* the current concentration of CO_2 in the room, *iii*) the current concentration of *VOC* in the room, *iv*) the current indoor temperature, *v*) the current outdoor temperature.

5.3.2 Action space

Actions are related to ventilation system, namely *stack ventilation* (open/close windows) or *mechanical ventilation* (turn on vents at low/high speed or turn off), *sanitization* system (turn on/off sanitizers) and combinations of them (see the Appendix A for full details on the action space).

5.3.3 Transition model

The *transition model* represents the model of the dynamics of the environment that we aim to improve for improving MCTS performance[1]. For each state-action pair the model returns the new state reached by performing the action from the state. The model consists of three *sub-models* used to describe the evolution of CO_2 concentration, *VOC* concentration, and indoor temperature, respectively. Each action is characterized in the transition model by a specific value called Air Change Rate (ACH) represented with δ_a, which indicates the number of times the entire room air volume is replaced in one hour

[1]In the following we describe the model we used as a true environment, then in Section 5.4 we briefly describe the error that we introduced into the expert's model.

with that action. This characteristic allows actions to affect the environment air quality (which is part of the reward function as detailed later) since the air renewal process acts on the concentrations of CO_2 and *VOC*. In the following, we provide an overview of the model. Full equations are available in Appendix A.

Evolution of CO_2 concentration. It considers both *stack* and *mechanical ventilation systems*. For stack ventilation we consider the model proposed in Teleszewski and Gładyszewska-Fiedoruk (2019) that computes the next value of CO_2 concentration i.e., $c_{CO_2}^{i+1}$, as a function of the ACH, the room volume, the number of its occupants, the concentration of CO_2, and the time difference between two time-steps. For mechanical ventilation the value of $c_{CO_2}^{i+1}$ is computed as a function of the ACH, the maximum occupancy of the room, and the number of persons in the room. We also consider *active* stack ventilation in unpopulated rooms. In this case $c_{CO_2}^{i+1}$ depends on the concentration of CO_2 and the difference between the indoor and outdoor CO_2 concentrations at time i. Model equations are available in Appendix A.

Evolution of VOC concentration. The estimated concentration of *VOC* depends on the concentration of *VOC* that each person emits in the room, the *VOC* removed by the sanitizer, the number of persons present in the room at time i, the room volume, the *VOC* concentration at time i and the relative variation of CO_2 concentration.

Evolution of internal temperature. The next value for internal temperature T_{in}^{i+1} is computed as a function of the difference between the indoor and the outdoor temperature, the time difference between two subsequent time-steps, the current indoor temperature and the heat generated by people and by sanitizers in operation. The temperature course depends on the value of the difference between the indoor and the outdoor temperature.

Evolution of room occupancy and outdoor temperature. We assume their values over time are given by the occupation schedule of the room and weather forecast.

5.3.4 Reward function

The reward function considers four components: air quality, comfort, energy consumption, and the energy factor. The air quality component corresponds to the mean between the reward due to CO_2 concentration and the reward due to *VOC* concentration when there are occupants, otherwise it is set to 1. The values of the rewards due to CO_2 and *VOC* concentrations linearly decrease when the values of c_{CO_2} and c_{VOC} are below their maximum acceptable concentration values. Conversely, these rewards quadratically decrease when their values are below the maximum concentration values we suppose the environment could reach, otherwise they are set to 0. The value of the comfort component is computed as a weighted mean between temperature and noise rewards when there are persons in the room, while it is set to 1 where there are no occupants. The temperature reward is expressed as an exponential function of the indoor temperature, while the noise reward and the value of the energy consumption reward are constant values associated

to the action performed by the agent. Thus, the noise reward represents a measure of noise pollution. Full details about the reward model are reported in Appendix A.

5.4 Experimental setting

We aim to compare the performance of our method with respect to that of baseline model-free and model-based algorithms on the air-quality control domain assuming the availability of an approximated expert's model of the environment. Since tests are performed on a simulated environment, we need a model representing the true environment (not known by the algorithm) and a second model (i.e., a modification of the first model) representing the expert's model.

5.4.1 True environment and expert's model

The transition model described in Section 5.3.3 (and Appendix A.1) is used as a true environment (i.e., correct transition model). As an approximated model (i.e., expert's model) we instead use inaccurate transition functions adding some errors to the real-world model (see Appendix A.2). In particular, the expert's model used for the *evolution of* CO_2 *concentration* overestimates CO_2 decrease and underestimates CO_2 increase. Contrary to the true environment model for the *evolution of* VOC *concentration*, the expert's model considers the effect of the action performed in terms of ACH, and it underestimates the concentration of *VOC* produced by each person in the room. Moreover, the expert's model does not consider the relative variation of CO_2 concentration. Finally, in the *evolution of internal temperature* the expert's model does not consider the heat produced by people in the room and by the sanitizers in operation, considered instead by the true model. Full details about the expert's transition models can be found in Appendix A.

5.4.2 Baseline algorithms for performance comparison

In our experiments, we compare the three versions of our approach against algorithms that represent the state-of-the-art about MCTS-based planning (two versions), model-free RL and model-based RL. The seven algorithms are briefly described in the following.

MCP_O: MCTS planning with the true transition model (oracle). This algorithm uses standard MCTS with UCT and performs simulations with a transition model identical to the true transition model. This planning method is proved to converge to the optimal policy for a large enough number of simulations and it represents a reference that we would like to reach by adapting our inaccurate model to the real model of the environment.

MCP_E: MCTS planning with the expert's transition model (without adaptation). This algorithm uses standard MCTS with UCT and performs simulations with the expert's model, which is different from the true transition

model, and it does not perform any form of adaptation. Hence, this method provides a sub-optimal policy, with actions biased by the simulation error.

PPO: Proximal Policy Optimization (model-free DRL). This algorithm uses the state-of-the-art actor-critic method in which the "actor" learns the policy given the state as input, while the "critic" learns the value of the state, i.e. it receives as input a state and it learns the value of the state given the policy that it is training. Both the "actor" and the "critic" are parametrized as neural networks.

SLBO: Stochastic Lower Bound Optimization (model-based DRL). This algorithm trains a neural network to learn the model of the environment leveraging real-world observations. Then SLBO uses the estimated model to produce samples on which it learns the policy with TRPO.

MCP_R: MCP_Real, MCTS planning with the expert's transition model as the starting model (with adaptation). This algorithm periodically updates the weights of the copy neural network of the expert's model using environment data as a training set.

MCP_M: MCP_Mix, MCTS planning with the expert's transition model as the starting model (with adaptation). This algorithm merges data (i.e., triplets state-action-next-state) generated by the original expert's model with data from the environment and periodically re-trains the copy neural network of the expert's model using this dataset as a training set.

MCP_S: MCP_Select, MCTS planning with the expert's transition model as the starting model (with adaptation). This algorithm keeps the copy neural network of the expert's model unchanged and trains a separate neural network with data from the environment, then during Monte Carlo simulations it selects the best model to perform each step according to how close is the current state-action pair to the training set by which the model has been trained.

5.4.3 Implementation details

The generation of the expert's dataset is achieved by running the expert's model with a time interval of 5 minutes for 800 days (considering different random occupation schedules in different days). We first generate a dataset containing 96000 state-action-next-state triplets i.e., data for 800 days. Then, this dataset is used to generate the network-copy of the expert's model in our method.

PPO and SLBO do not use the expert's model in their original framework, hence, to make a fair comparison we pre-trained them using the expert's model of the environment as a simulator of the real environment for the same 800 days in the expert's dataset used by our method. Moreover, in SLBO, we used the same structure of neural network used in our method to represent the model of the environment. Since we used an expert's model which has some similarities with the correct one, this process does not negatively impact the performance of PPO and SLBO but it allows them to start from higher performance. We repeated the pre-train process of a multilayer perceptron (ReLu activation function) composed of 3 hidden layers with

15, 30, 40 nodes, respectively, using 5 random seeds and kept the expert's network with the lowest validation loss to be used in MCP_R, MCP_S, and MCP_M, while for PPO and SLBO we chose the model with the highest mean cumulative reward. At this stage, we did not optimize the structure of the network but considered a model large enough to represent the structure of the transition model.

After generating the expert's model and pre-training PPO and SLBO with it, we evaluate the performance of the methods and compare them. We consider 10 rooms, each with a specific *100-day-profile* (i.e., occupation schedule and external temperature forecast). For each day, we consider a time interval from 8:00 to 18:00 with 12 samples per hour (i.e., a sampling interval of 5 minutes). Thus, each day corresponds to $10 \times 12 = 120$ samples, and each 100-day-profile has 12000 samples for room occupancy, 12000 samples for external temperature, and an initial value per day for internal temperature. After pre-training, the PPO agent is retrained at the end of each day of execution (i.e., after 120 steps), considering only real-world data acquired during that day, while the SLBO agent is retrained on samples of real-world data collected until the current time instant. Then, both the agents use the new model during the following day of execution (i.e., for 120 steps), namely until the next re-training. It is important to note that PPO, SLBO, and the MCP-based approaches are all trained for a specific number of iterations, not until convergence to avoid overfitting. To quickly simulate yearly executions, we assume each season has a length of 25 days. In different seasons we change the average external temperature. In the testing phase, for MCP_M and MCP_S, we tested 17 combinations of threshold values (i.e., sets $\mathcal{X}$, as defined in Section 5.2.2), and we kept the ones with the highest discounted return. As a stopping criterion, we used a threshold on the number of simulations in MCTS and a threshold for the number of iterations in PPO and SLBO. In particular, we stopped MCTS after 10000 simulations, PPO after 10 epochs for actor and critic, and SLBO after 75 iterations for dynamics and 50 for TRPO.

5.4.4 Performance measures

On the 10 rooms described above (100 days per room) we compute the average discounted return ($\overline{\rho_d}$) and the average difference between the discounted returns of different methods ($\overline{\Delta\rho_d}$). Averages are computed across rooms. We define ρ_d as the sum of the rewards collected in all time-steps of episode d. The episode is in general a set of steps between two transition model updates, but in our tests we consider episodes of 1 day since we update the transition model every day. The average discounted return of episode d (i.e., $\overline{\rho_d}$) is then computed as the mean of values ρ_d across all rooms, i.e., $\overline{\rho_d} = (\sum_{y=1}^{z} \rho_{d,y})/z$ where $\rho_{d,y}$ represents discounted return obtained in room y during episode d and z is the total number of rooms. The average difference between the discounted return of two methods (e.g., MCP_R and MCP_E) in episode d is instead computed as $\overline{\Delta\rho_d} = (\sum_{y=1}^{z} \Delta\rho_{d,y})/z$ where z is the total number of rooms, and $\Delta\rho_{d,y}$ represents the difference between the discounted returns

of the two methods during episode d in room y, namely, $\Delta\rho_{d,y} = \rho_{d,y}^{M_1} - \rho_{d,y}^{M_2}$, where M_1 and M_2 represent the two methods. The same measure is used to compare any pair of methods.

Time complexity

The complexity of the three versions of our approach are:

- MCP_R:
  - Model pre-train: $O(e{\cdot}|\widehat{\mathcal{D}}|{\cdot}(i{\cdot}j + j{\cdot}k + k{\cdot}l + l{\cdot}m))$
  - MCTS component: $O(d{\cdot}n{\cdot}(s^2{\cdot}b + 2{\cdot}s))$
  - Model update: $O(d^2{\cdot}e{\cdot}s{\cdot}(i{\cdot}j + j{\cdot}k + k{\cdot}l + l{\cdot}m))$
- MCP_M:
  - Model pre-train: $O(e{\cdot}|\widehat{\mathcal{D}}|{\cdot}(i{\cdot}j + j{\cdot}k + k{\cdot}l + l{\cdot}m))$
  - MCTS component: $O(d{\cdot}n{\cdot}(s^2{\cdot}b + 2{\cdot}s))$
  - Dataset update: $O(d{\cdot}(x + x{\cdot}s))$
  - Model update: $O(d{\cdot}e{\cdot}(d{\cdot}s + |\widehat{\mathcal{D}}|){\cdot}(i{\cdot}j + j{\cdot}k + k{\cdot}l + l{\cdot}m))$
- MCP_S:
  - Model pre-train: $O(e{\cdot}|\widehat{\mathcal{D}}|{\cdot}(i{\cdot}j + j{\cdot}k + k{\cdot}l + l{\cdot}m))$
  - MCTS component: $O(d{\cdot}n{\cdot}s{\cdot}(s^2{\cdot}b + s^2 + s))$
  - Model update: $O(d^2{\cdot}e{\cdot}s{\cdot}(i{\cdot}j + j{\cdot}k + k{\cdot}l + l{\cdot}m))$

All three methods share the same time complexity for pre-training the model, which is $O(e{\cdot}|\widehat{\mathcal{D}}|{\cdot}(i{\cdot}j + j{\cdot}k + k{\cdot}l + l{\cdot}m))$. This term represents the cost of pre-traing the network model based on the number of epochs e, the size of the dataset generated by the original expert's model, $|\widehat{\mathcal{D}}|$, i.e., the number of training samples, and the number of neurons in different layers (i, j, k, l, m).

The main difference between the proposed methods lies in the terms related to MCTS, the dataset update (negligible in MCP_R and MCP_S), and the model update. In MCP_R, the MCTS component scales with d (number of test days), n (number of MCTS iterations), s (number of steps), and b (branching factor). The term $s^2{\cdot}b$ is dominant here, implying a quadratic relationship with s and a linear relationship with b. The complexity is dominated by $s^2{\cdot}b$. MCP_M has the same complexity as MCP_R for the MCTS component. In addition, it introduces the term $O(d{\cdot}(x + x{\cdot}s))$ for the dataset update, the impact of which depends on the size of the dataset and the number of steps. Finally, MCP_S shows a different complexity for the MCTS component due to the selection of the best transition model at each step of the Monte Carlo simulations. This term grows significantly with the number of steps s. Regarding the model update component, in MCP_R and MCP_S, it scales quadratically with the number of test days d and linearly with the number of epochs e, steps per iteration s, and the structure of the neural network

(layers and neurons). In contrast, MCP_M includes an additional term that corresponds, in the worst case, to the entire size of the expert dataset $|\widehat{\mathcal{D}}|$, making its model update potentially more resource-intensive.

In summary, MCP_R is the most computationally efficient method, MCP_M introduces additional complexity due to dataset and model updates, and MCP_S presents the highest computational cost.

5.5 Results on the application domain

We present our results in a top-down way. First, we rank the performance of the three versions of our method to identify the best ones. Then, we compare the average performance of the best versions with that of MCP_O, MCP_E, PPO, and SLBO. Finally, we provide insight into the functioning of the proposed method and the motivation for the improvement.

5.5.1 Identification of the best versions of our approach

We compute the average difference between the discounted return obtained with each version of our approach and that achieved with MCP_E (averages are computed across the 10 rooms on which the algorithms are tested). The worst performance is achieved by MCP_R (i.e., $\overline{\Delta\rho_d} = 12.116$). This variant forgets the expert's knowledge at the first retrain of $\widehat{\mathcal{T}}^{ANN}$ and it requires several days to collect enough observations from the environment to learn a transition model not over-specialized on a small states-action subspace. MCP_S performs better (i.e., $\overline{\Delta\rho_d} = 12.997$). A motivation for this improvement is that it merges the network-copy of the expert's model with the network $\dot{\mathcal{T}}^{ANN}$ trained on observed data, in a more precise way, considering different networks in different state-action subspaces, according to the strategy described in Section 5.2.2. *The best performance is achieved by MCP_M ($\overline{\Delta\rho_d} = 13.067$).* This version is more efficient in merging the information acquired step-by-step from the environment with the knowledge in the expert's model, since it works at a training set level, i.e., it removes samples from the dataset $\widehat{\mathcal{D}}$ as new samples from the real-world environment become available. We performed Student's t-test to verify that these average differences are statistically different from zero obtaining, in all three cases, p-values lower than 0.05.

Table 5.1: Performance comparison between pairs of variants, namely MCP_R, MCP_M and MCP_S.

Variant	$\overline{\Delta\rho_d}$	p-value
MCP_R-MCP_M	-0.951	$6.101 * 10^{-5}$
MCP_R-MCP_S	-0.881	$2.341 * 10^{-4}$
MCP_S-MCP_M	-0.070	$7.380 * 10^{-1}$

In Table 5.1 we show the average difference of discounted return between each pair of versions of our approach (i.e, MCP_R vs MCP_M, MCP_R

vs MCP_S and MCP_S vs MCP_M). It emerges that both MCP_M and MCP_S perform better than MCP_R (see the first two lines of the table in which the p-value is less than 0.05), while there is no significant difference between the performance of MCP_M and MCP_S (see the third line of the table in which the p-value is 0.738 hence larger than 0.05). In the following, we fully analyze the functioning and performance of MCP_M and then briefly describe the main differences with MCP_S.

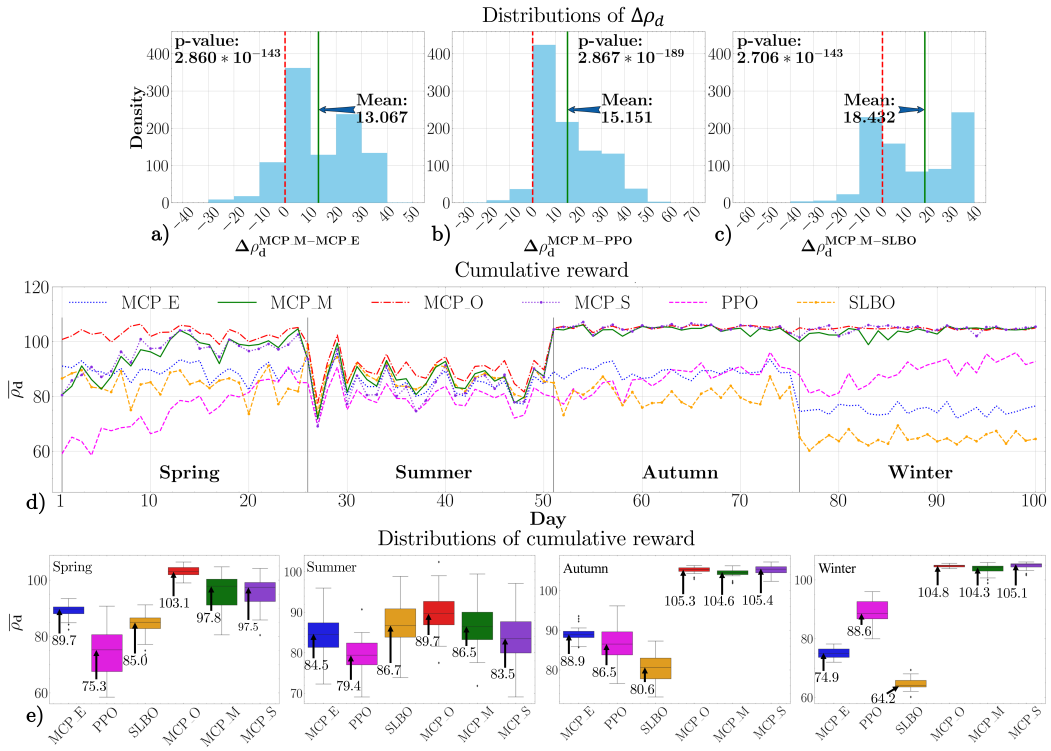

Figure 5.2: Density of difference in discounted return between *a)* MCP_M and MCP_E, *b)* MCP_M and PPO. *c)* MCP_M and SLBO. *d)* Discounted return over time in 10 rooms obtained by MCP_E, PPO, SLBO, MCP_O, MCP_M, MCP_S. *e)* Density of cumulative reward obtained by MCP_E, PPO, SLBO, MCP_O, MCP_M, MCP_S during Spring, Summer, Autumn and Winter (Zuccotto et al., 2024b).

5.5.2 Comparison of average performance: MCP_M vs MCP_E, PPO and SLBO

In this test, we compare the performance of MCP_M against baseline approaches. Figure 5.2.a compares pairs of algorithms in terms of distribution of $\Delta\rho_d$ and $\overline{\Delta\rho_d}$, where distributions and averages are computed across the 10 rooms on which the algorithms are evaluated. MCP_M outperforms MCTS with the expert's transition model (MCP_E) with an average difference of discounted return $\overline{\Delta\rho_d} = 13.067$ (as shown in the previous subsection). This is due to the improvement MCP_M introduces into the transition model over time. Figure 5.2.b shows that, on average, MCP_M performs better than PPO ($\overline{\Delta\rho_d} = 15.151$). This is a reasonable result since PPO is a model-free algorithm used in a non-trivial scenario, thus even though PPO adapts to the

environment (and it was pre-trained using the expert's model as a simulated environment), its performance is still worse than the performance of MCP_M, that can leverage the knowledge in the transition model. Finally, Figure 2.c shows that, on average, MCP_M performs better than SLBO ($\overline{\Delta\rho_d} = 18.432$). This important result shows that the proposed approach can outperform also state-of-the-art model-based approaches (note that the same neural network structure was used in both methods to represent the transition model).

5.5.3 Comparison of performance over time: MCP_M vs MCP_O, MCP_E, PPO and SLBO

Figures 5.2.d shows the average discounted return $\overline{\rho_d}$ (where the average is computed across the 10 rooms) of MCP_E, MCP_O, MCP_M, PPO, and SLBO, along the 100 days executions on which we performed our tests. The dash-dotted red line, representing the daily discounted return of MCTS with the true transition model (MCP_O), highlights the high performance of the method, which converges to the optimal policy when enough simulations are used. This result is clearly visible in Figure 5.2.e, that shows the average discounted return $\overline{\rho_d}$ of each approach, grouped by season. In Figures 5.2.d the solid green line of MCP_M starts below the dotted blue line of MCP_E. This trend reverses for the first time at day 6, confirming the result showed in Figure 5.2.a. In Figure 5.2.d we also show that the solid green line of MCP_M manages to reach the dash-dotted red line of MCP_O, stabilizing on the same values from day 50. Figure 5.2.e confirms this result, showing that MCP_M performance increase through seasons until stabilizing on that of MCP_O during autumn (median $\overline{\rho_d}$ of 104.6 and 105.3, respectively) and winter (median $\overline{\rho_d}$ of 104.3 and 104.8, respectively). Moreover, Figure 5.2.d shows that, at the beginning, the performance of PPO (dashed pink line) are the worst, but they gradually improve until they outperform MCP_E (dotted blue line) for the first time at day 64. However, PPO never reaches the performance of MCP_O (dash-dotted red line), as MCP_M does, due to a slow performance improvement. This trend is also visible in Figure 5.2.e, where PPO starts with a median $\overline{\rho_d}$ of 75.3 and it gradually increases until reaching the value of 88.6 in winter. According to Figure 5.2.d SLBO (dashed orange line with marker o) achieves the worst performance in this test. It outperforms PPO during spring and summer with median $\overline{\rho_d}$ of 85.0 and 86.7 respectively (Figure 5.2.e), then its performance decreases making SLBO the worst performing approach in autumn and winter achieving median $\overline{\rho_d}$ of 80.6 and 64.2, respectively (Figure 5.2.e). To the best of our knowledge, we attribute the poor performance of SLBO to the use of TRPO which uses a second-order optimization, which increments the overhead in the training possibly resulting in poor performance Schulman et al. (2015). These results confirm the average improvement of Figure 5.2.b,.c and provide insight on the origin of this improvement over time. Finally, Figure 5.2.d shows that, on average, MCP_M, PPO and SLBO usually present a performance decrease as the season changes (vertical black lines) due to the environment change and the need of several days to learn the new environment.

5.5.4 Performance of MCP_S

After analyzing in depth the performance of MCP_M, here we briefly analyze the performance of MCP_S (which are slightly lower than but not statistically different from that of MCP_M). We compared the performance of MCP_S against baseline approaches in terms of distribution of $\overline{\Delta\rho_d}$, computed across the 10 rooms on which the algorithms are evaluated. This test confirms that MCP_S outperforms MCP_E with an average difference of discounted return $\overline{\Delta\rho_d} = 12.997$ due to the improvement MCP_S introduces into the transition model over time. Moreover, MCP_S performs better than PPO ($\overline{\Delta\rho_d} = 15.081$) and SLBO ($\overline{\Delta\rho_d} = 18.362$) showing that our approach can outperform both state-of-the-art model-free and model-based approaches.

In Figure 5.2.d we show that the performance of MCP_S (dotted purple line with marker o) and MCP_M (solid green line) are very similar in terms of average discounted return along the 100 days executions. Figure 5.2.e confirms this result, highlighting the similarity with MCP_M in each season.

5.5.5 Additional comparison of performance over time: MCP_M vs PPO

In this section, we present the results obtained by comparing the performance of MCP, PPO updated considering only the last day's data, and PPO updated considering all data up to that instant, first over 100 days and then over 500 days.

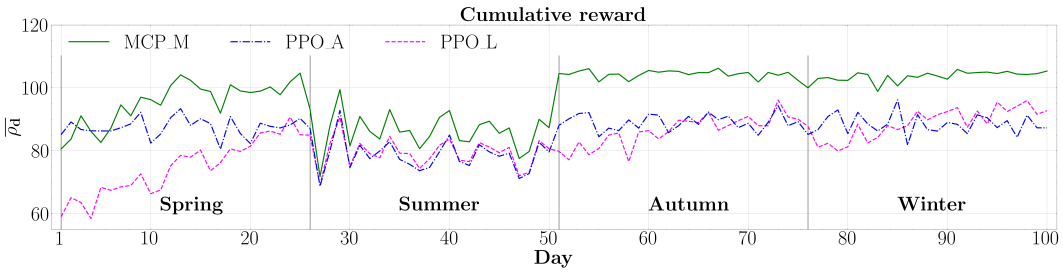

Figure 5.3: Discounted return over time in 10 rooms obtained by MCP_M, PPO trained with only the data of the last day (PPO_L), and PPO trained with all data until the current time instant (PPO_A) (Zuccotto et al., 2024b).

Performance over 100 days. Focusing on the comparison with PPO, we performed a test by retraining PPO agents each day on all real-world data collected up to that day, over 100 days (PPO_A). Then, we compare its performance with that obtained by retraining agents considering only the data of the last day (PPO_L). As shown in Figure 5.3, the performance of PPO trained on all real-world data collected up to that day, PPO_A, is improved only in the first season, but it does not reach the performance of MCP_M anyway. PPO, and other model-free RL methods, require larger amounts of data to compute an optimal policy. As Figure 5.2.d shows, PPO's performance improves slower than MCP-based proposed approaches, since they can leverage the explicit representation of the transition model.

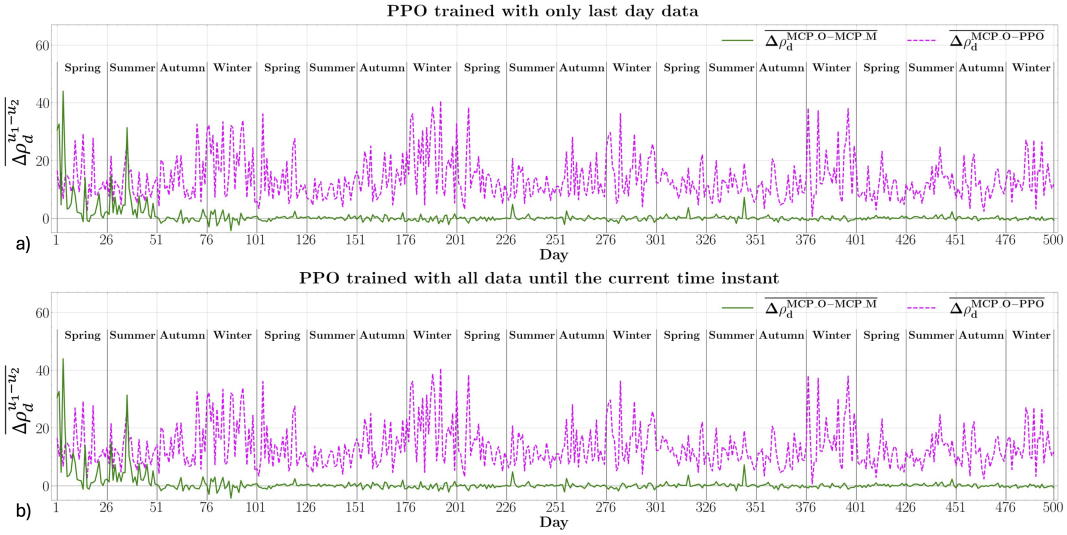

Figure 5.4: Difference in discounted return between MCP_O and MCP_M, MCP_O and PPO over a trajectory of 500 days training PPO with *a)* only last day data, *b)* all data until the current time instant (Zuccotto et al., 2024b).

Performance over 500 days. PPO manages to reach the same performance as MCP_M after many steps, showing that we used it in a fair way. To confirm this point, Figure 5.4.a shows the difference in performance between MCP_O and PPO (dashed pink line) and between MCP_O and MCP_M (solid green line) on a trajectory of 500 days. Clearly, both MCP_M and PPO achieve good performance at the end of the trajectory, but MCP_M achieves it earlier than PPO. The theoretical basis of this improvement can also be explained by a parallelism with Dyna-Q, a known model-based RL algorithm. Dyna-Q outperforms Q-learning (a model-free RL algorithm) if an approximated but meaningful transition model is used at the beginning and then updated, because DynaQ can exploit the knowledge in the approximate model. Our method can be seen as an extension of DynaQ in which the model is represented by a neural network and the Q-values are computed by MCTS (which explicitly uses the model and mixes planning and learning) instead of Q-learning. We repeated the same test training PPO with all real-world data collected up to the current time instead of using only data from the last day, but PPO obtains lower performance and it does not converge to good performance, as we can see in Figure 5.4.b.

Chapter 6

RL in Environmental Sustainability

This chapter reports the results of a survey analysis about the application of RL to environmental sustainability. This work provides a comprehensive overview of the different application domains where RL has been used, such as energy and water resource management and traffic management. The goal of this survey is to show practitioners the state-of-the-art RL methods that are currently used to solve environmental sustainability problems. Section 6.1 outlines motivation and scenarios. In section 6.2, we present the research methodology used in our survey. Section 6.3 describes the results of our research, considering different levels of detail. In particular, in Section 6.3.1, we provide a quantitative analysis of the state-of-the-art related to the application of RL in environmental sustainability over the last two decades. Then, we outline domains where RL techniques are applied and the RL-based approaches employed to address environmental sustainability. In Section 6.3.2, our focus shifts to a subset of 35 main papers for which we analyze the application domains of proposed RL techniques, provide technical insights into problem formalization, discuss the performance metrics used for evaluation, and consider the challenges addressed. Section 6.3.3 provides an in-depth analysis of each of these main papers. Finally, in Section 6.4, we discuss our findings.

Publication on this topic:

- M. Zuccotto, A. Castellini, D. La Torre, L. Mola, and A. Farinelli. Reinforcement learning applications in environmental sustainability: A review. Artificial Intelligence Review, 57, 2024. doi: 10.1007/S10462-024-10706-5.

6.1 Motivation and scenarios

AI is taking an increasingly important role in industry and society. AI techniques have been recently introduced in autonomous driving, personalized shopping, and fraud prevention, just to make a few examples. A key challenge faced by today's society for which AI can bring an important advance-

ment is environmental sustainability. Climate change, pollution, biodiversity decline, poor health, and poverty have led in the last years governments and companies to focus more and more their efforts and investments on solutions to environmental sustainability problems, which are usually characterized by an inefficient and increased use of resources. Environmental sustainability can be defined as a set of constraints regarding the use of renewable and nonrenewable resources on the one hand, pollution, and waste assimilation on the other (Goodland, 1995). In this regard, in 2015, the United Nations published the "2030 Agenda for Sustainable Development" the centerpiece of which is 17 Sustainable Development Goals (United Nations, 2015) to be fully achieved by 2030 to attain sustainable development in the economic, social, and environmental contexts, and eliminate all forms of poverty.

Various AI techniques, including dynamic programming (Angelidakis and Chalkiadakis, 2016, 2015a,b) and RL (Sultanuddin et al., 2023; Al-Jawad et al., 2021; Sheikhi et al., 2016), have been employed to tackle environmental sustainability challenges. One of the most important and interesting challenges in today's RL research is the application of RL algorithms to real-world domains, where uncertainty makes strategy learning and adaptation much more complex than in game environments. In particular, the application of RL to environmental sustainability has achieved, in the last decade, a strong interest from both the computer science community and the communities of environmental sciences and business. Reducing carbon emissions requires increasing renewable resources usage, such as solar and wind power. While these resources are economically efficient, their stochastic and intermittent nature poses challenges in replacing nonrenewable energy sources within energy networks. RL, with a systematic trial-and-error interaction with dynamic environments, offers a promising approach for learning optimal policies that can adapt to changing system dynamics and effectively manage environmental uncertainty. Thus, an RL agent is capable of handling variations in operating conditions, for instance, due to a change in resource availability or weather conditions.

6.2 Review Methodology

In this section, we outline the research methodology we used for this study. It consists of 5 steps: *i)* the definition of the research questions, *ii)* the paper collection process, *iii)* the definition of inclusion and exclusion criteria, *iv)* the identification of relevant studies based on inclusion and exclusion criteria, and *v)* data extraction and analysis.

Research questions. The first step involves defining the research questions we want to answer on the application of RL techniques for environmental sustainability. The goal of our questions is twofold: to offer a quantitative analysis of the state of the art related to the application of RL to environmental sustainability and to analyze the use of these techniques focusing on sustainability. Specifically, we aim to answer the following questions:

- RQ1: How many academic studies have been published from 2003 to

2023 about RL for environmental sustainability?

- RQ2: What were the most relevant publication channels used?
- RQ3: In which country were located the most active research centers?
- RQ4: What were the application domains and the methodologies used?
- RQ5: How was the RL problem formalized (i.e., type of state/action space, type of transition model, and type of dataset used)?
- RQ6: Which evaluation metrics were used to assess the performance?
- RQ7: What were the challenges addressed?

The databases we use to collect papers are those of the search engines Scopus and Web of Science. To limit the scope of research to the application of RL approaches for environmental sustainability, we define the following search strings, focusing on title, abstract, and keywords as search fields:

- "reinforcement learning AND sustainable AND environment";
- "reinforcement learning AND environmental AND sustainability";
- "reinforcement learning AND environment AND sustainability";
- "reinforcement learning AND environmental AND sustainable".

The search on the two databases led to a total of 375 papers, 236 collected from Scopus and 139 from Web of Science.

Selection criteria for the initial set of (181) papers. To refine the results of the search, we outline the following inclusion and exclusion criteria.

Inclusion criteria. To determine studies eligible for inclusion in this work, we consider the following criteria:

- It is written in English;
- It is clearly focused on RL for environmental sustainability;
- In the case of duplicate articles, the most recent version is included.

Exclusion criteria. To further refine our search, we apply the following exclusion criteria: the study is an editorial, a conference review, or a book chapter.

Following these criteria, we found 181 papers (104 articles, 70 conference papers, and 7 reviews). See Appendix B for details on the initial set of 181 papers. We combine the information in the index keywords of these papers with their number of citations and the publication year. In particular, we compute the number of occurrences of each keyword to identify the application domains and methodologies most used in the literature. To this aim, we standardize the keywords to avoid spelling variations. Then, we combine

these values with the number of citations and the publication year to identify the most recent and relevant studies. In cases where index keywords are missing, we use author keywords. For the only three papers that do not have author nor index keywords, we use the title as related keywords.

Selection criteria for the set of (35) main papers. To identify papers for the in-depth analysis, we applied the following criteria that consider the most important keyword occurrences (i.e., the most frequent keywords), the publication year, and the number of citations based on publication year.

- Presence of at least one keyword with no less than 10 occurrences;
- Publication year from 2013 to 2023;
- Number of citations:
  - Papers published in 2022 - 2021, at least 3 citations;
  - Papers published in 2020 - 2019, at least 10 citations;
  - Papers published in 2018 - 2013, at least 20 citations.

Following these criteria, we selected 35 studies that have been explored in-depth, and answers to the research questions defined above have been reported.

In the following sections, we first consider the initial 181 papers found using the search strings defined above and applying inclusion/exclusion criteria. In Section 6.3.1 we answer question RQ1 for those papers, in Section 6.3.1 we answer question RQ2, in Section 6.3.1 we answer question RQ3 and in Section 6.3.1 we answer question RQ4. Namely, we first analyze the number of papers that focus on RL for sustainability published in the last 20 years, then we identify the main international conferences, workshops, and journals used to disseminate research, subsequently we find the research centers that are particularly active in this research/application topic, and finally we analyze the application domains and RL methodologies used. From Section 6.3.2, we start focusing only on the main 35 papers identified using main papers selection criteria. In particular, we answer question RQ4 in Section 6.3.2, question RQ5 in Section 6.3.2, question RQ6 in Section 6.3.2, and question RQ7 in Section 6.3.2. Namely, for these main papers we first analyze the application domains of RL techniques and the RL-based approaches used to tackle environmental sustainability; then we analyze the way in which the problem has been formalized; subsequently we investigate the evaluation measures used; finally, we identify the main challenges addresses. Notice that questions RQ1, RQ2, and RQ3 have not been answered considering only the main 35 papers because these questions aim to provide a quantitative analysis of the state of the art as a whole, and this subset of articles is part of the 181 papers used to answer these three questions.

6.3 Results of the review

This section reports the results of the analysis provided in this survey, first for the initial set of 181 papers, then for the subset of the main 35 papers.

6.3.1 Analysis of the initial set of 181 papers

The initial set of papers, selected using the search strings of Section 6.2, is analyzed by answering questions RQ1, RQ2, RQ3, and RQ4.

RQ1: How many academic studies have been published from 2003 to 2023 about RL for environmental sustainability?

This research question aims to quantify the interest of the international scientific community in applying RL methods to environmental sustainability problems over the last 20 years. As shown in Figure 6.1, the number of publications (pink dots) remained relatively low until 2018, with the number of publications each year less than five. Since 2019, there has been a rapid growth of up to 53 papers in 2022, showing the increasing interest in this topic during the last few years. It is important to notice that the data for the year 2023 are updated to April 2023 and do not represent a decrease in the number of studies published. Application of inclusion and exclusion criteria leads to no publication in the year 2004, 2005, 2010, and 2011. In Figure 6.1, we also show that the increase in the number of publications fits an exponential pattern (green line) with a growth rate of 0.42 in the number of publications (from 2 papers in 2007 to 53 in 2022). To compute the regression model, we do not consider 2023 in the model since its information is partial.

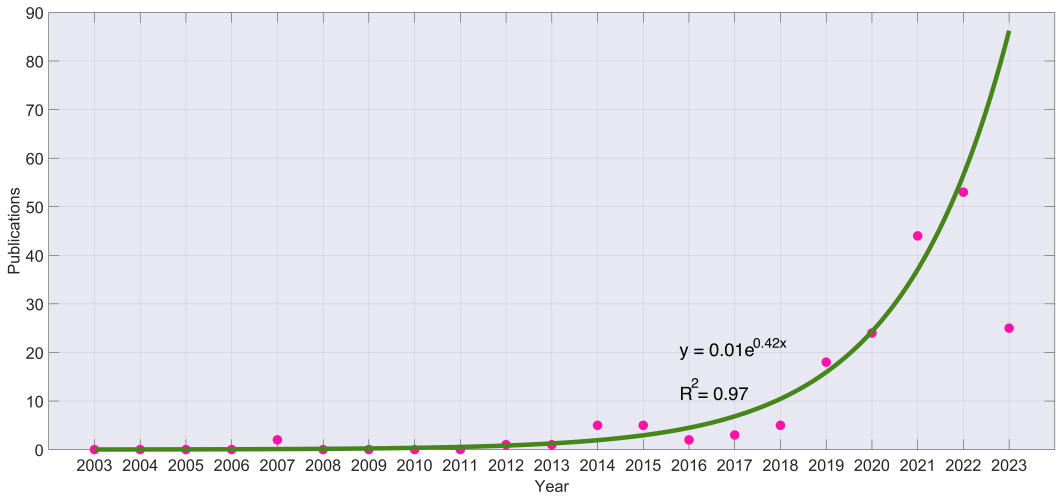

Figure 6.1: Academic studies published from 2003 to 2023. Pink dots represent the number of publications per year used to compute the regression model represented by the green line (Zuccotto et al., 2024a).

Table 6.1: Journals and conferences with at least two publications. [(*) including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics]. In the "Scope" column, "CS" and "APP" we indicate a technical/informatics or application-oriented perspective of the Conference/Journal, respectively, while "CS + APP" denotes a combination of them.

Conference	**Publications**	**Scope**
Lecture Notes in Computer Science (*)	5	CS
IEEE Conference on Intelligent Transportation Systems	3	CS + APP
International Conference on Autonomous Agents and Multiagent Systems	2	CS
IEEE International Conference on Distributed Computing Systems	2	CS
International Conference on Mobility, Sensing and Networking	2	CS + APP
IOP Conference Series: Earth and Environmental Science	2	APP
Land, Water and Environmental Management: Integrated Systems for Sustainability	2	APP
Journal	**Publications**	**Scope**
IEEE Access	7	APP
IEEE Internet of Things Journal	5	CS + APP
Sustainability (Switzerland)	5	CS + APP
IEEE Transactions on Intelligent Transportation Systems	4	CS + APP
Sustainable Cities and Society	4	CS + APP
Energies	3	APP
IEEE Transactions on Green Communications and Networking	3	CS + APP
IEEE Transactions on Vehicular Technology	3	CS + APP
Journal of Cleaner Production	3	APP
Applied Energy	2	APP
Applied Sciences (Switzerland)	2	APP
Electronics (Switzerland)	2	CS + APP
Energy and Buildings	2	APP
IEEE Sensors Journal	2	APP
IEEE Transactions on Network and Service Management	2	CS + APP
IEEE Wireless Communications	2	APP
Journal of Hydrology	2	APP
Resources, Conservation and Recycling	2	APP
Sensors	2	APP
Sustainable Energy Technologies and Assessments	2	APP

RQ2: What were the most relevant publication channels used?

With this research question, we aim to show what are the main channels used to disseminate research in the application of RL techniques to environmental sustainability problems. In Table 6.1, we show the journals and conferences with at least 2 publications. As can be seen, the topics of the journals and conferences are very varied. In particular, some of these communication channels are specific for sustainability, e.g., "Sustainability (Switzerland)" and "Sustainable Cities and Society", and many are related to environmental aspects such as "IOP Conference Series: Earth and Environmental Science" and "IEEE Transactions on Green Communications and Networking". Moreover, in the third column of Table 6.1, we provide an overview of the scope of the publication channels. To this aim, we analyze the information presented on the website of each conference and journal about its scope, indicating whether it has a technical/informatics or application-oriented perspective ("CS" or "APP", respectively) or a combination of them ("CS + APP"). As can be seen, most of the publication channels are application-oriented (2 conferences + 12 journals), followed by those that present a combined scope (2 conferences + 8 journals), finally, a few of them (3 conferences) have a more technical/informatics perspective.

RQ3: In which country were located the most active research centers?

This research question aims to show which countries whose research centers are most concerned with the application of RL methods to environmental sustainability issues. With this in mind, we leverage the information in the Scopus and Web of Science databases about the 181 papers that were not excluded by the application of inclusion and exclusion criteria. In Figure 6.2, we show only the countries with at least 5 publications and, as we can see, the highest number of papers comes from research centers located in China (33 papers), followed by the United States (29 papers), and the United Kingdom (17 papers). It is important to note that most of these works are developed in collaboration between research centers in multiple countries, so we count the paper for each collaborating country. To show co-author relationships, in Figure 6.3, we represent only countries with at least 5 occurrences among analyzed documents. Each country is depicted as a circle, a link between 2 circles represents a co-authorship relation, and the line weight is proportional to the number of papers in the co-authorship relationship. As we can see, the countries with more links are the United States (9 links), followed by Australia (7 links), China, and India (6 links).

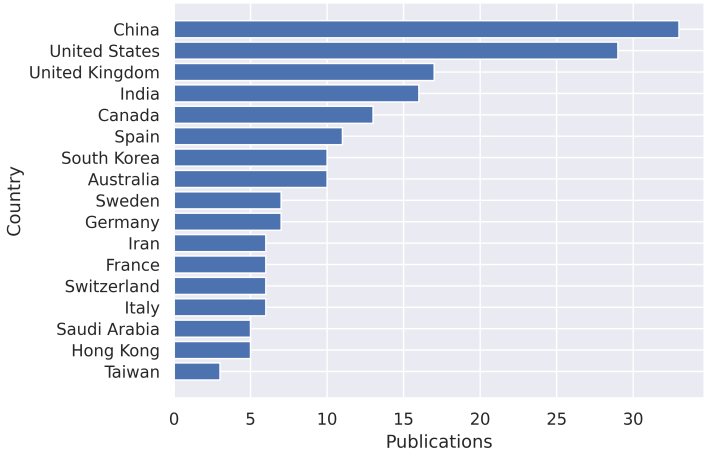

Figure 6.2: Number of publications per Country on RL approaches for environmental sustainability (Zuccotto et al., 2024a).

RQ4 (all 181 papers): What were the application domains and the methodologies used?

This research question aims to analyze the application domains and the RL methodologies used for tackling issues related to environmental sustainability. To this aim, we analyze the index keywords of the 181 papers that were not excluded by applying the inclusion and exclusion criteria and the authors' keywords for works with no index keywords. In Figure 6.4, we show the application domains with more than 10 index keyword occurrences. We group the keywords into macro areas, such as in "Energy" we include keywords

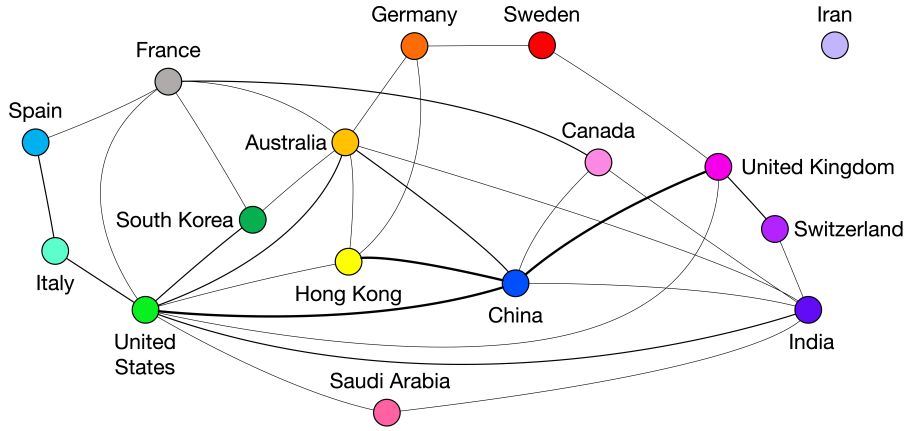

Figure 6.3: Co-author relationships with Country as a unit of analysis. Nodes represent states, and links depict co-authorship relationships. The thickness of the link is proportional to the number of papers in the co-authorship relationship (Zuccotto et al., 2024a).

like “energy”, “energy conservation”, “energy consumption”, etc., in “Electric energy” we group keywords such as “electric energy storage”, “electric load dispatching”, “smart grid”, etc. (see Appendix C for details). The image clearly shows that there is a wide variety of application domains, but most of the applications deal with sustainability issues related to energy fields.

Regarding the proposed approaches, we follow the same procedure as previously described for application domains grouping keywords that refer to the same method. For example, in “Actor-Critic” we group keywords such as “actor critic”, “advantage actor-critic (A2C)”, and “soft actor critic”. As we can see in Figure 6.5, the most widely used RL method for dealing with environmental sustainability in different application domains is a state-of-the-art model-free algorithm, namely Q-Learning (Watkins, 1989). It is important to note that, in the image, we show only RL approaches, but there are also index keywords related to other approaches like “genetic algorithm”, “simulated annealing”, etc.

Moreover, we perform a bibliometrics analysis on the co-occurrence of index keywords by using VOSviewer (Perianes-Rodriguez et al., 2016). Having a co-occurrence means that 2 keywords occur in the same work. After a data cleaning process, VOSviewer detects 17 clusters by considering keywords with 3 occurrences at least. In Figure 6.6, each cluster corresponds to a color, and each element of the cluster, namely a keyword, is depicted by a circle in the cluster color. For instance, the blue cluster is made of several blue nodes, each of which contains a keyword (e.g., electric vehicles, charging (batteries)) belonging cluster. The size of the circle and the circle label depend on the number of occurrences of the related keyword. Lines between items depict co-occurrences of keywords in a paper. Each cluster groups keywords identifying an application domain and/or the approaches used to tackle related environmental sustainability issues. For example, cluster 1 (red colored on the top-right) is somewhat related to traffic signals

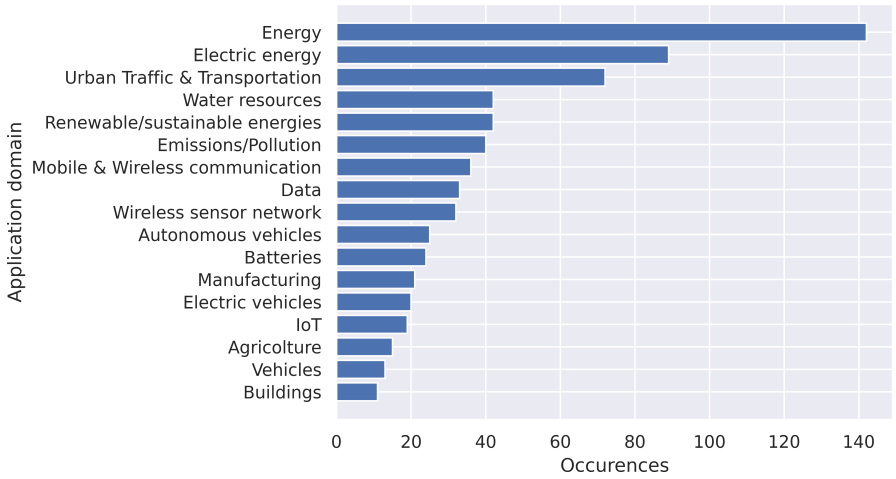

Figure 6.4: Overview of application domains. For each application domain (y-axis), we show the number of occurrences of keywords belonging to its macro-area (x-axis) (Zuccotto et al., 2024a).

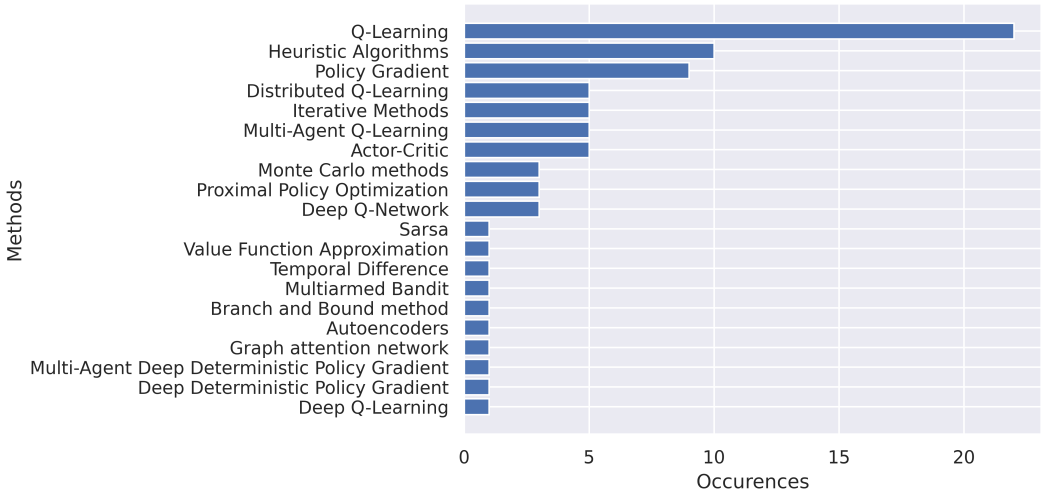

Figure 6.5: Overview of RL methods used. For each RL method (y-axis), we show the number of occurrences of corresponding keywords (x-axis) (Zuccotto et al., 2024a).

control for traffic management through the application of control strategies. Cluster 2 (green colored on the left) is related to power management and energy harvesting in wireless sensor networks.

6.3.2 Analysis of the 35 main papers

In this section, we focus on the 35 papers chosen using the selection criteria for the main papers (see Section 6.2). First, we provide a high-level analysis of the application domain and the RL approaches used to address environmental sustainability issues (research question RQ4). Then, we give an overview of the RL problem formalization (i.e., type of state/action space,

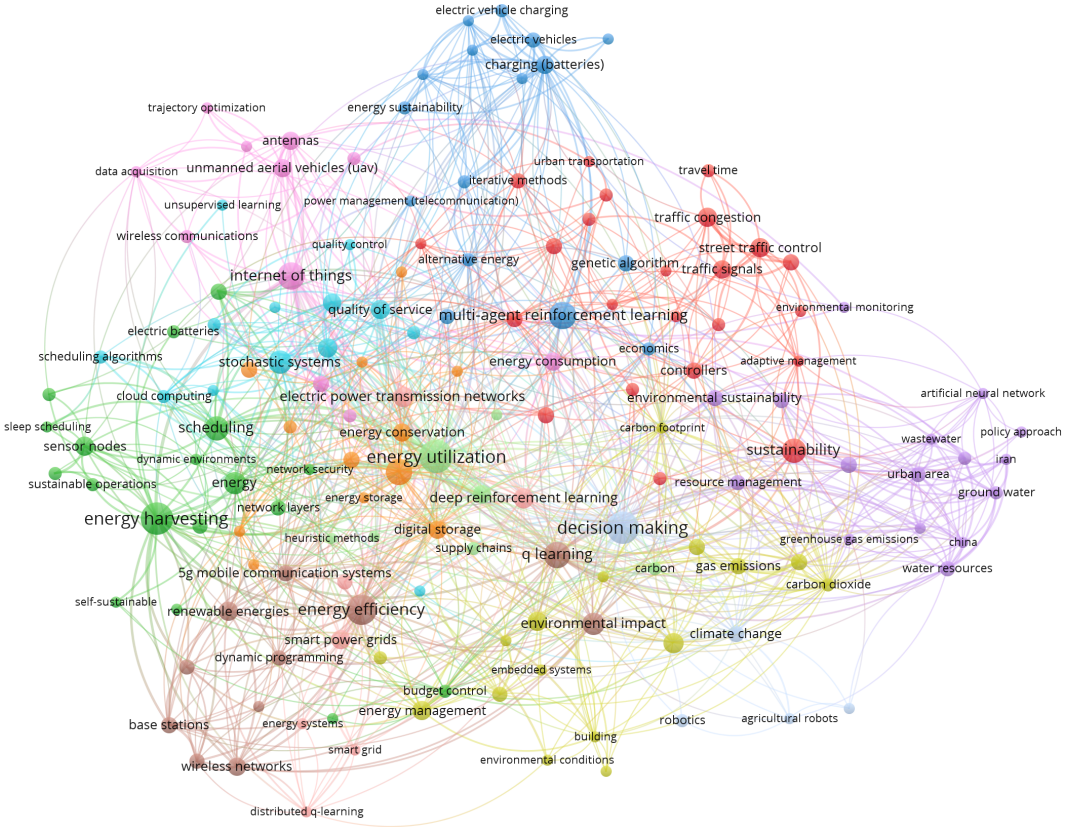

Figure 6.6: Bibliometrics analysis on the co-occurrence of index keywords. Each color outlines a cluster, and each circle of the cluster color represents a keyword, while edges represent co-occurrences of keywords in the same work (Zuccotto et al., 2024a).

transition model, glsrl method) (research question RQ5). Subsequently, we analyze the performance evaluation measures used (research question RQ6). Finally, we evaluate the main challenges faced (research question RQ7).

RQ4 (only on 35 main papers): What were the application domains and the methodologies used?

In Table 6.2, we summarize the application domains and the RL approaches used in the selected works. First, we group the 35 main works according to their main related application domains (first column). It is important to note that application domains may overlap consequently, we report all application domains common to all papers in the same group. Then, we indicate for each paper (second column) the method behind the proposed technique (third column). The selected papers tackle environmental sustainability issues in the application domains shown in Figure 6.4. In particular, the most relevant application domain correlates to the macro area of "Energy". Indeed, it involves about half of the papers in the table, considering both the works in which it represents the main application domain and those in which it is related to the main application domain. In Table 6.2, we also show that

16 out of the 35 selected papers use DRL approaches such as DQN (Mnih et al., 2015) and Double Deep Q-Network (DDQN) (Hasselt et al., 2016), and another 2 rely on DRL techniques in multi-agent contexts, such as Multi-Agent Deep Deterministic Policy Gradient (MADDPG) (Lowe et al., 2017). RL techniques are used by 10 articles, here the most used method is Q-Learning, and 7 apply RL approaches to a multi-agent context. Finally, only 1 paper adopts a Genetic Algorithm-based RL (GARL) approach.

RQ5: How was the RL problem formalized (i.e., type of state/action space, type of transition model, and type of dataset used)?

This research question deals with a technical point of view, which we think may be helpful for practitioners to get an overview of the environments considered by the authors in developing the proposed methods. In Table 6.2, we summarize the information related to problem formulation that we found in the selected papers. For each paper, we point out if the state and action spaces are continuous or discrete and whether the transition model is deterministic or stochastic. Finally, we provide information on the dataset used in the experiments, outlining whether real-world or synthetic data are used. It is important to note that not all papers explicitly provide this information. Thus, we mark with "*" all information inferred from reading the article. On the other hand, "N/A" specifies that the information available was not enough to infer the required data.

In the selected papers, most of the spaces of states and actions are discrete. Indeed, only 9 approaches use a continuous space state (third column of Table 6.2), and 6 use a continuous action space (fourth column of Table 6.2). Regarding the transition model, we can see that the model is stochastic in most cases where the information is available. In de Gracia et al. (2015), "Det." and "St." are both reported because the authors test the proposed methodology on both models. Finally, in the last column, we note that most of the experiments are performed on synthetic datasets. In fact, only 9 papers use real-world data, 6 of which combine them with synthetic data ("R + S" in the table), while 2 others use the real data to generate larger data sets from them ("S (R based)" in the table). Only Venkataswamy et al. (2023) test the proposed approach on both dataset types ("R, S" in the table).

RQ6: Which evaluation metrics were used to assess the performance?

This research question aims to provide an overview of the authors' performance measure choices to evaluate the proposed approaches in the 35 selected papers. In the second column of Table 6.3, we report information about the metrics found in the articles, which are also indicated in the in-depth analysis of each paper in Section 6.3.3. As we can see in Table 6.3, the performance measures vary widely depending on the application domain and the goal of the method proposed in each paper. For example, reward is used as a metric in 9 articles but is computed differently depending on the context. Concerning electric vehicles, in Sultanuddin et al. (2023), the re-

Table 6.2: Technical information about the selected works. In the third column, we report the RL methodology used. In the fourth and fifth columns, we indicate with "C" and "D" continuous and discrete state/action spaces, respectively. In the sixth column, "Det." and "St." denote deterministic and stochastic transition models respectively. In the seventh column "S" and "R" indicate synthetic and real-world datasets, respectively

Application Domain	Paper	Method	State	Action	Tr. Model	Dataset
Electric vehicles, Batteries, Energy	Sultanuddin et al. (2023)	DDQN	N/A	D	St.	S
	Zhang et al. (2021a)	MA AC	N/A	D*	Det.*	R
IoT	Ajao and Apeh (2023)	Q-Learning	D*	D*	St.	S
	Zhang et al. (2021b)	Adaptive regression	D*	C*	Det.*	S
	Han et al. (2020)	MAB	N/A	D*	St.	S
Water resources	Emamjomehzadeh et al. (2023)	Q-Learning	N/A	N/A	N/A	S
	Skardi et al. (2020)	Q-Learning	N/A	N/A	Det., St.	R + S
Emissions/Pollution	Chen et al. (2021)	MADDPG	C	C	N/A	S
	Huo et al. (2023)	Multi-agent Q-Learning	D*	D	N/A	S
Agriculture	Elavarasan and Durairaj Vincent (2020)	DRQN	D*	N/A	N/A	R + S
Data, Energy	Shaw et al. (2022)	Q-Learning, SARSA	D*	D*	St.	S (R based)
	Venkataswamy et al. (2023)	AC	D*	D	N/A	S, R
Urban traffic & Transportation	Ounoughi et al. (2022)	DQN	N/A	D*	N/A	R
	Alizadeh Shabestray and Abdulhai (2019)	DQN	D*	D	St.	S
	Aziz et al. (2018)	RMART	D*	D*	St.*	S
	Khalid et al. (2023)	DQN	D*	D*	Det.*	S
Buildings, Energy	Kathirgamanathan et al. (2021)	SAC	C	N/A	N/A	R + S
	de Gracia et al. (2015)	SARSA(λ)	D*	D*	Det.	S
Manufacturing	Wang and Wang (2022)	Policy Network	D*	D*	Det.*	S*
	Leng et al. (2021)	Q-Learning*	N/A	D*	St.*	S (R based)
Mobile & Wireless communication, Energy, Renewable/sustainable energies	Liu et al. (2021)	DDPG	C	C	N/A	S
	Miozzo et al. (2015)	Distributed Q-Learning	D*	D	N/A	S
	Miozzo et al. (2017)	Distributed Q-Learning	D*	D	N/A	S
	Giri and Majumder (2022)	Deep Q-Learning	C*	D*	St.*	S
Mobile & wireless communication	Al-Jawad et al. (2021)	Q-Learning	D*	D*	N/A	S
Energy, Electric energy	Sheikhi et al. (2016)	Q-Learning	N/A	N/A	St.	S
	Harrold et al. (2022)	Rainbow DQN	C*	D	N/A	R + S
	Jendoubi and Bouffard (2022)	MADDPG	N/A	C	St.*	S*
Energy	Gao et al. (2023)	Q-Learning	N/A	D*	St.*	R
Wireless Sensor Network, Energy, Renewable/sustainable energies	Hsu et al. (2014)	Q-Learning*	D*	D*	N/A	S
	Chen et al. (2016)	Q-Learning	D*	D*	St.*	R + S
	Feng et al. (2023)	DDPG	C	C	St.*	S
Autonomous vehicles	Bouhamed et al. (2020)	DDPG	C*	C	N/A	S
		Q-Learning	D*	D*		
	Sacco et al. (2021)	AC	C*	D*	N/A	R + S
	Gu et al. (2023)	Policy Gradient	C*	D	St*	S

Table 6.3: We summarize the performance measures used by the authors to evaluate the proposed approaches in the second column and the challenges they address in the third column. The papers are grouped according to the main related application domains, as in Table 6.2. Performance measures written in italics are used in both Miozzo et al. (2015) and Miozzo et al. (2017).

Paper	Performance Measures	Challenges
Sultanuddin et al. (2023)	Reward, voltage levels, load curves, charging/discharging curves	Grid overload prevention, driving pattern uncertainty, dimensionality
Zhang et al. (2021a)	MCWT, MCP, TSF, CFR	Dimensionality, coordination and cooperation among agents, charging requests competitiveness, joint optimization of multiple optimization objectives
Ajao and Apeh (2023)	Detection accuracy, recall, precision, specificity, F-measure	Security threat to sustainability functionality
Zhang et al. (2021b)	Operational logs, power wastage, power requirements, average failure ratio	Energy requirements management, smart power allocation
Han et al. (2020)	Number of ready nodes, throughput	Spatial uncertainty
Emamjomehzadeh et al. (2023)	Water table level, nitrate concentration, energy usage, GHG emissions	WEF nexus modeling and management for an urban area integrated water resource management
Skardi et al. (2020)	Water and wastewater allocation, water and groundwater level, nitrate concentration	Social attachments quantification and consideration, cooperation among agents
Chen et al. (2021)	Reward, Q-values, influents, inflow rate, DO and dosage values, energy consumption, cost, EP, and GHG emissions	WWTP impact optimization
Huo et al. (2023)	Productivity, operational mistakes, GHG emissions, queuing time	Operational randomness and uncertainties
Elavarasan and Durairaj Vincent (2020)	R2, MAE, MSE, RMSE, MedAE, MSLE, MAPE, PDF, explained variance score, accuracy	Mapping between raw data and crop yield values, effectiveness dependence on extracted features quality
Shaw et al. (2022)	Energy consumption, SLAV, number of migrations, ESV	Energy awareness, slow convergence to optimal policy
Venkataswamy et al. (2023)	Monetary job value	Intermittent power, environment nonuniformity, system design and configuration effects, learning and improving available heuristic policies
Ounoughi et al. (2022)	MSE, MAE, noise levels, CO_2 emission, fuel consumption	Sustainability and proactivity integration
Alizadeh Shabestray and Abdulhai (2019)	Average of intersection travel time, queue time, and network travel time, weighted average intersection person travel time	Regular and transit vehicles consideration
Aziz et al. (2018)	Average delay, stopped delay, number of stops, and network-wide delay, GHG emissions	Traffic congestion information sharing, reward function dynamic adaptation
Khalid et al. (2023)	Execution time, reward, path planned, distance	Quality of experience assurance, optimal order user serving, distance minimization

Paper	Performance Measures	Challenges
Kathirgamanathan et al. (2021)	Energy purchased and cost, discomfort, reward, temperature, power demand	Robustness, scalability, lack of well-established environments
de Gracia et al. (2015)	Electrical energy saving	Energy saving maximization, thermal energy storage optimization
Wang and Wang (2022)	Overall Nondominated Vector Generation, C Metric, Hyper volume, $D1_R$	Energy awareness with simultaneous makespan and energy minimization
Leng et al. (2021)	MSE, MSLE, RMSE, R2, unit and total profit, acceptance rate	Demand uncertainty, order customization
Liu et al. (2021)	Achievable rate, transmission power	Effective communication, IRS phase optimization
Miozzo et al. (2015)	*Throughput gain, traffic drop rate, energy efficiency, energy efficient improvement*, traffic demand, harvested energy, battery level, policy, normalized load at the macro bs, total energy spent, average load, average cell load for the macro bs battery outage, Jain's fairness index	Energy harvesting
Miozzo et al. (2017)	Switch-off rate, battery level, excess energy	
Giri and Majumder (2022)	Reward, capacity, network lifetime, average delay	Dimensionality, efficient use of collected energy for QoS
Al-Jawad et al. (2021)	Throughput, packet loss, rejected flows, PSNR, MOS	Sustainable QoS
Sheikhi et al. (2016)	Storage charge level, operational cost, primary energy involved	Energy system parameter variability or stochasticity, smart grid architecture issues
Harrold et al. (2022)	MAPE, energy cost savings, relative savings, episodic rewards, and value distribution	Energy arbitrage, renewables usage improvement, limited data availability
Jendoubi and Bouffard (2022)	Annual total cost, daily operation cost, PV production, aggregated demand, power to be charged/discharged, generator provided power, provider delivered electricity, PAR	Energy dispatch
Gao et al. (2023)	Working and non-working energy consumption, the ratio between working and non-working energy consumption	Energy consumption minimization
Hsu et al. (2014)	RBE, EDC, OTRT, ToD achievability	Simultaneous achievement of throughput on demand satisfaction and power consumption reduction
Chen et al. (2016)	Nodes potential energy, network lifetime, area coverage ratio, number of residual alive nodes versus the network lifetime, recharging cycle	Simultaneous area coverage and energy balancing
Feng et al. (2023)	Distribution of the ratio between the expected per-slot harvested energy and the MS-to-sink distance within the network, moving trajectories and steps, reward, actor-loss, battery level, accuracy, convergence, training time	Lack of energy-related information, energy harvesting and data transmission trade-off
Bouhamed et al. (2020)	Path followed, reward, battery level, completion time of the tour against the ground unit transmission power	Limited battery capacity, obstacle awareness
Sacco et al. (2021)	Task completion time, utility, average node-antenna distance, energy consumption, task completion time against the average computing workload, CDF and utility evolution	Task completion time reduction, energy efficiency
Gu et al. (2023)	Energy loss, collision loss, reward, lane changes	Efficient response to environmental observations

ward corresponds to a penalty function considering the cost of charging and a departure incentive. Instead, in Wastewater Treatment Plants (WWTPs), Chen et al. (2021) use a reward function that takes into account the operational cost, consisting of multiple components, such as energy cost and biogas price, and several indicators, like energy consumed by the aeration and sludge treatment processes and GHG emissions. Another performance measure common to multiple application domains is, for example, energy consumption. Indeed, it is used in contexts such as water resources management (Emamjomehzadeh et al., 2023), WWTPs (Chen et al., 2021), data centers (Shaw et al., 2022), and AVs (Sacco et al., 2021). Even approaches related to the same application domain may differ in terms of performance measures depending on their objective. Considering, for example, the water resources context, both Emamjomehzadeh et al. (2023) and Skardi et al. (2020) evaluate their proposed approaches using resource level and nitrate concentration. However, in Emamjomehzadeh et al. (2023), energy consumption and GHG emissions are also considered, while in Skardi et al. (2020), resource allocation is used.

RQ7: What were the challenges addressed?

This research question aims to offer an overview of the issues that the authors have tackled within the 35 selected papers. In the third column of Table 6.3, we summarize information about the challenges addressed in the articles, which are also indicated in the in-depth analysis of each paper in Section 6.3.3. As with the performance measures, we can see in Table 3 that the challenges faced vary greatly depending on the application context and the goal of the method proposed in each paper. As an example, considering the domain of electric vehicles, Sultanuddin et al. (2023) address several challenges, like avoiding network energy overload at peak times, considering the uncertainty of driving patterns, and managing large state spaces. On the other hand, in addition to the challenge related to dimensionality, Zhang et al. (2021a) also address issues related to coordination and collaboration among agents, the competitiveness of charging demands, and joint optimization of multiple objective functions. However, although not explicitly stated by the authors, the challenge that unites these papers is the development of approaches capable of adapting to changes in a dynamic environment and managing the uncertainty associated with the environment that, in many cases, arises from the use of renewable resource sources, which have a stochastic and intermittent nature whose management adds further complexity to the problem.

6.3.3 Analysis of single papers (grouped by application domain)

In this section, we group the 35 main papers by application domain and analyze each single paper answering research questions RQ4, RQ5, RQ6, and RQ7. This provides the reader interested in a specific application domain with a deep knowledge of the main features of these papers. Notice that in

answering RQ5, we use the information available in Table 6.2 and report in the text a "(*)" for all information inferred from reading the article.

Electric vehicles, Batteries, Energy

The transportation system is characterized by an increasing presence of EVs due to their eco-friendly features. In Sultanuddin et al. (2023), it is proposed a DDQN-based approach to provide a smart scalable charging strategy for EV fleets that ensures all cars have sufficient charging for their trips without exceeding the maximum energy threshold of the power grid. The charging management system combines information on the current state of the network and vehicle with historical data, being able to schedule charging at least 24 hours in advance. In developing the proposed approach, it is considered an environment with discrete actions and a stochastic transition model. The experimental evaluation is performed on a synthetic dataset by using as metrics the reward, the voltage levels, the load curves, and the charging/discharging curves. The rapid growth in the popularity of EVs subjects the power grid infrastructure to challenges, such as preventing grid overload at peak times. Moreover, the authors address issues related to driving pattern uncertainty and handling large state spaces.

Zhang et al. (2021a) propose a framework for charging recommendations based on MARL, called Multi-Agent Spatio-Temporal Reinforcement Learning (MASTER). By leveraging a multi-agent actor-critic framework with Centralized Training and Decentralized Execution (CTDE), the proposed approach increases the collaboration and cooperation among agents, and it can make use of information about possible future charging competition through the use of a delayed access strategy. The framework is further extended to multi-critics for addressing multiple objective optimizations. MASTER works in environments characterized by discrete actions (*) and a deterministic transition model (*), and it has been tested on a real-world dataset. To evaluate its performance, the Mean Charging Wait Time (MCWT), Mean Charging Price (MCP), Total Saving Fee (TSF), and Charging Failure Rate (CFR) are used as performance measures. In the development of the proposed charging recommendation approach, the authors face several challenges such as dealing with large state and action space, coordination and cooperation among agents in a large-scale system, potential competitiveness of future charging requests, and the joint optimization of multiple optimization objectives.

IoT

Recent years have seen rapid advances in IoT technology enabling the development of smart services such as smart cities, buildings, and oceans. Regarding smart cities, Ajao and Apeh (2023) consider the Industrial Internet of Things and present a framework for edge computing vulnerabilities. Indeed, edge computing security threatens the sustainability functionality of urban infrastructure with various attacks, such as Man-in-the-Middle and denial of service. In particular, to tackle authentication and privacy violation prob-

lems, this work proposes a secure framework modeling in Petri Net, namely Secure Trust-Aware Philosopher Privacy and Authentication (STAPPA), on which a Distributed Authorization Algorithm is implemented. Moreover, a GARL approach is developed to optimize the network during learning, detect anomalies, and optimize routing. This work regards an environment characterized by discrete state and action spaces (*), and a stochastic transition model. The authors test the proposed approach on a synthetic dataset and assess the performance in anomaly detection and detection accuracy by using the popular detection accuracy, recall, precision, specificity, and F-measure. Ajao and Apeh (2023) deal with security challenges, in particular authentication and privacy violation problems.

Zhang et al. (2021b) propose an IoT-based Smart Green Energy (IoT-SGE) management system for improving the energy management of power grids allowed by DRL. The proposed approach is able to balance power availability and demand by keeping grid states steady, thus reducing power wastage. In developing IoT-SGE, it is considered an environment with discrete states (*), continuous actions (*), and a deterministic transition model (*). The proposed approach has been evaluated on a synthetic dataset by the use of operational logs, power wastage and requirement, and average failure ratio as metrics. The authors address an energy sustainability issue, in particular, they aim to manage energy requirements and allocate smart power systems.

In the context of smart ocean systems, Han et al. (2020) present an analytical model to evaluate the performance of an Internet of Underwater Things network with energy harvesting capabilities. The goal of this work is the maximization of IoT nodes throughput by optimally selecting the window size. To this aim, the authors propose an RL approach and leverage the Branch and Bound method to solve the optimization problem by autonomously adapting random access parameters through interaction with the network environment. Considering a realistic scenario, it is proposed a MARL approach to deal with the lack of network information. In this case, random access parameters autonomously adapt by using a distributed Multi-Armed Bandit (MAB)-based algorithm for each node. The environment considered in this work is characterized by deterministic actions (*) and a stochastic transition model. The authors test the proposed approach on a synthetic dataset, evaluating its performance in channel access regulation in relation to the number of ready nodes per time slot and throughput. Finally, this work addresses a fairness issue due to spatial uncertainty in underwater acoustic communication, to deal with which the authors formalize an optimization problem for maximizing the IoT network nodes throughput.

Water resources

Water resource management is a key aspect of sustainable development and usually does not include social aspects. Emamjomehzadeh et al. (2023) propose a novel urban water metabolism model that combines urban metabolism with the Water, Energy, and Food (WEF) (Radini et al., 2021) nexus and thus it can consider interconnections among water, energy, food, material, and

GHG emissions. Moreover, this work proposes a physical-behavioral model that relates the proposed approach to a MARL agent-based model neither fully cooperative nor fully competitive developed using Q-learning. In this case, the only technical information available concerns the use of a synthetic dataset. The proposed approach is evaluated in terms of water table level, nitrate density, energy usage, and GHG emissions. Considering water resource management challenges related to sustainability, the authors aim to model and manage the WEF nexus for Integrated Water Resource Management in an urban area, taking into account stakeholders' characteristics.

Skardi et al. (2020) propose, instead, an approach for quantifying and including social attachments in water and wastewater allocation tasks. This work proposes a paired physical-behavioral model, and the authors leverage Q-Learning to include social and behavioral aspects in the decision-making process. Specifically, it uses the approach proposed by Bazzan et al. (2011) to integrate Social Analysis in Q-Learning, and they choose between individual or social behavior through the use of specific reward functions. In developing the proposed method both a deterministic and a stochastic transition model is considered. Tests are performed on a dataset that combines real-world and synthetic data, and the performance evaluation is conducted considering water and treated wastewater allocation to the agents, water and groundwater level, and the concentration of nitrates to measure groundwater quality. Using Social Network Analysis, the authors tackle a key challenge in common resource management, i.e., the cooperation among agents. Also, they aim to quantify and include social attachments in water resource management.

Emissions/Pollution

The development of WWTPs has a positive impact on environmental protection by reducing pollution but, at the same time, they consume resources and produce GHG emissions as well as residual sludge. With this in mind, Chen et al. (2021) propose an approach based on MADDPG to control Dissolved Oxygen (DO) and chemical dosage at once and improve sustainability accordingly. Specifically, the proposed approach uses two agents, one to control DO and one to control chemical dosage. Moreover, a reward function is designed based on life cycle cost and various Life Cycle Assessment mid-point indicators respectively. The proposed approach is developed considering an environment with continuous state and action spaces and tested on a synthetic dataset. To evaluate the training process, the reward and the Q-values determined by trained critic networks are used as metrics, while to analyze the variation of the influents and control parameters, the authors leverage the influents (COD, TN, TP, and NH_3-N), inflow rate, DO and dosage values. Finally, to assess the impact of the proposed approach, energy consumption, cost, eutrophication potential (EP), and GHG emissions are used. WWTPs have a positive impact on environmental protection since they reduce contaminants and environmental pollution. However, at the same time, WWTPs consume resources and produce GHG emissions and residual sludge, thus the authors seek to optimize their impact on environmental sustainability.

Intelligent fleet management is crucial in mitigating direct GHG emissions in open-pit mining operations. In this context, Huo et al. (2023) propose a MARL-based dispatching system for reducing GHG emissions. To this aim, this work presents an environment for haulage simulation that integrates a component for real-time computing of GHG emissions. Then, Q-Learning is leveraged to improve fleet productivity and reduce trucks' emissions by decreasing their waiting time. In the development of the proposed approach, an environment characterized by discrete state (*) and action spaces is considered. Tests are performed on a synthetic dataset and productivity, number of operational mistakes, GHG emissions, and time spent in queue are used as evaluation metrics. In this work, the authors tackle operational randomness and uncertainties in fleet management for reducing haul trucks' GHG emissions in open-pit mining operations

Agriculture

In the context of sustainable agriculture, one of the key aspects of food security is crop yield prediction. Elavarasan and Durairaj Vincent (2020) tackle this problem by using a DRL approach, specifically a Deep Recurrent Q-Network (DRQN) (Hausknecht and Stone, 2015) model. It consists of a RNN (Rumelhart et al., 1986) on top of the DQN. The proposed approach sequentially stacks the RNN layers, feeds the network with pre-trained parameters, and adds a linear layer to map the RNN output into Q-values. The Q-Learning network builds a crop yield prediction environment as a 'yield prediction game' that leverages both parametric feature combinations and thresholds useful in agricultural production. The authors consider an environment with discrete states (*) and test their approach on a dataset combining real-world and synthetic data, evaluating the performance by using the following metrics: Determination Coefficient (R2), Mean Absolute Error (MAE), MSE, Root Mean Squared Error (RMSE), Median Absolute Error (MedAE), Mean Squared Logarithmic Error (MSLE), Mean Absolute Percentage Error (MAPE), Probability Density function (PDF), Explained Variance Score, and accuracy. Finally, Elavarasan and Durairaj Vincent (2020) address issues related to the application of DL methods to crop yield prediction for increasing food production. Specifically, the authors tackle the incapability of DL approaches to directly map, linearly or non-linearly, raw data with crop yield values and the strong dependence of their effectiveness on the quality of features extracted from data.

Data, Energy

Data centers are among the largest consumers of energy. In Shaw et al. (2022), an RL-based Virtual Machine (VM) consolidation algorithm named Advanced Reinforcement Learning Consolidation Agent (ARLCA). Its aim consists of simultaneously improving energy efficiency and delivery service guarantees. In this work, a global resource manager constantly monitors the state of the system and identifies hosts that may be overloaded due to

the resource demand change over time. The proposed approach rebalances the VM distribution and avoids the rapid overloading of hosts while ensuring efficient operation. This work presents two implementations of ARLCA based on two RL methods, i.e., Q-Learning and SARSA, and it tests two different approaches to balance the exploration-exploration tradeoff, namely ϵ-greedy, and softmax. Finally, the authors leverage the Potential Based Reward Shaping (Ng et al., 1999) technique to include domain knowledge in the reward structure and speed up the learning process. ARLCA works in an environment with discrete state and action spaces (*) and a stochastic transition model. Its performance is evaluated on a synthetic dataset (real-world-based). To evaluate the proposed VM consolidation algorithms, energy consumption, Service Level Agreement Violations (SLAV), number of migrations, and Energy Service Level Agreement Violations (ESV) are used as performance measures. In this work, the authors tackle a key challenge for cloud computing services, namely energy awareness. Further, they also face the slow convergence to the optimal policy of conventional RL algorithms.

Renewable energy Aware Resource management (RARE), a DRL approach for job scheduling in a green data center, is presented in Venkataswamy et al. (2023). This work proposes a customized actor-critic method in which the authors use three Deep Neural Networks (DNNs): the encoder, the actor, and the critic. The encoder summarizes information about the state of the environment into a compact representation of it, used as input for both the action and the critic. The actor returns the probability of choosing each scheduling action, while the critic estimates, for each action, the total expected value achieved by starting in the current state and applying a specific action. Moreover, since DRL requires a significant amount of interactions with the environment to explore it and then to adapt a randomly initialized DNN policy, the authors leverage an offline learning algorithm, namely, Behavioral Cloning, to learn a policy based on existing heuristic policy data used as prior experience. In particular, the actor network is trained to imitate the action selection process of data within the replay memory. In developing RARE, it is considered an environment characterized by discrete states (*) and actions and tested the performance on both synthetic and real-world datasets by using the total job economic value as metrics. In this work, the authors tackle several challenges related to the application of RL techniques to the context of green datacenters. The first issue relates to the environment. The dynamic of green data center environments makes the scheduling process difficult as it has to consider and manage the intermittent and variable nature of renewable energy sources. Moreover, the lack of uniformity in the environments makes it challenging to compare different approaches. The second challenge highlights the absence of discussion regarding the systems design choices effect (e.g., the planning horizon size). Such lack does not help to clarify the reasons for the better performance of the RL scheduler over heuristic policies. Furthermore, the authors discuss employing RL schedulers as a black box, without considering different configurations, such as the size of the neural network, which can lead to improved performance.

Finally, the last challenge highlights that existing RL schedulers do not focus on learning and improving available heuristic policies.

Urban traffic & Transportation.

In recent years, the traffic congestion level has increased significantly with a consequent negative impact on the environment. Ounoughi et al. (2022) present EcoLight, an approach for controlling traffic signals based on DRL, which aims to reduce noise pollution, CO_2 emissions, and fuel consumption. The proposed method combines the Sequence to Sequence Long Short Term Memory (SeqtoSeq-LSTM) prediction model with the DQN algorithm. SeqtoSeq-LSTM is used to forecast the traffic noise level that is part of the traffic information given as input to the DQN to determine the action to perform. EcoLight works in environments with discrete actions (*) and has been tested on a real-world dataset. The performance of EcoLight is evaluated by using the MSE, MAE, noise levels, CO_2 emission, and fuel consumption as metrics. In this work, the authors tackle the issue of developing a control method that considers not only mobility and current traffic conditions but also integrates sustainability and proactivity.

On the other hand, Alizadeh Shabestray and Abdulhai (2019) present Multimodal iNtelligent Deep (MiND), a DRL-based traffic signal controller that considers both regular vehicles and public transit and leverages sensors' information, like occupancy, position, and speed, to optimize the flow of people through an intersection by using DQN. In developing MiND, the authors regard an environment characterized by discrete states (*) and action and a stochastic transition method and test the proposed approach on a synthetic dataset. To assess the performance of the proposed approach the following measures are used: average intersection travel time, average in queue time, average network travel time, and weighted average intersection person travel time. In this work, the authors have to fulfill some important requirements to develop a real-time adaptive traffic signal controller. Indeed, the controller has to consider both regular vehicles and public transit traffic, and leverage sensors' data on vehicle speed, position, and occupancy, moreover, the decision-making process should be fast.

Aziz et al. (2018) present an RL-based approach to control traffic signals in connected vehicle environments for reducing travel delays and GHG emissions. The proposed method, the R-Markov Average Reward Technique (RMART), leverages congestion information sharing among neighbor signal controllers and a multi-reward structure that can dynamically adapt the reward function according to the level of congestion at intersections. The considered environment presents discrete state (*) and action spaces (*) and a stochastic (*) transition model. The authors test RMART on a synthetic dataset and to evaluate its performance they use as metrics the average delay, stopped delay, number of stops, and network-wide delay, while to assess the performance from a sustainability point of view they leverage GHG emissions, i.e., CO, CO_2, NOX, VOC, PM10. Finally, this work deals with the traffic signal control problem to reduce travel delays and GHG emissions

by addressing the following issues: the sharing of congestion information among neighbor signal controllers and the dynamic adaptation of the reward function on the base of congestion level.

Reducing the number of drivers who commute in search of car parking in urban centers has a positive impact on environmental sustainability. In this context, Khalid et al. (2023) propose a Long-range Autonomous Valet Parking framework that optimizes the path planning of AVs to minimize distance while serving all users by picking them up and dropping them off at their required spots. The authors propose two learning-based solutions: Double-Layer Ant Colony Optimization (DL-ACO) and DQN-based algorithms. DL-ACO can be applied in new or unfamiliar environments, while DQN can be used in familiar environments to make efficient and fast decisions since it is pre-trainable. The DL-ACO approach determines the most efficient path between pair of spots and subsequently establishes the optimal order in which users can be served. To deal with dynamic environments, it is proposed a DQN-based algorithm in which the agent learns to solve the task by interacting with the environment and using memory experience replay and the target network. The proposed techniques aim to improve the carpool and parking experience while reducing the congestion rate. In this work, the environment considered is characterized by discrete states (*) and actions (*), and a deterministic (*) transition model. The proposed approach is tested on a synthetic dataset and execution time, reward, path planned, and distance are used as performance measures. In this work, the authors deal with path planning problems in dynamic environments while ensuring the quality of experience for each user, optimizing the order of user pick-up and drop-off, and finally minimizing the overall distance.

Buildings

Buildings are interesting from a DR and Demand Side Management point of view. In this context, Kathirgamanathan et al. (2021) leverage a DRL algorithm, namely Soft Actor-Critic (SAC), intending to automatize energy management and harness energy flexibility by controlling the cooling set point in a commercial building environment. In developing the proposed approach, the authors regard an environment with continuous states, and they evaluate the performance on a dataset that combines real-world and synthetic data using as evaluation metrics the energy purchased, energy cost, discomfort, total reward, temperature evolution, and power demand. Kathirgamanathan et al. (2021) tackle the application of DRL methods to automatize DR without the need for a specific building model and their robustness to different operating environments and scalability. Moreover, the authors point out that the lack of well-established environments makes it challenging to compare RL algorithms over different buildings.

de Gracia et al. (2015) instead consider Thermal Energy Storage, and in particular latent heat, techniques to maximize energy savings leveraging a Ventilated Double Skin Facade (VDSF) with Phase Change Material (PCM) used as a cold energy storage system. By using an RL approach, i.e.,

SARSA(λ), the authors control the VDSF to optimally schedule the solidification of PCM through mechanical ventilation during nighttime and the stored cold release into the indoor environment at peak demand time, considering weather and indoor conditions. The environment considered in this work presents discrete states (*), discrete actions (*), and a deterministic transition model. Moreover, the proposed approach is evaluated on a synthetic dataset considering electrical energy savings. This work aims to maximize energy savings by considering both the benefit of VDSF and the energy used in the solidification process. Therefore, it is crucial to determine the best time for the charging process to solidify the PCM and store coldness.

Manufacturing

Manufacturing industries are among the largest energy consumers, so it is crucial to develop approaches that make them more energy efficient. In this regard, Wang and Wang (2022) tackle the Energy-Aware Distributed Hybrid Flow-shop Scheduling Problem (EADHFSP). The goal of this work consists of simultaneously minimizing two conflicting objectives, makespan, and Total Energy Consumption. To this aim, the authors formulate a mixed-integer linear programming model of the EADHFSP and combine a Cooperative Memetic Algorithm with an RL-based agent to solve the problem. The authors combine two heuristics to initialize the population with various solutions and finally propose an improvement scheme in which solutions are refined by using the appropriate operator determined by a policy agent, while the solution selection is performed through the use of a decomposition strategy for balancing convergence and diversity. In this work, it is considered an environment characterized by discrete state (*) and action (*) spaces, a deterministic (*) transition model, and the performance of the presented approach is tested on a synthetic dataset considering the Overall Nondominated Vector Generation, C Metric, Hyper volume, and $D1_R$ as evaluation metrics. This work addresses the EADHFSP with the minimization of makespan and total energy consumption, a challenging problem due to the simultaneous optimization of two conflicting objectives.

Leng et al. (2021) focus on Printed Circuit Board (PCB) manufacturing and propose a Loosely-Coupled Deep Reinforcement Learning (LCDRL) model for energy-efficient order acceptance decisions. The authors leverage DL, specifically a Convolutional Neural Network (Le Cun, 1989), to obtain an accurate prediction of the production cost, makespan, and carbon consumption of each order by considering historical order labeled data. Then, the proposed approach combines the forecasted data with order features to decide whether to accept the order and determine the optimal acceptance sequence by using a reinforcement learning approach based on Q-Learning. The authors regard an environment with discrete actions (*) and a stochastic transition model, and they test the proposed method on a synthetic dataset (real-world-based). As performance measures, the metrics MSE, MSLE, RMSE, and R2 are used to evaluate the prediction accuracy of LCDRL, while the performance of the approach is assessed in terms of unit profit, total profit,

and acceptance rate. This work tackles the problem of order acceptance in PCB manufacturing to achieve energy efficiency, reduce carbon emissions, and improve material usage. Two critical aspects of PCB manufacturing are demand uncertainty and order customization which can lead to different profits, energy consumption, and carbon emissions. These two factors have to be considered in production planning under production constraints.

Mobile & Wireless communication

Sustainable energy infrastructures need high-quality communication systems to connect user facilities and power plants for providing information interaction. In this context, Liu et al. (2021) propose the use of a 6G network and Intelligent Reflective Surface (IRS) technology to create a wireless networking platform and suggests a DRL method to optimize the phase shift of IRS and therefore improve the communication quality. Combining the 6G Network with the IRS technology, the authors provide high-quality coverage while gaining energy-saving benefits. In particular, this work proposes the application of the Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2015) algorithm to configure the IRS phase shift for enhancing system coverage. The authors consider an environment characterized by continuous state and action spaces. The performance of the proposed approach is assessed on a synthetic dataset using two reflection units as metrics: the achievable rate to measure the service quality and the transmission power. Developing sustainable energy infrastructure is challenging from several points of view. Liu et al. (2021) tackle the need for an effective global covering communication system using the IRS technology, whose phase shift configuration is challenging itself.

In the context of two-tier urban Heterogeneous Networks (HetNets), Miozzo et al. (2015) model the Small Cell (SC) network as a decentralized multi-agent system. The authors' goal consists of improving system performance and self-sustainability of the SCs in terms of energy consumption. To this aim, they leverage the distributed Q-Learning algorithm so that every agent learns an appropriate Radio Resource Management policy. Miozzo et al. (2015) is extended in Miozzo et al. (2017). Here, it is proposed to train offline the algorithm to compute Q-values with which initialize the Q-tables of the SCs that will be used in the online method. In both approaches, the environment presents discrete states (*) and actions (*) and the dataset used is synthetic. In both works, the authors evaluate the proposed approaches in terms of network performance by using the throughput gain and traffic drop rate and their energy performance in terms of energy efficiency and energy efficiency improvement. Moreover, Miozzo et al. (2015) analyze the behavior of the HetNet considering traffic demand, harvested energy, battery level, policy, and normalized load at the macro Base Station (BS). Also, the authors consider as performance metrics the total amount of energy the system spent, the average load, the average cell load for the macro BS battery outage, and Jain's fairness index to assess the Quality of Service (QoS) improvement. Finally, Miozzo et al. (2017) assess the computed policy by leveraging the

switch-off rate as a performance measure and use the battery level to analyze the convergence of the online algorithm and evaluate the excess energy over storage capacity. Both works address the problem of introducing energy harvesting into the computation of sleeping strategies to achieve energy efficiency. This is challenging due to the irregular and intermittent nature of renewable energies.

Giri and Majumder (2022), instead, leverage a Deep Q-Learning algorithm for optimizing resource allocation in energy-harvested cognitive radio networks, where primary users networks share channel resources with secondary users and nodes can harvest energy from the environment, such as solar or wind. The proposed approach addresses the dynamic allocation of resources to achieve optimal network and throughput capacity, considering QoS, energy constraints, and interference limitations. Moreover, the authors utilize both linear and non-linear energy-harvested models, proposing a novel reward function that incorporates the non-linear one/model. The proposed approach works in environments characterized by continuous states (*), discrete actions (*), and a stochastic transition model (*) and it has been tested on a synthetic dataset by using reward, capacity, network lifetime, and average delay as performance measures. Giri and Majumder (2022) address the limitations of Q-learning-based allocation methods, then allow dealing with high-dimensional problems, improving convergence performance, and efficiently harnessing the collected energy to meet the network's QoS requirements.

Internet traffic has increased in recent years, and in the development of next-generation networks, it is important to address the QoS issue sustainably. In this context, Al-Jawad et al. (2021) propose an RL-based algorithm to solve routing problems in a Software Defined Network (SDN) environment, named Reinforcement lEarning-based Dynamic rOuting (REDO). Indeed, the proposed approach leverages Q-Learning to handle traffic flows by determining the most appropriate routing strategy among a set of conventional routing algorithms with the aim to maximize flows meeting the Service Level Agreement as to throughput, packet loss, and rejection rate. In developing REDO, the authors consider an environment with discrete state (*) and action spaces (*). The performance of the proposed approach is evaluated on a synthetic dataset in terms of throughput, packet loss, rejected flows, PSNR, and Mean Opinion Score (MOS). In the development of next-generation networks like SDN, Al-Jawad et al. (2021) address the problem of providing QoS sustainably through the solution of a traffic flow routing problem.

Electric energy

One way to increase environmental sustainability is to improve the energy efficiency of smart hubs. To this aim, Sheikhi et al. (2016) present the new Smart Energy Hub framework for modeling distinct energy infrastructures under a single framework. The authors' goal consists of optimizing the electrical and natural gas consumption of a residential customer through the use

of Q-Learning. Moreover, to improve and support information management among users and utility service providers, the proposed framework leverages Cloud Computing based systems. In this case the only technical information available concern the use of a stochastic transition model and a synthetic dataset. To evaluate the performance of the proposed approach, the metrics used are the storage charge level, the operational cost, and the primary energy involved. As regards dynamic load management in smart hubs, the authors tackle two issues. The first one is related to energy system parameters which are often assumed to be constant but can vary with time or be stochastic in practice. The second one is, instead, related to the conventional smart grid architecture, which has several reported issues, including exposure to cyber-attacks, single failure problems, limited memory and storage capacity in the energy management system, and difficulties in implementing real-time early warning systems due to limited energy and bandwidth resources.

In the context of smart energy networks, Harrold et al. (2022) consider a microgrid environment and leverage DRL to control a battery for energy arbitrage and increased use of renewable energies, namely solar and wind energy. Specifically, the authors apply the Rainbow Deep Q-Network (Hessel et al., 2018) algorithm and add predicted values for demand, Renewable Energy Source (RES), and energy price to agents' information by leveraging an Artificial Neural Network. In this work, the environment considered is characterized by continuous states (*) and discrete actions. The authors test the proposed approach on a dataset that considers both real-world and synthetic data and assess the prediction accuracy using the MAPE. Also, they evaluate the performance of the proposed approach through energy cost savings, relative savings, episodic rewards, and value distribution. This work tackles the problem of controlling an Energy Storage System in a microgrid with its demand, RES, and dynamic energy pricing to perform energy arbitrage and improve the use of RES leading to reduced energy cost. Finally, the authors point out the limited availability of data that requires an efficient algorithm training procedure.

A key aspect of sustainability and cost-effectiveness in grid operation is optimal energy dispatch. Jendoubi and Bouffard (2022) address a multi-dimensional power dispatch problem within a power system by leveraging MARL, specifically the MADDPG algorithm. The proposed control framework performs CTDE to improve the coordination among dispatchable units without communication needed, and thus it mitigates data privacy and communication issues. In developing the approach, an environment with continuous actions, and a stochastic transition model (*) is considered. The dataset used to evaluate the performance is synthetic and the proposed method is evaluated in terms of the annual total cost, variation of daily operation cost, photovoltaics (PV) production, aggregated demand, amount of power to be charged/discharged, amount of power provided by a diesel generator, amount of electricity delivered by the electricity provider, the difference in the amount of electricity delivered by the electricity provider between two

consecutive time steps and Peak-to-Average Ratio (PAR). The authors address the energy dispatch aspects related to the development of distributed energy resources control strategies in grid operation to simultaneously reduce costs and delays and allow local coordination among energy resources.

Energy

In recent years, international trade and container handling at port terminals have increased greatly. Improving sustainability in port operations closely relates to the energy consumption at Automated Container Terminals, where Automatic Stacking Cranes (ASCs) are used to load, unload, and pile containers. In this context, Gao et al. (2023) propose a digital twin-based approach for container yard management. Specifically, this work focuses on determining the optimal allocation of container tasks and scheduling of ASCs to reduce the energy consumption of ASCs while maintaining efficient loading and unloading operations. The proposed approach leverages a virtual container yard to simulate the operating plan and mixed integer programming model to optimize the scheduling problem taking into account the energy consumption. Finally, the authors use the Q-Learning algorithm to determine the optimal scheduling plan and minimize energy consumption. The environment considered in this work presents discrete actions (*) and a stochastic (*) transition model. The performance of the proposed approach is evaluated on a real-world dataset using working and non-working energy consumption and the ratio between them as metrics. To improve the sustainability of port operations, in this work the problem of optimizing container yard operations to minimize energy consumption is addressed. Indeed, several factors can introduce randomness and uncertainty into these operations, and incorrect distribution of tasks can lead to suboptimal utilization of ASCs.

Wireless Sensor Network

Regarding embedded systems powered by a renewable energy source like an Energy Harvesting Wireless Sensor Node (EHWSN), Hsu et al. (2014) present a method called Reinforcement Learning-based throughput on-demand provisioning dynamic power management (RLTDPM). By leveraging the Q-learning algorithm, the proposed approach allows the EHWSN to adapt the operational duty cycle to satisfy both the energy neutrality condition and the throughput on-demand (ToD) requirement, ensuring perpetual operation. In developing RLTDPM, the authors regard an environment characterized by discrete state (*) and actions (*) and evaluate the performance on a synthetic dataset by considering the residual battery energy (RBE), exercised duty cycle (EDC), offset to the required ToD (OTRT), and ToD achievability. In this work, the authors address the problem of simultaneously achieving two mutually conflicting goals i.e., satisfying ToD and reducing power consumption.

Energy-Harvesting Wireless Sensor Networks (WSNs) are widely used in energy-constrained operation problems. In particular, Chen et al. (2016) fo-

cus on Solar-Powered Wireless Sensor Networks (SPWSNs) and present an RL-based Sleep Scheduling for Coverage algorithm to improve the sustainability of SPWSN's operations. The proposed approach leverages a precedence operator in the group formation algorithm to prioritize sensors in sparsely covered areas ensuring the desired coverage distribution. Then, the authors propose a multi-sensor cooperation Q-learning group model to properly choose nodes' working modes by leveraging the developed learning and action selection strategies. The whole group learns the sleep schedule by changing the role of the active node. The environment considered in this work presents discrete state (*) and action (*) spaces and a stochastic transition model. The proposed approach is tested on a dataset that combines real-world and synthetic data, and its performance is evaluated in terms of energy balancing between group members by using the potential energy of nodes as metrics, network lifetime, area coverage ratio, number of residual alive nodes versus the network lifetime, and the recharging cycle. In this work, the authors tackle a sleep scheduling problem to simultaneously achieve the desired area coverage and energy balance between group nodes to extend the network lifetime.

On the other hand, Feng et al. (2023) propose an RL-based approach to maximize data throughput in self-sustainable WSNs. The authors consider a Mobile Sensor (MS) that collects and transmits data to a fixed sink while moving within the network and harvesting energy from the environment. By leveraging DDPG, the MS can determine the optimal trajectory to optimize the energy harvesting performance and data transmission dealing with unknown energy supply dynamics. The environment considered in this work is characterized by continuous states and actions, and a stochastic transition model (*). Moreover, the performance of the proposed approach is assessed on a synthetic dataset considering as evaluation metrics the distribution of the ratio between the expected per-slot harvested energy and the MS-to-sink distance within the network, moving trajectories of the MS, reward, actor-loss, battery level, accuracy, moving steps, convergence, and training time. The authors tackle two main challenges concerning the MS's trajectory optimization to maximize data throughput. The first relates to the lack of energy-related information such as the energy sources' placement, future energy harvesting potential, and statistical parameters like the average energy harvesting rate, which makes the problem challenging. The second consists of the tradeoff between energy harvesting and data transmission. Indeed, moving closer to energy sources allows the MS to increase the energy harvesting amount. However, this may lead to decreasing data transmission power due to a possible increase in the distance between the MS and the sink.

Autonomous vehicles

In the last decade, Unmanned Aerial Vehicles (UAVs), i.e., drones, have been used in various scenarios such as rapid disaster response, Search-And-Rescue, environmental monitoring, etc. where the human is unable to timely

and efficiently operate for example due to the presence of physical obstacles. Bouhamed et al. (2020) consider the application of UAVs as mobile data collection units in delay-tolerant WSNs. The authors propose a mechanism that exploits two RL techniques, namely DDPG and Q-learning algorithms. The proposed approach uses DDPG to determine the best trajectory for a UAV to reach the target destination while avoiding obstacles in the environment. Q-Learning, on the other hand, is used to schedule the best order of visiting nodes to minimize the time needed to collect data within a predefined time limit. In this work, the environment presents continuous state (*) and action spaces for the DDPG-based part of the approach while discrete states (*) and actions (*) for the Q-Learning-based one. The proposed mechanism is tested on a synthetic dataset and to evaluate its obstacle avoidance and scheduling performance, the authors analyze the path followed by the UAV, the reward collected, the UAV's battery level, and the completion time of the tour against the ground unit transmission power. This work addresses issues related to the limited battery capacity of UAVs and challenges related to navigating in obstacle-prone environments to enable communication between the UAV and low transmission power sensors.

Sacco et al. (2021) propose a MARL approach based on the actor-critic framework to tackle task offloading problems from UAV swarms in edge computing environments to simultaneously reduce task completion time and improve energy efficiency. The proposed approach determines a distributed decision strategy through the collaboration among the system's mobile nodes that share information about the overall system state. This information is then used by the agents to decide whether to compute a task locally or offload it to the edge cloud and in this case, the proposed technique chooses the best transmission technology among Wi-Fi access points and mobile network. In developing the proposed approach, the environment considered presents continuous states (*) and discrete actions (*), and the dataset used for testing combines real-world and synthetic data. The performance of the presented techniques is assessed in terms of task completion time and utility against a varying number of agents, and average node-antenna distance. Then, the authors evaluate the energy consumption necessary to complete the task by varying the average node-antenna distance and computing workload and assess the task completion time against the average computing workload. In addition, the cumulative distribution function (CDF) and utility evolution through episodes are considered to analyze the variability of performance among nodes and convergence performance, respectively. Finally, in this work, the authors tackle the problem of reducing the time necessary for task completion of UAV swarms by leveraging task offloading to the edge cloud while improving energy efficiency.

In the context of autonomous driving, Gu et al. (2023) tackle the application of RL methods focusing on energy-saving and environmentally friendly driving strategies within a cooperative adaptive cruise control platoon. More precisely, the goal of this work consists of training platoon member vehicles to react effectively when the leading vehicle faces a severe collision. The au-

thors leverage the Policy Gradient algorithm for training an RL agent to minimize the energy consumption in inter-vehicle communication for decision-making while avoiding collisions or minimizing the resulting damage. To this aim, two different loss functions are used, i.e., collision loss and energy loss. Moreover, utilizing a specific reward function can both ensure the vehicle's safety and consider the fuel consumption resulting from the action performed by the vehicle. This work considers an environment characterized by continuous states (*), discrete actions, and a stochastic transition model (*). The proposed approach has been tested on a synthetic dataset using energy loss, collision loss, reward, and lane changes as metrics. A key challenge of green autonomous driving addressed in this work is the development of effective strategies that can respond to environmental observations by automatically generating appropriate control signals.

6.4 Discussion

The analysis of the literature performed in this work shows that most of the works about RL for environmental sustainability concern the energy application domain, followed by urban traffic and transportation. The main RL technique used in the reviewed manuscripts is Q-learning. Concerning the 35 selected articles, we observe that energy-related issues involve most of the papers, and about half of them leverage DRL approaches, such as DQN and DDQN. In developing the proposed methods, the authors mainly consider domains with discrete state and action spaces, and stochastic transition models, using synthetic datasets to evaluate the performance.

Problems related to environmental sustainability were traditionally tackled with optimization techniques in which the concept of adaptability has to be introduced explicitly. In contrast, one of the strengths of RL is its natural way of dealing with adaptability to changing or different environments, a crucial feature in environmental sustainability problems since in this context the agent has to handle variations in operating conditions due to, for example, changes in resource availability or weather conditions. For instance, Chen et al. (2016) introduce a RL-based Sleep Scheduling for Coverage (RLSSC) approach to ensure sustainable time-slotted operations in solar-powered wireless sensor networks. This algorithm is compared to a high-energy-efficient hierarchical routing protocol, i.e., LEACH (Heinzelman et al., 2002), wherein the node chosen to be active in the current round is ineligible for selection in the subsequent round, and a random algorithm that randomly determines active nodes within a group. Among the various aspects considered, a crucial criterion for evaluating algorithm effectiveness lies in maintaining equilibrium in energy levels, as significant disparities in current residual energy arise when a node receives an energy supplement. RLSSC initially exhibits fluctuations but eventually converges through iterative learning, exhibiting slight oscillations up and down in response to varying solar strength throughout the day. Moreover, the proposed approach demonstrates real-time energy balancing among sensor nodes. In contrast,

non-RL-based methods lack the capacity to adapt to the dynamic environment. Another aspect to consider is network lifetime, where RLSSC excels in adapting to uncertainties associated with harvesting time and the amount of acquired energy. This adaptability enables RLSSC to dynamically adjust its scheme in real-time, effectively extending the overall network lifetime. This is only one of several examples showing that RL can provide a strong advantage in solving problems related to environmental sustainability because of its natural capability to deal with uncertainty and adaptation in sequential decision-making.

However, we identify several open problems in the application of RL techniques to environmental sustainability. These concern scalability, data efficiency, and the necessity to deal with large data volumes, often posing cost challenges. In future developments, it is crucial to improve pre-training methods that allow the generation of initial policies by simulation and leverage knowledge acquired by solving a related task. RL methods are also sensitive to reward function therefore reward engineering is important to avoid a negative impact on performance. Moreover, in dealing with environmental sustainability problems in specific contexts like IoT, it is of particular importance to consider the presence of computational limitations and then optimize the computational complexity of the method. Finally, we note that most of the approaches involve single-agent systems. Extending the proposed approaches to the multi-agent context would allow the cooperative computation of optimal policies accounting for common performance objectives to improve shared resources management and environmental sustainability.

Chapter 7

Conclusions and Future Work

In this concluding chapter, we draw the conclusions of the work conducted in the thesis and summarize the key findings derived from our research (Section 7.1). Additionally, we outline some possible future work to improve the presented approaches and advance research in the real of MCTS-based planning (Section 7.2).

7.1 Conclusions

This thesis proposes novel methods for learning in MCTS-based planning. One of the main goals of our approaches is to use the information the agent acquires while interacting with the environment to improve the agent's policy and related performance. We chose MCTS as a planning strategy because of its capability to compute the Q-function online and locally (i.e., only for the states actually visited by the agent). This feature makes it scalable to extensive domains commonly encountered in real-world applications. In addition, MCTS sampling differs from various RL methods as it does not require a complete transition matrix but it relies on a black-box simulator, a function that generates the next state based on the current state-action pair.

We developed methods for both POMDPs (Chapter 4) and MDPs (Chapter 5). In designing the different learning methods, we addressed several problems such as deciding when to stop the learning process and start using the acquired information or determining how to integrate the collected information during the interaction with the environment to achieve improved planning performance. In particular, we propose contributions to the literature in three areas: *1)* learning state-variable relationships in POMCP, *2)* learning the dynamics model of the environment in MCTS, *3)* surveying the application of RL techniques to environmental sustainability. More in detail, this thesis advances the state-of-the-art with the following contributions:

1. Learning state-variable relationships in POMCP

    - We developed three approaches for learning probabilistic state-variable relationships, represented by MRFs, in POMCP. The first MRF learning approach uses observations in the real world, while

the other two consider information in the belief of the agent, and respectively, the maximum likelihood state and a weighted sum of the states. Furthermore, we proposed two criteria to decide when the learning stage can be stopped and the MRF can be used by POMCP to obtain performance improvement. The first criterion is based on confidence intervals of MRF potentials and the second one is based on the convergence of MRF potentials. Our empirical analysis on two robotics inspired domains (i.e., rocksample and velocity regulation) shows that the learning approach based on the maximum likelihood state in the belief generates the most accurate MRFs. Its usage in tandem with the stopping criterion based on confidence intervals of MRF potentials yields a statistically significant performance improvement over the standard POMCP without any increase in time complexity.

- We proposed a framework to integrate POMCP within ROS for enabling the employment of the MRF learning algorithm on real robotic platforms. The proposed framework supports both the phase in which state-variable relationships are learned and the phase in which such knowledge is used. The ROS-based architecture is made of three ROS nodes: environment, agent, and planning. The environment node discretizes the real world by exploiting a task-specific representation. The agent node, instead, holds information about odometry and interfaces the ROS-based robotic platform with the environment and the planner. Finally, the planner node runs the learning algorithm. We tested the proposed ROS-based architecture on a Gazebo simulator of rocksample. The architecture enables the generation of informative MRFs that produces statistically significant performance improvements.
- We developed an algorithm, called *Adapt*, for adapting variable values constraints to episodes having unlikely state-variable configurations, as new observations are acquired from the environment. This algorithm runs when the knowledge provided by the learned MRF does not reflect the true state-variable values. In such cases the MRF is misleading, since it forces the belief probabilities towards configurations of state-variables that are discordant from that of the true state, decreasing the probability of the true state. Thus, the proposed algorithm adapts (i.e. changes) the MRF potentials when the agent acquires knowledge about the true state-variable values and detects a mismatch between the information in the learned MRF and the specific state-variable relationships of the episode, to fix the mismatch. Our empirical analysis shows that the MRF adaptation method improves the performance obtained using the MRF without adaptation. We tested the algorithm on two domains, i.e., rocksample and velocity regulation. Results show an average improvement in discounted reward of 6.54% on rocksample and of 3.51% on velocity regulation.

2. Learning the dynamics model of the environment

    - We proposed a MCTS-based planning approach that improves an expert-defined approximate model of the dynamics according to the information gathered online from the environment. Such a method represents the model as an ANN and it has been used in MCTS-based planning. To integrate new information into the current model, we propose three different approaches. The first version periodically updates the weights of the copy neural network using environment data as a training set. The second version merges data (i.e., state-action-next-state triplets) generated by the original expert's model with data from the environment and periodically re-trains the copy neural network using this dataset as a training set. The third version keeps the copy neural network of the expert's model unchanged and trains a separate neural network with data from the environment, then during Monte Carlo simulations it selects the best model to perform each step according to how close is the current state-action pair to the training set by which the model has been trained. To evaluate our approach we consider a real-world application related to air quality and thermal comfort control in smart buildings. We evaluated the performance of the three versions of our method and compared them to three baseline approaches, namely, *i)* MCTS using the expert's model with no adaptations to the real environment, *ii)* Proximal Policy Optimization (PPO), and *iii)* Stochastic Lower Bounds Optimization (SLBO). The second version of our approach, called MCP_M, achieved the best performance among the proposed versions. More importantly, this version outperformed the three state-of-the-art competitors, in terms of discounted return, in our reference domain.

3. Surveying the application of RL techniques to environmental sustainability

    - We surveyed the use of RL in environmental sustainability applications. Our goal is to provide practitioners with a comprehensive insight about state-of-the-art methods for environmental sustainability in various application domains. We first analyzed 181 papers collected from Scopus and Web of Science providing a quantitative analysis of the literature related to the application of RL in environmental sustainability over the last two decades. We then focused on a subset of 35 main papers for which we analyzed the application domains of the proposed RL techniques, provided technical insights into the problem formalization, discussed the performance metrics used for evaluation, and considered the new challenges in this field.

7.2 Future Work

The work developed in this PhD project opens to several future research lines within MCTS-based planning.

- Regarding the state-variable relationship learning approach, an interesting direction from a methodological point of view concerns the integration of the learning process in the context of information gain problems on POMDPs. The goal is to tune the exploration-exploitation trade-off in an efficient way considering the learning of the MRF to further increase the performance improvement obtained by introducing prior knowledge in POMCP.

- New methods to integrate the MRF specifically in the reinvigoration process of POMCP can be developed to deal with very large domains in which standard rejection is known to be suboptimal.

- A new criterion for stopping the MRF learning process can be developed as an alternative to the convergence-based stop learning criterion. This approach, despite its simplicity, uses thresholds whose settings can be difficult and time-consuming to tune.

- From the application perspective, it would be interesting to extend the proposed ROS architecture to support other kinds of platforms, such as robotic manipulators, for evaluating our method on different problems that can be formalized as POMDPs.

- Another interesting direction in the context of learning state-variable relationships is that of extending the proposed approach to multi-agent reinforcement learning (MARL). Especially in large domains, the Centralized Training with Decentralized Execution (CTDE) paradigm could be a noteworthy strategy to optimize the learning of state-variable relationships. In the learning phase, each agent, operating within its designated segment of the domain, can share local observations acquired. The collected knowledge can then be successively combined into a unified MRF that can be used by all agents in the operational phase.

- In the area of learning the dynamics model in MCTS, future work primarily involves the application of the proposed methodology to diverse domains, to further evaluate it and possibly extend its capabilities. The domain on which we tested the approach in this paper is characterized by a continuous state space and a discrete action space. Our focus is now on exploring its applicability to domains characterized by discrete state/action spaces, as well as in those with continuous state/action spaces. Furthermore, we would like to specialize the proposed method to stochastic environments. This investigation aims to evaluate the generality and effectiveness of the method in different environmental structures and complexities.

- Another interesting direction is to test different expert models in the context of model learning/adaptation for MCTS. We would like to analyze how the properties of these models affect policy performance.

- Continual Learning (CL) (Lesort et al., 2020) shares some similarities with our work since in both cases training data are not available at once and need to be integrated by keeping the most representative data, as rehearsal approaches do (Lesort et al., 2019; Rebuffi et al., 2017). Our idea is to leverage CL approaches to facilitate data fusion in our ANN models of the dynamics preserving the knowledge acquired from data that might become inaccessible. Dealing with the problem of catastrophic forgetting (French, 1999), especially in non-stationary application domains, could improve the performance of our method for learning dynamics models in MCTS. Catastrophic forgetting is a phenomenon in ML where a model, especially neural networks, tends to lose knowledge of previously learned tasks as it adapts to new information. This occurs because the model's weight updates to accommodate new data can overwrite the established knowledge pertinent to earlier tasks. This issue poses a significant challenge in scenarios where systems are expected to accumulate knowledge over time without compromising performance on previously encountered tasks. Recently, several approaches have been proposed to address this problem in both single-agent (Hurtado et al., 2023; Cossu et al., 2020) and multi-agent scenarios (Carta et al., 2022). In future work, we could compare the performance of some of these methods with that of our model-learning technique for MCTS and also integrate some CL approaches into the proposed model-learning method.

- The proposed approach for learning the environment's dynamics relies on ANNs to model these dynamics, with MCTS serving as an internal planner that utilizes the ANN for simulations. The current network structure is an initial design choice, and further investigation of how optimizing the number of hidden layers and nodes within those layers could affect the performance of our approach would be an interesting direction for further study.

- Another intriguing avenue for future exploration involves considering alternative frameworks for learning the dynamics model. While ANNs offer effective modeling capabilities, they lack explicit structure about the specific dynamics under investigation. Introducing such a structure in the model (if available from prior knowledge) in an explicit way can allow to improve sample efficiency. Therefore, a promising direction for future research is the substitution of ANNs with alternative models, such as decision trees, regression models, etc., to enhance the interpretability and insight into the problem's structural aspects.

- Also, we plan to investigate the use of more sophisticated techniques for determining the best time to perform model retraining. This could

include exploring methods such as concept drift (Lu et al., 2019b) or conformal prediction (Shafer and Vovk, 2008). Concept drift refers to unpredictable alterations in the underlying data distribution over time. This evolution can occur due to various factors, such as alterations in environmental conditions, or changes in the system being modeled. Managing concept drift is essential for maintaining the effectiveness of ML models in dynamic, real-world scenarios. Effectively managing concept drift ensures that models remain accurate in the face of changing data distributions, allowing them to provide reliable predictions and insights over time. Conformal prediction, on the other hand, relies on historical data to establish precise confidence levels for new predictions. This methodology relies on the construction of prediction regions that are expected to contain the true outcome with a predefined confidence level. In this way, conformal prediction provides a principled approach to quantify the uncertainty associated with individual predictions, making it particularly valuable in scenarios where robust uncertainty estimation is essential.

- Future work could extend the study of the application of RL approaches to environmental sustainability problems to compare these results with those of a quantitative analysis of the state of the art of more general RL approaches over the past two decades. This comparison could focus on methodologies, application domains, and keywords. For instance, we could explore whether specific techniques are more frequently referenced when considering environmental sustainability aspects compared to their references in more general RL applications.

- Finally, a further extension of the study of the application of RL to environmental sustainability problems to include dynamic programming and MDP planning-based approaches and conduct a wider comparative analysis.

Bibliography

P. Abbeel, D. Koller, and A. Y. Ng. Learning factor graphs in polynomial time and sample complexity. *Journal of Machine Learning Research*, 7:1743–1788, 2006.

A. Aczel and J. Sounderpandian. *Complete business statistics*. McGraw-Hill, 2008.

Y. Adulyasak, P. Varakantham, A. Ahmed, and P. Jaillet. Solving uncertain MDPs with objectives that are separable over instantiations of model uncertainty. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*, volume 29, pages 3454–3460. AAAI Press, 2015. doi: 10.1609/aaai.v29i1.9695.

L. A. Ajao and S. T. Apeh. Secure edge computing vulnerabilities in smart cities sustainability using Petri net and genetic algorithm-based reinforcement learning. *Intelligent Systems with Applications*, 2023. doi: 10.1016/j.iswa.2023.200216.

A. Al-Jawad, I. S. Comsa, P. Shah, O. Gemikonakli, and R. Trestian. REDO: A reinforcement learning-based dynamic routing algorithm selection method for SDN. In *IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pages 54–59. IEEE Press, 2021. doi: 10.1109/NFV-SDN53031.2021.9665140.

K. Alanne and S. Sierla. An overview of machine learning applications for smart buildings. *Sustainable Cities and Society*, 76, 2022. doi: 10.1016/j.scs.2021.103445.

S. M. Alizadeh Shabestray and B. Abdulhai. Multimodal intelligent deep (MiND) traffic signal controller. In *IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 4532–4539. IEEE Press, 2019. doi: 10.1109/ITSC.2019.8917493.

C. Amato and F. A. Oliehoek. Scalable planning and learning for multiagent POMDPs. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*, volume 29, pages 1995–2002. AAAI Press, 2015.

A. Angelidakis and G. Chalkiadakis. Factored MDPs for optimal prosumer decision-making. In *Proceedings of the 14th International Conference on*

Autonomous Agents and MultiAgent Systems (AAMAS), pages 503–511. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2015a.

A. Angelidakis and G. Chalkiadakis. Factored MDPs for optimal prosumer decision-making in continuous state spaces. In *Multi-Agent Systems and Agreement Technologies - 13th European Conference (EUMAS)*, volume 9571, pages 91–107. Springer, 2015b. doi: 10.1007/978-3-319-33509-4_8.

A. Angelidakis and G. Chalkiadakis. Optimal prosumer decision-making using factored MDPs. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4110–4114. ijcai.org, 2016.

M. Araya, O. Buffet, V. Thomas, and F. Charpillet. A POMDP extension with belief-dependent rewards. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 1, pages 64–72. Curran Associates, Inc., 2010.

A. Atrash and J. Pineau. A Bayesian method for learning POMDP observation parameters for robot interaction management systems. In *The POMDP Practitioners Workshop*, 2010.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002. doi: 10.1023/A:1013689704352.

F. Auffenberg, S. Snow, S. Stein, and A. Rogers. A comfort-based approach to smart heating and air conditioning. *ACM Transactions on Intelligent Systems and Technology*, 9, 2017. doi: 10.1145/3057730.

H. M. A. Aziz, F. Zhu, and S. V. Ukkusuri. Learning-based traffic signal control algorithms with neighborhood information sharing: An application for sustainable mobility. *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*, 22:40–52, 2018. doi: 10.1080/15472450.2017.1387546.

D. Azzalini, A. Castellini, M. Luperto, A. Farinelli, and F. Amigoni. HMMs for anomaly detection in autonomous robots. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 105–113. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2020. doi: 10.5555/3398761.3398779.

J. A. Bagnell, A. Y. Ng, and J. G. Schneider. Solving uncertain Markov decision processes. *Technical report, Carnegie Mellon University*, 2001. doi: 10.1184/R1/6560927.v1.

A. L. C. Bazzan, A. Peleteiro-Ramallo, and J. C. Burguillo-Rial. Learning to cooperate in the iterated prisoner's dilemma by means of social attachments. *Journal of the Brazilian Computer Society*, 17:163–174, 2011. doi: 10.1007/s13173-011-0038-2.

R. Bellman. Dynamic programming. *Science*, 153:34–37, 1966. doi: 10.1126/science.153.3731.34.

D. P. Bertsekas and D. A. Castanon. Adaptive aggregation methods for infinite horizon dynamic programming. *IEEE Transactions on Automatic Control*, 34(6):589–598, 1989. doi: 10.1109/9.24227.

J. Besag. Efficiency of pseudolikelihood estimation for simple Gaussian fields. *Biometrika*, 64:616–618, 1977. doi: 10.1093/biomet/64.3.616.

F. Bianchi, A. Castellini, P. Tarocco, and A. Farinelli. Load forecasting in district heating networks: Model comparison on a real-world case study. In *Proceedings of the 5th International Conference on Machine Learning, Optimization, and Data Science (LOD)*, pages 553–565. Springer-Verlag, 2019. doi: 10.1007/978-3-030-37599-7_46.

F. Bianchi, D. Corsi, L. Marzari, D. Meli, F. Trotti, M. Zuccotto, A. Castellini, and A. Farinelli. Safe and efficient reinforcement learning for environmental monitoring. In *Proceedings of the Italia Intelligenza Artificiale - Thematic Workshops co-located with the 3rd CINI National Lab AIIS Conference on Artificial Intelligence (Ital-IA)*, volume 3486, pages 610–615. CEUR-WS.org, 2023.

A. Bifet, R. Gavaldà, G. Holmes, and B. Pfahringer. *Machine Learning for Data Streams: With Practical Examples in MOA*. The MIT Press, 2018. doi: 10.7551/mitpress/10654.001.0001.

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, 2006.

F. Bistaffa, A. Farinelli, G. Chalkiadakis, and S. D. Ramchurn. A cooperative game-theoretic approach to the social ridesharing problem. *Artificial Intelligence*, 246:86–117, 2017. doi: 10.1016/J.ARTINT.2017.02.004.

F. Bistaffa, C. Blum, J. Cerquides, A. Farinelli, and J. A. Rodríguez-Aguilar. A computational approach to quantify the benefits of ridesharing for policy makers and travellers. *IEEE Transactions on Intelligent Transportation Systems*, 22:119–130, 2021. doi: 10.1109/TITS.2019.2954982.

O. Bouhamed, H. Ghazzai, H. Besbes, and Y. Massoud. A UAV-assisted data collection for wireless sensor networks: Autonomous navigation and scheduling. *IEEE Access*, 8:110446–110460, 2020. doi: 10.1109/ACCESS.2020.3002538.

R. I. Brafman, M. Bar-Sinai, and M. Ashkenazi. Performance level profiles: A formal language for describing the expected performance of functional modules. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1751–1756. IEEE Press, 2016. doi: 10.1109/IROS.2016.7759280.

J. Brown, A. Abate, and A. Rogers. QUILT: Quantify, infer and label the thermal efficiency of heating and cooling residential homes. In *Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys)*, pages 51–60. Association for Computing Machinery (ACM), 2021. doi: 10.1145/3486611.3486653.

C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4:1–43, 2012. doi: 10.1109/TCIAIG.2012.2186810.

L. Buesing, T. Weber, S. Racaniere, S. M. A. Eslami, D. Rezende, D. P. Reichert, F. Viola, F. Besse, K. Gregor, D. Hassabis, and D. Wierstra. Learning and querying fast generative models for reinforcement learning. *arXiv preprint arXiv:1802.03006*, 2018.

M. Capuzzo, A. Zanella, M. Zuccotto, F. Cunico, M. Cristani, A. Castellini, A. Farinelli, and L. Gamberini. IoT systems for healthy and safe life environments. In *Proceedings of the 7th IEEE Forum on Research and Technologies for Society and Industry Innovation (RTSI)*, pages 31–37. IEEE Press, 2022. doi: 10.1109/RTSI55261.2022.9905193.

A. Carta, A. Cossu, V. Lomonaco, D. Bacciu, and J. van de Weijer. Projected latent distillation for data-agnostic consolidation in distributed continual learning. *arXiv preprint arXiv:2303.15888*, 2022.

H. Caselles-Dupré, M. Garcia-Ortiz, and D. Filliat. S-TRIGGER: Continual state representation learning via self-triggered generative replay. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE Press, 2021. doi: 10.1109/IJCNN52387.2021.9533683.

A. Castellini, G. Beltrame, M. Bicego, J. Blum, M. Denitto, and A. Farinelli. Unsupervised activity recognition for autonomous water drones. In *Proceedings of the 33rd ACM/SIGAPP Symposium on Applied Computing (SAC)*, pages 840–842. Association for Computing Machinery (ACM), 2018a. doi: 10.1145/3167132.3167396.

A. Castellini, J. Blum, D. Bloisi, and A. Farinelli. Intelligent battery management for aquatic drones based on task difficulty driven POMDPs. In *Proceedings of the 5th Italian Workshop on Artificial Intelligence and Robotics - A workshop of the 17th International Conference of the Italian Association for Artificial Intelligence (AIRO@AI*IA)*, volume 2352, pages 24–28. CEUR-WS.org, 2018b.

A. Castellini, M. Bicego, D. Bloisi, J. Blum, F. Masillo, S. Peignier, and A. Farinelli. Subspace clustering for situation assessment in aquatic drones: A sensitivity analysis for state-model improvement. *Cybernetics and Systems*, 50:658–671, 2019a. doi: 10.1080/01969722.2019.1677333.

A. Castellini, G. Chalkiadakis, and A. Farinelli. Influence of state-variable constraints on partially observable Monte Carlo planning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 5540–5546. ijcai.org, 2019b. doi: 10.24963/ijcai.2019/769.

A. Castellini, E. Marchesini, and A. Farinelli. Online Monte Carlo planning for autonomous robots: Exploiting prior knowledge on task similarities. In *Proceedings of the 6th Italian Workshop on Artificial Intelligence and Robotics - A workshop of the 18th International Conference of the Italian Association for Artificial Intelligence (AIRO@AI*IA)*, volume 2594 of *CEUR Workshop Proceedings*, pages 25–32. CEUR-WS.org, 2019c.

A. Castellini, F. Masillo, M. Bicego, D. Bloisi, J. Blum, and A. Farinelli. Subspace clustering for situation assessment in aquatic drones. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing (SAC)*, pages 930–937. Association for Computing Machinery (ACM), 2019d. doi: 10.1145/3297280.3297372.

A. Castellini, F. Masillo, R. Sartea, and A. Farinelli. eXplainable Modeling (XM): Data analysis for intelligent agents. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 2342–2344. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2019e.

A. Castellini, M. Bicego, F. Masillo, M. Zuccotto, and A. Farinelli. Time series segmentation for state-model generation of autonomous aquatic drones: A systematic framework. *Engineering Applications of Artificial Intelligence*, 90, 2020a. doi: 10.1016/j.engappai.2020.103499.

A. Castellini, D. Bloisi, J. Blum, F. Masillo, and A. Farinelli. Multivariate sensor signals collected by aquatic drones involved in water monitoring: A complete dataset. *Data in Brief*, 30, 2020b. doi: 10.1016/j.dib.2020.105436.

A. Castellini, E. Marchesini, G. Mazzi, and A. Farinelli. Explaining the influence of prior knowledge on POMCP policies. In *Multi-Agent Systems and Agreement Technologies - 17th European Conference (EUMAS)*, volume 12520, pages 261–276. Springer, 2020c. doi: 10.1007/978-3-030-66412-1_17.

A. Castellini, F. Bianchi, and A. Farinelli. Predictive model generation for load forecasting in district heating networks. *IEEE Intelligent Systems*, 36: 86–95, 2021a. doi: 10.1109/MIS.2020.3005903.

A. Castellini, E. Marchesini, and A. Farinelli. Partially observable Monte Carlo planning with state variable constraints for mobile robot navigation. *Engineering Applications of Artificial Intelligence*, 104, 2021b. doi: 10.1016/j.engappai.2021.104382.

A. Castellini, F. Bianchi, and A. Farinelli. Generation and interpretation of parsimonious predictive models for load forecasting in smart heating networks. *Applied Intelligence*, 52:9621–9637, 2022a. doi: 10.1007/s10489-021-02949-4.

A. Castellini, C. Morasso, and A. Farinelli. HMM-based anomaly interpretation for intelligent robots in Industry 4.0. In *Proceedings of the 9th Italian Workshop on Artificial Intelligence and Robotics - A workshop of the 21st International Conference of the Italian Association for Artificial Intelligence (AIRO@AI*IA)*, volume 3417, pages 96–103. CEUR-WS.org, 2022b.

A. Castellini, F. Bianchi, E. Zorzi, T. D. Simão, A. Farinelli, and M. T. J. Spaan. Scalable safe policy improvement via Monte Carlo tree search. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202, pages 3732–3756. PMLR, 2023a.

A. Castellini, F. Masillo, D. Azzalini, F. Amigoni, and A. Farinelli. Adversarial data augmentation for HMM-based anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45:14131–14143, 2023b. doi: 10.1109/TPAMI.2023.3303099.

N. Charef, A. B. Mnaouer, M. Aloqaily, O. Bouachir, and M. Guizani. Artificial intelligence implication on energy sustainability in internet of things: A survey. *Information Processing and Management*, 60, 2023. doi: 10.1016/j.ipm.2022.103212.

G. Chaslot, S. Bakkes, I. Szita, and P. Spronck. Monte-Carlo tree search: A new framework for game ai. In *Proceedings of the 4th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIDE)*, pages 216–217. AAAI Press, 2008. doi: 10.1609/aiide.v4i1.18700.

H. Chen, X. Li, and F. Zhao. A reinforcement learning-based sleep scheduling algorithm for desired area coverage in solar-powered wireless sensor networks. *IEEE Sensors Journal*, 16:2763–2774, 2016. doi: 10.1109/JSEN.2016.2517084.

H. Chen, T. Zhao, C. Li, and Y. Guo. Green internet of vehicles: Architecture, enabling technologies, and applications. *IEEE Access*, 7:179185–179198, 2019. doi: 10.1109/ACCESS.2019.2958175.

K. Chen, H. Wang, B. Valverde-Pérez, S. Zhai, L. Vezzaro, and A. Wang. Optimal control towards sustainable wastewater treatment plants based on multi-agent reinforcement learning. *Chemosphere*, 279, 2021. doi: 10.1016/j.chemosphere.2021.130498.

K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pages 4759–4770. Curran Associates, Inc., 2018.

A. Cossu, A. Carta, and D. Bacciu. Continual learning with gated incremental memories for sequential data processing. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE Press, 2020. doi: 10.1109/IJCNN48605.2020.9207550.

R. Coulom. Efficient selectivity and backup operators in Monte-Carlo tree search. In *Computers and Games*, pages 72–83. Springer, 2007.

A. de Gracia, C. Fernández, A. Castell, C. Mateu, and L. F. Cabeza. Control of a PCM ventilated facade using reinforcement learning techniques. *Energy and Buildings*, 106:234–242, 2015. doi: 10.1016/j.enbuild.2015.06.045.

R. Dearden, N. Friedman, and S. Russell. Bayesian Q-learning. In *Proceedings of the 15th AAAI Conference on Artificial Intelligence (AAAI)*, pages 761–768. American Association for Artificial Intelligence, 1998.

R. Dechter. *Constraint Processing*. Morgan Kaufmann Publishers Inc., 2003.

M. P. Deisenroth and C. E. Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 465–472. Omnipress, 2011.

K. V. Delgado, S. Sanner, and L. N. de Barros. Efficient solutions to factored MDPs with imprecise transition probabilities. *Artificial Intelligence*, 175: 1498–1527, 2011. doi: https://doi.org/10.1016/j.artint.2011.01.001.

S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft. Learning and policy search in stochastic dynamical systems with Bayesian neural networks. *arXiv preprint arXiv:1605.07127*, 2016.

N. H. van Der Blij, D. Chaifouroosh, C. A. Cañizares, T. B. Soeiro, L. M. Ramirez-Elizondo, M. T. J. Spaan, and P. Bauer. Improved power flow methods for DC grids. In *Proceedings of 29th IEEE International Symposium on Industrial Electronics (ISIE)*, pages 1135–1140. IEEE Press, 2020. doi: 10.1109/ISIE45063.2020.9152570.

H. Dong, Z. Ding, S. Zhang, H. Yuan, H. Zhang, J. Zhang, Y. Huang, T. Yu, H. Zhang, and R. Huang. *Deep Reinforcement Learning: Fundamentals, Research, and Applications*. Springer Nature, 2020.

F. Doshi-Velez. The infinite partially observable Markov decision process. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 22, pages 477–485. Curran Associates, Inc., 2009.

K. Doya, K. Samejima, K. I. Katagiri, and M. Kawato. Multiple model-based reinforcement learning. *Neural Computation*, 14:1347–1369, 2002. doi: 10.1162/089976602753712972.

D. Elavarasan and P. M. Durairaj Vincent. Crop yield prediction using deep reinforcement learning model for sustainable agrarian applications. *IEEE Access*, 8:86886–86901, 2020. doi: 10.1109/ACCESS.2020.2992480.

O. Emamjomehzadeh, R. Kerachian, M. J. Emami-Skardi, and M. Momeni. Combining urban metabolism and reinforcement learning concepts for sustainable water resources management: A nexus approach. *Journal of Environmental Management*, 2023. doi: 10.1016/j.jenvman.2022.117046.

Y. Feng, X. Zhang, R. Jia, F. Lin, J. Lu, Z. Zheng, and M. Li. Intelligent trajectory design for mobile energy harvesting and data transmission. *IEEE Internet of Things Journal*, 10:403–416, 2023. doi: 10.1109/JIOT.2022.3202252.

J. Fischer and Ö. S. Taş. Information particle filter tree: An online algorithm for POMDPs with belief-based rewards on continuous domains. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119, pages 3177–3187. PMLR, 2020.

R. M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3:128–135, 1999. doi: 10.1016/S1364-6613(99)01294-2.

K. Friston. Hierarchical models in the brain. *PLOS Computational Biology*, 4:1–24, 2008. doi: 10.1371/journal.pcbi.1000211.

F. Fröhlich, F. J. Theis, and J. Hasenauer. Uncertainty analysis for non-identifiable dynamical systems: Profile likelihoods, bootstrapping and more. In *Proceedings of the 12th International Conference on Computational Methods in Systems Biology (CMSB)*, volume 8859 of *Lecture Notes in Computer Science*, pages 61–72. Springer, 2014. doi: 10.1007/978-3-319-12982-2_5.

Y. Gal, R. McAllister, and C. E. Rasmussen. Improving PILCO with Bayesian neural network dynamics models. In *Data-Efficient Machine Learning workshop, International Conference on Machine Learning (ICML)*, volume 4, 2016.

Y. Gao, D. Chang, and C. H. Chen. A digital twin-based approach for optimizing operation energy consumption at automated container terminals. *Journal of Cleaner Production*, 385, 2023. doi: 10.1016/j.jclepro.2022.135782.

M. K. Giri and S. Majumder. Deep Q-learning based optimal resource allocation method for energy harvested cognitive radio networks. *Physical Communication*, 53, 2022. doi: 10.1016/j.phycom.2022.101766.

F. Giuliari, A. Castellini, R. Berra, A. Del Bue, A. Farinelli, M. Cristani, F. Setti, and Y. Wang. POMP++: POMCP-based active visual search in unknown indoor environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1523–1530. IEEE Press, 2021. doi: 10.1109/IROS51168.2021.9635866.

A. Goldhoorn, A. Garrell, R. Alquézar, and A. Sanfeliu. Continuous real time POMCP to find-and-follow people by a humanoid service robot. In *Proceedings of the 14th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 741–747. IEEE Press, 2014. doi: 10.1109/HUMANOIDS.2014.7041445.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

R. Goodland. The concept of environmental sustainability. *Annual Review of Ecology and Systematics*, 26:1–24, 1995.

A. Graves, G. Wayne, and I. Danihelka. Neural Turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Z. Gu, Z. Liu, Q. Wang, Q. Mao, Z. Shuai, and Z. Ma. Reinforcement learning-based approach for minimizing energy loss of driving platoon decisions. *Sensors*, 23, 2023. doi: 10.3390/s23084176.

A. Guez, D. Silver, and P. Dayan. Efficient Bayes-adaptive reinforcement learning using sample-based search. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 25, pages 1034–1042. Curran Associates, Inc., 2012.

A. Guez, D. Silver, and P. Dayan. Scalable and efficient Bayes-adaptive reinforcement learning based on Monte-Carlo tree search. *Journal of Artificial Intelligence Research*, 48:841–883, 2013. doi: 10.1613/jair.4117.

A. Guez, N. Heess, D. Silver, and P. Dayan. Bayes-adaptive simulation-based search with value function approximation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 27, pages 451–459. Curran Associates, Inc., 2014.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pages 1321–1330. PMLR, 2017.

X. Guo, S. Singh, H. Lee, R. L. Lewis, and X. Wang. Deep learning for real-time Atari game play using offline Monte-Carlo tree search planning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 27, pages 3338–3346. Curran Associates, Inc., 2014.

M. Han, J. Duan, S. Khairy, and L.X. Cai. Enabling sustainable underwater IoT networks with energy harvesting: A decentralized reinforcement learning approach. *IEEE Internet of Things Journal*, pages 9953–9964, 2020. doi: 10.1109/JIOT.2020.2990733.

D. J. B. Harrold, J. Cao, and Z. Fan. Data-driven battery operation for energy arbitrage using rainbow deep reinforcement learning. *Energy*, 238, 2022. doi: 10.1016/j.energy.2021.121958.

H. van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)*, volume 30, pages 2094–2100. AAAI Press, 2016. doi: 10.1609/aaai.v30i1.10295.

M. Hausknecht and P. Stone. Deep Recurrent Q-Learning for Partially Observable MDPs. In *AAAI Fall Symposia*, pages 29–37. AAAI Press, 2015.

M. Hauskrecht. Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, 13: 33–94, 2000. doi: 10.1613/JAIR.678.

T. L. Hayes, N. D. Cahill, and C. Kanan. Memory efficient experience replay for streaming learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 9769–9776, 2019. doi: 10.1109/ICRA.2019.8793982.

W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1:660–670, 2002. doi: 10.1109/TWC.2002.804190.

M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, volume 32, pages 3215–3222. AAAI Press, 2018. doi: 10.1609/AAAI.V32I1.11796.

T. Hester and P. Stone. *Learning and Using Models*, pages 111–141. Springer, 2012. doi: 10.1007/978-3-642-27645-3_4.

Y. Himeur, M. Elnour, F. Fadli, N. Meskin, I. Petri, Y. Rezgui, F. Bensaali, and A. Amira. AI-big data analytics for building automation and management systems: A survey, actual challenges and future perspectives. *Artificial Intelligence Review*, 56:4929–5021, 2022. doi: 10.1007/s10462-022-10286-2.

R. C. Hsu, C. T. Liu, and H. L. Wang. A reinforcement learning-based ToD provisioning dynamic power management for sustainable operation of energy harvesting wireless sensor node. *IEEE Transactions on Emerging Topics in Computing*, 2:181–191, 2014. doi: 10.1109/TETC.2014.2316518.

D. Huo, Y. A. Sari, R. Kealey, and Q. Zhang. Reinforcement learning-based fleet dispatching for greenhouse gas emission reduction in open-pit mining operations. *Resources, Conservation and Recycling*, 188, 2023. doi: 10.1016/j.resconrec.2022.106664.

J. Hurtado, A. Raymond-Sáez, V. Araujo, V. Lomonaco, A. Soto, and D. Bacciu. Memory population in continual learning via outlier elimination. In *Proceedings of the IEEE International Conference on Computer Vision*

(ICCV) Workshops, pages 3473–3482. IEEE Press, 2023. doi: 10.1109/ICCVW60793.2023.00373.

I. Jendoubi and F. Bouffard. Data-driven sustainable distributed energy resources' control based on multi-agent deep reinforcement learning. *Sustainable Energy, Grids and Networks*, 32, 2022. doi: 10.1016/j.segan.2022.100919.

M. I. Jordan and D. E. Rumelhart. Forward models: Supervised learning with a distal teacher. *Cognitive Science*, 16:307–354, 1992. doi: 10.1016/0364-0213(92)90036-T.

C. Käding, E. Rodner, A. Freytag, and J. Denzler. Fine-tuning deep neural networks in continuous learning scenarios. In *Computer Vision – ACCV 2016 Workshops*, pages 588–605. Springer International Publishing, 2017. doi: 10.1007/978-3-319-54526-4_43.

L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998. doi: https://doi.org/10.1016/S0004-3702(98)00023-X.

M. Karl, M. Soelch, J. Bayer, and P. Van der Smagt. Deep variational Bayes filters: Unsupervised learning of state space models from raw data. *arXiv preprint arXiv:1605.06432*, 2016.

A. Kathirgamanathan, E. Mangina, and D. P. Finn. Development of a soft actor critic deep reinforcement learning approach for harnessing energy flexibility in a large office building. *Energy and AI*, 5, 2021. doi: 10.1016/j.egyai.2021.100101.

S. Katt, F. A. Oliehoek, and C. Amato. Learning in POMDPs with Monte Carlo tree search. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pages 1819–1827. JMLR.org, 2017.

S. Katt, F. A. Oliehoek, and C. Amato. Bayesian reinforcement learning in factored POMDPs. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 7–15. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2019.

M. Khalid, L. Wang, K. Wang, N. Aslam, C. Pan, and Y. Cao. Deep reinforcement learning-based long-range autonomous valet parking for smart cities. *Sustainable Cities and Society*, 89, 2023. doi: 10.1016/j.scs.2022.104311.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114.

L. Kocsis and C. Szepesvári. Bandit based Monte-Carlo planning. In *Proceedings of the 17th European Conference on Machine Learning (ECML)*, pages 282–293, 2006. doi: 10.1007/11871842_29.

N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2149–2154. IEEE Press, 2004. doi: 10.1109/IROS.2004.1389727.

A. M. Koufakis, E. S. Rigas, N. Bassiliades, and S. D. Ramchurn. Offline and online electric vehicle charging scheduling with V2V energy transfer. *IEEE Transactions on Intelligent Transportation Systems*, 21:2128–2138, 2020. doi: 10.1109/TITS.2019.2914087.

M. Lauri and R. Ritala. Planning for robotic exploration based on forward simulation. *Robotics and Autonomous Systems*, 83:15–31, 2016. doi: 10.1016/j.robot.2016.06.008.

S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical report, 1998.

Y. Le Cun. *Generalization and network design strategies*. Elsevier, 1989.

J. Lee, G. H. Kim, P. Poupart, and K. E. Kim. Monte-Carlo tree search for constrained POMDPs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pages 7934–7943. Curran Associates Inc., 2018.

J. Leng, G. Ruan, Y. Song, Q. Liu, Y. Fu, K. Ding, and X. Chen. A loosely-coupled deep reinforcement learning approach for order acceptance decision of mass-individualized printed circuit board manufacturing in Industry 4.0. *Journal of Cleaner Production*, 280, 2021. doi: 10.1016/j.jclepro.2020.124405.

T. Lesort, H. Caselles-Dupré, M. Garcia Ortiz, A. Stoian, and D. Filliat. Generative models from the perspective of continual learning. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE Press, 2019. doi: 10.1109/IJCNN.2019.8851986.

T. Lesort, V. Lomonaco, A. Stoian, D. Maltoni, D. Filliat, and N. Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information Fusion*, 58:52–68, 2020. doi: 10.1016/j.inffus.2019.12.004.

S. Levine and P. Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 27, pages 1071–1079. Curran Associates, Inc., 2014.

C. Li, L. Bai, L. Yao, S. T. Waller, and W. Liu. A bibliometric analysis and review on reinforcement learning for transportation applications. *Transportmetrica B*, 11, 2023. doi: 10.1080/21680566.2023.2179461.

Z. Li and D. Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017. doi: 10.1109/TPAMI.2017.2773081.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

L. J. Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8:293–321, 1992. doi: 10.1007/BF00992699.

L. J. Lin and T. Mitchell. Memory approaches to reinforcement learning in non-Markovian domains. Technical report, 1992.

Q. Liu, S. Sun, B. Rong, and M. Kadoch. Intelligent reflective surface based 6G communications for sustainable energy infrastructure. *IEEE Wireless Communications*, 28:49–55, 2021. doi: 10.1109/MWC.016.2100179.

R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 6382–6393. Curran Associates Inc., 2017.

J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31 (12):2346–2363, 2019a. doi: 10.1109/TKDE.2018.2876857.

J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31: 2346–2363, 2019b. doi: 10.1109/TKDE.2018.2876857.

F. M. Luo, T. Xu, H. Lai, X.-H. Chen, W. Zhang, and Y. Yu. A survey on model-based reinforcement learning. *arXiv preprint arXiv:2206.09328*, 2022.

D. Ma, G. Lan, M. Hassan, W. Hu, and S.K. Das. Sensing, computing, and communications for energy harvesting IoTs: A survey. *IEEE Communications Surveys and Tutorials*, 22:1222–1250, 2020. doi: 10.1109/COMST.2019.2962526.

P. Mabina, P. Mukoma, and M. J. Booysen. Sustainability matchmaking: Linking renewable sources to electric water heating through machine learning. *Energy and Buildings*, 246, 2021. doi: 10.1016/j.enbuild.2021.111085.

A. Mallya and S. Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7765–7773. IEEE Press, 2018. doi: 10.1109/CVPR.2018.00810.

E. Marchesini, D. Corsi, and A. Farinelli. Benchmarking safe deep reinforcement learning in aquatic navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5590–5595. IEEE Press, 2021. doi: 10.1109/IROS51168.2021.9635925.

E. Marder-Eppstein, E. Berger, T. Foote, B. P. Gerkey, and K. Konolige. The office marathon: Robust navigation in an indoor office environment. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 300–307. IEEE Press, 2010. doi: 10.1109/ROBOT.2010.5509725.

G. Mazzi, A. Castellini, and A. Farinelli. Policy interpretation for Partially Observable Monte-Carlo Planning: A rule-based approach. In *Proceedings of the 7th Italian Workshop on Artificial Intelligence and Robotics - A workshop of the 19th International Conference of the Italian Association for Artificial Intelligence (AIRO@AI*IA)*, volume 2806, pages 44–48. CEUR-WS.org, 2020.

G. Mazzi, A. Castellini, and A. Farinelli. Identification of unexpected decisions in partially observable Monte-Carlo planning: A rule-based approach. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 889–897. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2021a.

G. Mazzi, A. Castellini, and A. Farinelli. Rule-based shielding for partially observable Monte-Carlo planning. In *Proceedings of the 31st International Conference on Automated Planning and Scheduling (ICAPS)*, volume 31, pages 243–251. AAAI Press, 2021b. doi: 10.1609/icaps.v31i1.15968.

G. Mazzi, A. Castellini, and A. Farinelli. Rule-based shield synthesis for Partially Observable Monte Carlo Planning. In *Proceedings of the 3rd Workshop on Artificial Intelligence and Formal Verification, Logic, Automata, and Synthesis - A workshop of the 12th International Symposium on Games, Automata, Logics, and Formal Verification (OVERLAY@GandALF)*, volume 2987, pages 19–23. CEUR-WS.org, 2021c.

G. Mazzi, A. Castellini, and A. Farinelli. Active generation of logical rules for POMCP shielding. In *Proceedings of the 21st International Conference on*

Autonomous Agents and Multiagent Systems (AAMAS), pages 1696–1698. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2022. doi: 10.5555/3535850.3536080.

G. Mazzi, A. Castellini, and A. Farinelli. Risk-aware shielding of partially observable Monte Carlo planning policies. *Artificial Intelligence*, 324, 2023a. doi: 10.1016/J.ARTINT.2023.103987.

G. Mazzi, D. Meli, A. Castellini, and A. Farinelli. Learning logic specifications for soft policy guidance in POMCP. In *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 373–381. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2023b.

D. A. McAllester and S. Singh. Approximate planning for factored POMDPs using belief state simplification. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 409–416. Morgan Kaufmann Publishers Inc., 1999.

D. Meli, G. Mazzi, A. Castellini, and A. Farinelli. From POMDP executions to policy specifications. In *Proceedings of the 4rd Workshop on Artificial Intelligence and Formal Verification, Logic, Automata, and Synthesis - A workshop of the 21st International Conference of the Italian Association for Artificial Intelligence (OVERLAY@AI*IA)*, volume 3311, pages 93–98. CEUR-WS.org, 2022.

M. Miozzo, L. Giupponi, M. Rossi, and P. Dini. Distributed Q-learning for energy harvesting heterogeneous networks. In *IEEE International Conference on Communication Workshop (ICCW)*, pages 2006–2011. IEEE Press, 2015. doi: 10.1109/ICCW.2015.7247475.

M. Miozzo, L. Giupponi, M. Rossi, and P. Dini. Switch-on/off policies for energy harvesting small cells through distributed Q-learning. In *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 1–6. IEEE Press, 2017. doi: 10.1109/WCNCW.2017.7919075.

S. Mischos, E. Dalagdi, and D. Vrakas. Intelligent energy management systems: A review. *Artificial Intelligence Review*, 56:11635–11674, 2023. doi: 10.1007/s10462-023-10441-3.

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015. doi: 10.1038/nature14236.

T. M. Moerland, J. Broekens, and C. M. Jonker. Learning multimodal transition dynamics for model-based reinforcement learning. *arXiv preprint arXiv:1705.00470*, 2017.

T. M. Moerland, J. Broekens, A. Plaat, and C. M. Jonker. A0C: alpha zero in continuous action space. *arXiv preprint arXiv:1805.09613*, 2018.

T. M. Moerland, J. Broekens, A. Plaat, and C. M. Jonker. Model-based reinforcement learning: A survey. *Foundations and Trends in Machine Learning*, 16:1–118, 2023. doi: 10.1561/2200000086.

D. C. Montgomery and G. C. Runger. *Applied statistics and probability for engineers*. John Wiley & Sons, 2010.

A. W. Moore and C. G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, 13:103–130, 1993. doi: 10.1023/A:1022635613229.

K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE Press, 2018. doi: 10.1109/ICRA.2018.8463189.

K. S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1: 4–27, 1990. doi: 10.1109/72.80202.

A. S. Nemirovski and D. B. Yudin. Cesari convergence of the gradient method of approximating saddle points of convex-concave functions. In *Doklady Akademii Nauk*, volume 239, pages 1056–1059. Russian Academy of Sciences, 1978.

A. Y. Ng, D. Harada, and S. J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 278–287. Morgan Kaufmann Publishers Inc., 1999.

R. Nian, J. Liu, and B. Huang. A review on reinforcement learning: Introduction and applications in industrial process control. *Computers and Chemical Engineering*, 139, 2020. doi: 10.1016/j.compchemeng.2020.106886.

D. Ognibene, L. Mirante, and L. Marchegiani. Proactive intention recognition for joint human-robot search and rescue missions through Monte-Carlo planning in POMDP environments. In *Proceedings of the 11th International Conference on Social Robotics (ICSR)*, Lecture Notes in Computer Science, pages 332–343. Springer, 2019. doi: 10.1007/978-3-030-35888-4_31.

J. Oh, X. Guo, H. Lee, R. Lewis, and S. Singh. Action-conditional video prediction using deep networks in Atari games. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, pages 2863–2871. Curran Associates, Inc., 2015.

S. Orfanoudakis and G. Chalkiadakis. A novel aggregation framework for the efficient integration of distributed energy resources in the smart grid. In *Proceedings of the 22nd International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 2514–2516. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2023. doi: 10.5555/3545946.3598986.

C. Ounoughi, G. Touibi, and S.B. Yahia. Ecolight: Eco-friendly traffic signal control driven by urban noise prediction. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pages 205–219, 2022. doi: 10.1007/978-3-031-12423-5_16.

A. A. Panagopoulos, M. Alam, A. Rogers, and N. R. Jennings. AdaHeat: A general adaptive intelligent agent for domestic heating control. In *Proceedings of the 14th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 1295–1303. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2015.

C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12:441–450, 1987.

R. Parr, L. Li, G. Taylor, C. Painter-Wakefield, and M. L. Littman. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th International Conference on Machine learning (ICML)*, pages 752–759. Association for Computing Machinery (ACM), 2008. doi: 10.1145/1390156.1390251.

B. K. Patle, G. Babu L, A. Pandey, D. R. K. Parhi, and A. Jagadeesh. A review: On path planning strategies for navigation of mobile robot. *Defence Technology*, 15(4):582–606, 2019. doi: 10.1016/j.dt.2019.04.011.

A. Perianes-Rodriguez, L. Waltman, and N. J. van Eck. Constructing bibliometric networks: A comparison between full and fractional counting. *Journal of Informetrics*, 10:1178–1195, 2016. doi: 10.1016/j.joi.2016.10.006.

J. Pineau, N. Roy, and S. Thrun. A hierarchical approach to pomdp planning and execution. In *Proceedings of the Workshop on Hierarchy and Memory in Reinforcement Learning - A workshop of the 18th International Conference on Machine Learning (ICML)*, 2001.

J. Pineau, G. Gordon, and S. Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, 27:335–380, 2006. doi: 10.1613/jair.2078.

J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10 (3):61–74, 1999.

P. Pletscher, C. S. Ong, and J. Buhmann. Spanning tree approximations for conditional random fields. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 5, pages 408–415. PMLR, 2009.

M. L. Puterman. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc., 1994. doi: 10.1002/9780470316887.

S. Radini, E. Marinelli, Ç. Akyol, A. L. Eusebi, V. Vasilaki, A. Mancini, E. Frontoni, G. B. Bischetti, C. Gandolfi, E. Katsou, and F. Fatone. Urban water-energy-food-climate nexus in integrated wastewater and reuse systems: Cyber-physical framework and innovations. *Applied Energy*, 298, 2021. doi: 10.1016/j.apenergy.2021.117268.

L. Rampini and F. Re Cecconi. Artificial intelligence in construction asset management: A review of present status, challenges and future opportunities. *Journal of Information Technology in Construction*, 27:884–913, 2022. doi: 10.36680/j.itcon.2022.043.

D. Rangel-Martinez, K. D. P. Nigam, and L. A. Ricardez-Sandoval. Machine learning on sustainable energy: A review and outlook on renewable energy systems, catalysis, smart grid and energy storage. *Chemical Engineering Research and Design*, 174:414–441, 2021. doi: 10.1016/j.cherd.2021.08.013.

S. A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. iCaRL: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5533–5542. IEEE Press, 2017. doi: 10.1109/CVPR.2017.587.

M. Roncalli, F. Bistaffa, and A. Farinelli. Decentralized power distribution in the smart grid with ancillary lines. *Mobile Networks and Applications*, 24: 1654–1662, 2019. doi: 10.1007/s11036-017-0893-y.

S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, 2008. doi: 10.1613/JAIR.2567.

S. Ross, J. Pineau, B. Chaib-draa, and P. Kreitmann. A Bayesian approach for learning and planning in partially observable Markov decision processes. *Journal of Machine Learning Research*, 12:1729–1770, 2011.

S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

S. J. Russell and P. Norvig. *Artificial intelligence - A modern approach*. Pearson Education, 2010.

A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

S. Sabet and B. Farooq. Green vehicle routing problem: State of the art and future directions. *IEEE Access*, 10:101622–101642, 2022. doi: 10.1109/ACCESS.2022.3208899.

A. Sacco, F. Esposito, G. Marchetto, and P. Montuschi. Sustainable task offloading in UAV networks via multi-agent reinforcement learning. *IEEE Transactions on Vehicular Technology*, 70:5003–5015, 2021. doi: 10.1109/TVT.2021.3074304.

R. Salakhutdinov. Learning in Markov random fields using tempered transitions. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 22, pages 1598–1606. Curran Associates, Inc., 2009.

M. Sangermano, A. Carta, A. Cossu, and D. Bacciu. Sample condensation in online continual learning. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE Press, 2022. doi: 10.1109/IJCNN55064.2022.9892299.

S. Sanner. Relational dynamic influence diagram language (RDDL): Language description. 2010.

J. K. Satia and R. E. Lave. Markovian decision processes with uncertain transition probabilities. *Operations Research*, 21(3):728–740, 1973.

J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, volume 37, pages 1889–1897. PMLR, 2015.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

G. Shafer and V. Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9:371–421, 2008.

A. Shah, D. Shah, and G. W. Wornell. On learning continuous pairwise Markov random fields. In *Proceedings of the the 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 130, pages 1153–1161. PMLR, 2021.

R. Shaw, E. Howley, and E. Barrett. Applying reinforcement learning towards automating energy efficient virtual machine consolidation in cloud data centers. *Information Systems*, 107, 2022. doi: 10.1016/j.is.2021.101722.

A. Sheikhi, M. Rayati, and A. M. Ranjbar. Dynamic load management for a residential customer; Reinforcement learning approach. *Sustainable Cities and Society*, 24:42–51, 2016. doi: 10.1016/j.scs.2016.04.001.

H. Shin, J. K. Lee, J. Kim, and J. Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30. Curran Associates, Inc., 2017.

D. Silver. *Reinforcement learning and simulation based search in computer Go*. PhD thesis, University of Alberta, Edmonton, Alta., Canada, 2009.

D. Silver and J. Veness. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 23, pages 2164–2172. Curran Associates, Inc., 2010.

D. Silver, R. S. Sutton, and M. Müller. Sample-based learning and search with permanent and transient memories. In *Proceedings of the 25th International Conference on Machine learning (ICML)*, volume 307, pages 968–975. Association for Computing Machinery (ACM), 2008. doi: 10.1145/1390156.1390278.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016. doi: 10.1038/NATURE16961.

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550:354–359, 2017. doi: 10.1038/NATURE24270.

T. D. Simão, M. Suilen, and N. Jansen. Safe policy improvement for POMDPs via finite-state controllers. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI)*, volume 37, pages 15109–15117. AAAI Press, 2023. doi: 10.1609/aaai.v37i12.26763.

K. Sivamayil, E. Rajasekar, B. Aljafari, S. Nikolovski, S. Vairavasundaram, and I. Vairavasundaram. A systematic study on reinforcement learning based applications. *Energies*, 16, 2023. doi: 10.3390/en16031512.

M. J. E. Skardi, R. Kerachian, and A. Abdolhay. Water and treated wastewater allocation in urban areas considering social attachments. *Journal of Hydrology*, 2020. doi: 10.1016/j.jhydrol.2020.124757.

R. D. Smallwood and E. J. Sondik. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research*, 21:1071–1088, 1973. doi: 10.1287/opre.21.5.1071.

T. Smith and R. Simmons. Heuristic search value iteration for POMDPs. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 520–527. AUAI Press, 2004.

E. J. Sondik. The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs. *Operations Research*, 26:282–304, 1978. doi: 10.1287/opre.26.2.282.

M. T. J. Spaan and N. A. Vlassis. A point-based POMDP algorithm for robot planning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2399–2404. IEEE Press, 2004. doi: 10.1109/ROBOT.2004.1307420.

M. T. J. Spaan, T. S. Veiga, and P. U. Lima. Decision-theoretic planning under uncertainty with information rewards for active cooperative perception. In *Proceeding of the 14th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, volume 29, pages 1157–1185. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2015. doi: 10.1007/s10458-014-9279-8.

M. Sridharan, J. L. Wyatt, and R. Dearden. HiPPo: Hierarchical POMDPs for planning information processing and sensing actions on a robot. In *Proceedings of the 18th International Conference on Automated Planning and Scheduling (ICAPS)*, pages 346–354. AAAI Press, 2008.

C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Robotics: Science and Systems*, volume 2, pages 65–72, 2005. doi: 10.15607/RSS.2005.I.009.

L. Steccanella, D. D. Bloisi, A. Castellini, and A. Farinelli. Waterline and obstacle detection in images from low-cost autonomous boats for environmental monitoring. *Robotics and Autonomous Systems*, 124, 2020. doi: 10.1016/j.robot.2019.103346.

S. J. Sultanuddin, R. Vibin, A. R. Kumar, N. R. Behera, M. J. Pasha, and K. K. Baseer. Development of improved reinforcement learning smart charging strategy for electric vehicle fleet. *Journal of Energy Storage*, 2023. doi: 10.1016/j.est.2023.106987.

Z. N. Sunberg and M. J. Kochenderfer. Online algorithms for POMDPs with continuous state, action, and observation spaces. In *Proceedings of the 28th International Conference on Automated Planning and Scheduling (ICAPS)*, pages 259–263. AAAI Press, 2018.

R. S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the 7th International Conference on Machine Learning (ICML)*, pages 216–224. Morgan Kaufmann, 1990. doi: 10.1016/B978-1-55860-141-3.50030-4.

R. S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *SIGART Bulletin*, 2(4):160–163, 1991. doi: 10.1145/122344.122377.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. A Bradford Book, 2018.

R. S. Sutton, C. Szepesvári, A. Geramifard, and M. Bowling. Dyna-style planning with linear function approximation and prioritized sweeping. In *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 528–536. AUAI Press, 2008.

A. Tamar, D. Di Castro, and S. Mannor. Learning the variance of the reward-to-go. *Journal of Machine Learning Research*, 17:361–396, 2016.

X. Tang, P. K. Misztal, W. W. Nazaroff, and A. H. Goldstein. Volatile organic compound emissions from humans indoors. *Environmental Science & Technology*, 50:12686–12694, 2016. doi: 10.1021/acs.est.6b04415.

T. Teleszewski and K. Gładyszewska-Fiedoruk. The concentration of carbon dioxide in conference rooms: A simplified model and experimental verification. *International Journal of Environmental Science and Technology*, 16: 8031–8040, 2019. doi: 10.1007/s13762-019-02412-5.

G. Theocharous, K. Rohanimanesh, and S. Mahadevan. Learning hierarchical observable Markov decision process models for robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 511–516, 2001. doi: 10.1109/ROBOT.2001.932601.

G. Theocharous, K. P. Murphy, and L. P. Kaelbling. Representing hierarchical POMDPs as DBNs for multi-scale robot localization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 1045–1051. IEEE Press, 2004. doi: 10.1109/ROBOT.2004.1307288.

V. Thomas, G. Hutin, and O. Buffet. Monte Carlo information-oriented planning. In *Proceedings of the 24th European Conference on Artificial Intelligence (ECAI)*, volume 325, pages 2378–2385. IOS Press, 2020. doi: 10.3233/FAIA200368.

S. Thrun. Monte Carlo POMDPs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 12, pages 1064–1070. Curran Associates, Inc., 2000.

United Nations. Transforming our world: The 2030 agenda for sustainable development. 2015.

G. Upton and I. Cook. *A dictionary of statistics*. Oxford quick reference, 2008.

P. E. Utgoff. Incremental induction of decision trees. *Machine Learning*, 4(2): 161–186, 1989. doi: 10.1023/A:1022699900025.

T. Veiga, M. Spaan, and P. Lima. Point-based POMDP solving with factored value function approximation. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI)*, volume 28, pages 2513–2519. AAAI Press, 2014. doi: 10.1609/AAAI.V28I1.9070.

T. S. Veiga. *Information Gain and Value Function Approximation in Task Planning Using POMDPs*. PhD thesis, Instituto Superior Técnico, Universidade de Lisboa, 2015.

V. Venkataswamy, J. Grigsby, A. Grimshaw, and Y. Qi. RARE: Renewable energy aware resource management in datacenters. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 13592:108–130, 2023. doi: 10.1007/978-3-031-22698-4_6.

T. Vodopivec, S. Samothrakis, and B. Šter. On Monte Carlo tree search and reinforcement learning. *Journal of Artificial Intelligence Research*, 60(1): 881–936, 2017. doi: 10.1613/jair.5507.

V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag, 2005.

M. Vuffray, S. Misra, and A. Y. Lokhov. Efficient learning of discrete graphical models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 13575–13585. Curran Associates Inc., 2020.

E. Walraven and M. T. J. Spaan. Point-based value iteration for finite-horizon POMDPs. *Journal of Artificial Intelligence Research*, 65:307–341, 2019. doi: 10.1613/jair.1.11324.

H. Wang, N. Liu, Y. Zhang, D. Feng, F. Huang, D. S. Li, and Y. Zhang. Deep reinforcement learning: A survey. *Frontiers of Information Technology and Electronic Engineering*, 21:1726–1744, 2020a. doi: 10.1631/FITEE.1900533.

J. J. Wang and L. Wang. A cooperative memetic algorithm with learning-based agent for energy-aware distributed hybrid flow-shop scheduling. *IEEE Transactions on Evolutionary Computation*, 26:461–475, 2022. doi: 10.1109/TEVC.2021.3106168.

Y. Wang, F. Giuliari, R. Berra, A. Castellini, A. Del Bue, A. Farinelli, M. Cristani, and F. Setti. POMP: POMCP-based online motion planning

for active visual search in indoor environments. In *Proceedings of the 31st British Machine Vision Conference (BMVC)*. BMVA Press, 2020b.

Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, volume 48, pages 1995–2003. PMLR, 2016.

C. J. C. H. Watkins. *Learning from delayed rewards*. PhD thesis, King's College, Oxford, 1989.

C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992. doi: 10.1007/BF00992698.

O. Wertheim, R. I. Brafman, S. Shekhar, T. Feiner, and I. Pinsky. ROS-POMDP - A platform for robotics planning using PLPs and RDDL in ROS. In *Proceedings of the 30th International Conference on Automated Planning and Scheduling (ICAPS)*. AAAI Press, 2020.

C. C. White and H. K. Eldeib. Markov decision processes with imprecise transition probabilities. *Operations Research*, 42:739–749, 1994. doi: 10.1287/opre.42.4.739.

G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996. doi: 10.1007/BF00116900.

J. D. Williams and S. Young. Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21(2):393–422, 2007. doi: 10.1016/j.csl.2006.06.008.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696.

T. Yang, L. Zhao, W. Li, and A. Y. Zomaya. Reinforcement learning in sustainable energy and electric systems: A survey. *Annual Reviews in Control*, 49:145–163, 2020. doi: 10.1016/j.arcontrol.2020.03.001.

R. Yao, Y. Hu, and L. Varga. Applications of agent-based methods in multi-energy systems Ñ A systematic literature review. *Energies*, 16, 2023. doi: 10.3390/en16052456.

L. Yu, W. Zhang, J. Wang, and Y. Yu. SeqGAN: Sequence generative adversarial nets with policy gradient. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*, volume 31, pages 2852–2858. AAAI Press, 2017. doi: 10.1609/AAAI.V31I1.10804.

B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 694–699. Association for Computing Machinery (ACM), 2002. doi: 10.1145/775047.775151.

F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 3987–3995. PMLR, 2017.

W. Zhang, H. Liu, F. Wang, T. Xu, H. Xin, D. Dou, and H. Xiong. Intelligent electric vehicle charging recommendation based on multi-agent reinforcement learning. In *Proceedings of the World Wide Web Conference (WWW)*. Association for Computing Machinery (ACM), 2021a. doi: 10.1145/3442381.3449934.

X. Zhang, G. Manogaran, and B. Muthu. IoT enabled integrated system for green energy into smart cities. *Sustainable Energy Technologies and Assessments*, 46, 2021b. doi: 10.1016/j.seta.2021.101208.

M. Zuccotto, A. Castellini, M. Piccinelli, E. Marchesini, and A. Farinelli. Learning environment properties in partially observable Monte Carlo planning. In *Proceedings of the 8th Italian Workshop on Artificial Intelligence and Robotics - A workshop of the 20th International Conference of the Italian Association for Artificial Intelligence (AIRO@AI*IA)*, volume 3162, pages 50–57. CEUR-WS.org, 2021.

M. Zuccotto, A. Castellini, and A. Farinelli. Learning state-variable relationships for improving POMCP performance. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing (SAC)*, pages 739–747. Association for Computing Machinery (ACM), 2022a. doi: 10.1145/3477314.3507049.

M. Zuccotto, M. Piccinelli, A. Castellini, E. Marchesini, and A. Farinelli. Learning state-variable relationships in POMCP: A framework for mobile robots. *Frontiers in Robotics and AI*, 9:663–704, 2022b. doi: 10.3389/FROBT.2022.819107.

M. Zuccotto, A. Castellini, D. La Torre, L. Mola, and A. Farinelli. Reinforcement learning applications in environmental sustainability: A review. *Artificial Intelligence Review*, 57, 2024a. doi: 10.1007/S10462-024-10706-5.

M. Zuccotto, E. Fusa, A. Castellini, and A. Farinelli. Online model adaptation in Monte Carlo tree search planning. *Optimization and Engineering - Special Issue on Machine Learning and Inverse Problems*, 2024b. doi: 10.1007/s11081-024-09896-2.

Appendix A

Air quality domain: full details

A.1 True Environment Model

In this section, we give full details about the simulator described in Section 5.3. The simulator is used in Chapter 5 as the real-world environment model, i.e., the true one, to evaluate the model learning and adaptation methods for MCTS proposed in Chapter 5. Taking inspiration from Teleszewski and Gładyszewska-Fiedoruk (2019); Tang et al. (2016) we developed a realistic simulator of the environment to test our framework in silico. We notice that the simulator is inspired by models of real-world systems but it has been extended with elements (e.g., room occupancy and *VOC* dynamics) useful for our application domain. For those elements we introduced complex transition functions that are interesting from a computational viewpoint but which are not validated with physical experiments in the real-world.

A.1.1 State

The *state* contains: *i*) number of persons present in the room, *ii)* the concentrations of CO_2 in the room, *iii*) the concentrations of *VOC* in the room, *iv*) the indoor temperature, and *v*) the outdoor temperature.

A.1.2 Actions

Actions are related to ventilation system, namely *stack ventilation* (open/close windows) or *mechanical ventilation* (turn on vents at low/high speed or turn off), *sanitization* system (turn on/off sanitizers) and combinations of them. Each action is characterized by a specific value called ACH represented with δ_a (see Table A.1), which indicates the number of times the entire room air volume is replaced in one hour with that action. This characteristic allows actions to affect the environment air quality (which is part of the reward function as detailed later) since the air renewal process acts on the concentrations of CO_2 and *VOC*.

Table A.1: Description of domain's actions and the corresponding ACH values.

Action	Description	Ventilation	Sanitization	δ_a (h^{-1})
ALL_OFF	Windows closed, all devices off	No	No	0.1
WINDOW	Windows opened, all devices off	Stack	No	8.0
VENT$_L$	Windows closed, only vents on (low speed)	Mechanical	No	11.0
VENT$_H$	Windows closed, only vents on (high speed)	Mechanical	No	21.0
SANITIZER	Windows closed, only sanitizers on	No	Yes	0.1
WINDOW-SANITIZER	Windows opened and sanitizers on	Stack	Yes	8.0
VENT$_L$-SANITIZER	Vents (low speed) and sanitizers on	Mechanical	No	11.0
VENT$_H$-SANITIZER	Vents (high speed) and sanitizers on	Mechanical	No	21.0

A.1.3 Transition Model

For each state-action pair the model returns the new state reached by performing the action from the state. The model consists of three *sub-models* used to describe the evolution of CO_2 concentration, *VOC* concentration, and indoor temperature, respectively.

Evolution of CO_2 Concentration

It considers both *stack ventilation system*, characterized by actions with $\delta_a <$ 9.97 h^{-1}, and *mechanical ventilation systems*, related to actions with $\delta_a \geq$ 9.97 h^{-1}.

- *Stack ventilation system in populated room.* We consider the model proposed in Teleszewski and Gładyszewska-Fiedoruk (2019) that computes the next value of CO_2 concentration ($c^{i+1}_{CO_2}$ (ppm)), as a function of the number of persons (p^i), the room volume (V) up to 420 m^3, the ACH (δ_a) and the time (t):

$$c^{i+1}_{CO_2} = A \cdot v \cdot k \cdot t + c^i_{CO_2}, \qquad v = p^i/V \tag{A.1}$$

$$k = B + C \cdot \delta_a + D \cdot \delta_a^{1.5} + E \cdot \delta_a^2 + F \cdot \delta_a^3 \tag{A.2}$$

  where $A =$ 180 m$^3 \cdot$ ppm $\cdot$ person$^{-1} \cdot$ min^{-1}, $t =$ 5 min and $B = 1.350$, $C = -1.261$, $D = 0.945$, $E = -0.236$, $F = 0.005$ (every coefficient has its own unit of measure to make k dimensionless). Equation (A.1) can be used only with *stack ventilation compatible actions*, since Equation (A.2) has a local minimum at $\delta_a =$ 9.97 h^{-1} and with mechanical ventilation compatible actions ($\delta_a \geq$ 9.97 h^{-1}) it becomes an increasing function.

- *Stack ventilation system in unpopulated room.* It could be either *active* (actions with $\delta_a >$ 0.10) or *inactive* (actions with $\delta_a =$ 0.10). In the *active* case we also consider the closeness between the CO_2 concentration in the room and the outdoor one. Then, at time i, we compute

$$\Delta c^i_{CO_2} = c^i_{CO_2} - c^{i,out}_{CO_2}. \tag{A.3}$$

In this case the estimated CO_2 concentration is given by Equation (A.4). If the value of $\Delta c^i_{CO_2}$ is higher than the threshold $\varepsilon_1 = 100$ ppm we compute CO_2 concentration at time $i+1$ as a function of ACH (δ_a) and CO_2 concentration at time i ($c^i_{CO_2}$), otherwise it is expressed in terms of CO_2 concentration at time i ($c^i_{CO_2}$), the difference between the internal and external CO_2 concentration at time i ($\Delta c^i_{CO_2}$) and the threshold (ε_1):

$$c^{i+1}_{CO_2} = \begin{cases} (1 - \frac{\delta_a}{200}) \cdot c^i_{CO_2} & \Delta c^i_{CO_2} > \varepsilon_1 \\ c^i_{CO_2} - \frac{(\Delta c^i_{CO_2})^2}{20 \cdot \varepsilon_1} & otherwise. \end{cases} \tag{A.4}$$

In the *inactive case*, instead, the concentration of CO_2 must not change over time, i.e., $c^{i+1}_{CO_2} = c^i_{CO_2}$ (following the observations provided in Teleszewski and Gładyszewska-Fiedoruk (2019)). Then Equation (A.1) holds, since $\alpha = 0$ when $p^i = 0$.

- *Mechanical ventilation system.* We introduce a new equation to compute the CO_2 concentration in the case of *mechanical ventilation compatible actions*. Equation (A.5) computes the value of $c^{i+1}_{CO_2}$ as an exponential function of k'. Notice that when $c^{i+1}_{CO_2} < c^{i,out}_{CO_2}$ we always set the value of $c^{i+1}_{CO_2} = c^{i,out}_{CO_2}$.

$$c^{i+1}_{CO_2} = \exp\left(\frac{15.20 - k'}{35.00}\right) \cdot c^i_{CO_2} \tag{A.5}$$

with

$$k' = \begin{cases} m_{k'} \cdot p^i + q_{k'} & \delta_a \in (9.97, 15.00) \\ \frac{p_{max} \cdot \delta_a}{\frac{10.00}{49.00} \cdot p^i + 39.80} & \delta_a \in [15.00, +\infty) \end{cases} \tag{A.6}$$

The computation of k' (Equation (A.6)) changes with respect to the ACH (δ_a), since we want to imitate a different evolution for the CO_2 concentration on the basis of the air renewal effect that each action has on the environment.

The values of $m_{k'}$ and $q_{k'}$ are the result of a computation to find the linear equation (i.e., the first branch of Equation (A.6)) given p_{max}, g_{min} and g_{max}. In particular,

$$m_{k'} = \frac{g_{min} - g_{max}}{p_{max}}, \qquad q_{k'} = g_{max} \tag{A.7}$$

since the two points of the line are (0, g_{max}) and (p_{max}, g_{min}). Then,

$$\begin{aligned} g_{min} &= 14.90000 \\ g_{max} &= 0.36660 * \delta_a + 13.34499 \end{aligned} \tag{A.8}$$

Moreover, p_{max} in Equations (A.6) and (A.7) represents the maximum occupancy of the room, that we assume to be a constant value set to 50 persons. Notice that by changing the value of p_{max}, the equations

for the mechanical ventilation system do not longer hold. In particular, with δ_a = 15.00, we can observe a relevant jump discontinuity for k' (using the same value of p^i for both cases of Equation (A.6)), and thus for $c_{CO_2}^{i+1}$ as well. For instance, consider two actions a_1 and a_2 with $\delta_{a_1} = 14.90$ h^{-1} and $\delta_{a_2} = 15.00$ h^{-1}. If we set $p_{max} = 100$ and compute the first case of Equation (A.6) with δ_{a_1} and the second one with δ_{a_2} (using $p^i = 50$ for both cases), we get k'_{a_1} = 16.854 (instead of the original 14.900) and k'_{a_2} = 29.998 (instead of the original 14.999) respectively. Thus, moving only 0.1 h^{-1} from δ_{a_1} to δ_{a_2}, we observe the jump discontinuity between k'_{a_1} and k'_{a_2}. This also causes an inconsistency in the computation of $c_{CO_2}^{i+1}$: with k'_{a_1} the term that multiplies $c_{CO_2}^i$ in Equation (A.5) becomes 0.954, while with k'_{a_2} becomes 0.655 .

In equation (A.6), the numerical terms are not dimensionless because of the need to make k' dimensionless. This also happens in equations (A.4), (A.8), (A.13), (A.14), (A.15) and in the expert's model equations which are relative to CO_2 and VOC variation. Moreover, in equations (A.4) and (A.5) the term t, representing the time difference between two subsequent time-steps, is not present. The rationale is that we assume to acquire samples every 5 minutes (i.e., $t = 5$ min) and it is implicit in these equations.

Evolution of VOC Concentration

The estimated concentration of *VOC* (c_{VOC}^{i+1} (μg $\cdot$ m^{-3})) depends on the concentration of *VOC* that each person emits in the room (VOC_p), and the *VOC* removed by the sanitizer (VOC_r^i), the number of persons (p^i) present in the room at time i, the room volume (V), the *VOC* concentration (c_{VOC}^i) at time i and the relative variation of CO_2 concentration ($\phi_{c_{CO_2}}$):

$$c_{VOC}^{i+1} = \left(\frac{VOC_p \cdot p^i - VOC_r^i}{V} + c_{VOC}^i \right) \cdot (1 + \phi_{c_{CO_2}}) \tag{A.9}$$

where

$$VOC_p = \frac{VOC_h}{60 \text{ min} \cdot \text{h}^{-1}} \cdot t = 520.83 \, \mu\text{g} \cdot \text{person}^{-1} \tag{A.10}$$

with VOC_h = 6250 μg $\cdot$ h^{-1} $\cdot$ person^{-1} Tang et al. (2016), t = 5 min and VOC_r = 10000 μg that depends on the effectiveness of the sanitizer.

We set $\phi_{c_{CO_2}}$ to 0 when its value is greater than 0, otherwise it corresponds to the ratio between the difference of CO_2 concentration between two subsequent time-steps and the current CO_2 concentration:

$$\phi_{c_{CO_2}} = \begin{cases} 0 & \phi_{c_{CO_2}} > 0 \\ \frac{c_{CO_2}^{i+t} - c_{CO_2}^i}{c_{CO_2}^i} & otherwise. \end{cases} \tag{A.11}$$

Notice that when $c_{VOC}^{i+1} < c_{VOC}^{i,out}$ we always set the value of $c_{VOC}^{i+1} = c_{VOC}^{i,out}$.

Evolution of Internal Temperature

Depending on ΔT^i, the temperature course can follow a quadratic, linear or constant course. When the action performed by the agent opens the windows, the next value for internal temperature (T_{in}^{i+1} ($°C$)) is computed as a function of the difference between the indoor and the outdoor temperature ($\Delta T^i = T_{out}^i - T_{in}^i$), the heat generated by people in the room (h_p) and by sanitizers in operation (h_s) and the current indoor temperature (T_{in}^i):

$$T_{in}^{i+t} = \begin{cases} j + \frac{h_p + h_s}{4} + T_{in}^i & \Delta T^i > 0 \\ -j + \frac{h_p + h_s}{4} + T_{in}^i & \Delta T^i \leq 0 \end{cases} \tag{A.12}$$

To compute j we distinguish the following cases on the basis of ΔT^i and the thresholds $\varepsilon_2 = 1.6$ and $\varepsilon_3 = 2.5$. The value of j depends on the difference between the indoor and the outdoor temperature (ΔT^i) and the time difference between two subsequent time-steps (t):

$$j = \begin{cases} 0.080 \cdot (\Delta T^i)^2 \cdot t & |\Delta T^i| < \varepsilon_2 \\ \frac{1.084 \cdot |\Delta T^i| - 0.711}{5} \cdot t & \varepsilon_2 \leq |\Delta T^i| \leq \varepsilon_3 \\ 0.4 \cdot t & |\Delta T^i| > \varepsilon_3 \end{cases} \tag{A.13}$$

The value of h_p is computed as a function of the number of people currently in the room (p^i), the time difference between two subsequent time-steps (t) and the room volume (V):

$$h_p = \frac{p^i \cdot t}{5 \cdot V} \tag{A.14}$$

while the value of the heat produced by sanitizers in operation is computed as follows:

$$h_s = \frac{VOC_r \cdot t}{1000 \cdot V} \tag{A.15}$$

where VOC_r is the VOC removed by sanitizers, t is the time difference between two subsequent time-steps and V is the room volume. Notice that we set h_s to zero when the sanitizers are deactivated.

Otherwise, if the action performed by the agent closes the windows, the value of T_{in}^{i+1} depends on the heat generated by people in the room (h_p) and by sanitizers in operation (h_s) and the current value of the internal temperature (T_{in}^i)

$$T_{in}^{i+t} = h_p + h_s + T_{in}^i \tag{A.16}$$

where h_p and h_s are computed in the same way as in Equations (A.14) and (A.15).

Evolution of Room Occupancy and Outdoor Temperature

We assume their values over time are given by the occupation schedule of the room and weather forecast.

A.1.4 Reward Model

The reward model considers four components: air quality r_q, comfort r_w, energy consumption r_E and the energy factor EF (set to 0.1):

$$r = \frac{r_q + r_w + EF \cdot r_E}{2 + EF}. \tag{A.17}$$

The air quality component (r_q) is computed on the base of the presence of persons in the room, and it corresponds to the mean between the reward due to CO_2 concentration ($r_{c_{CO_2}}$) and the reward due to *VOC* concentration ($r_{c_{VOC}}$) when there are occupants, otherwise it is set to 1.

$$r_q = \begin{cases} \frac{r_{c_{CO_2}} + r_{c_{VOC}}}{2} & p^i > 0 \\ 1 & otherwise. \end{cases} \tag{A.18}$$

The values of $r_{c_{CO_2}}$ and $r_{c_{VOC}}$ linearly decrease when the values of c_{CO_2} and c_{VOC} are below their maximum acceptable concentration values (i.e., $\varepsilon_4 = 1000$ ppm and $\varepsilon_6 = 600\ \mu g \cdot m^{-3}$ respectively). Conversely, these rewards quadratically decrease when their values are below the maximum concentration values we suppose the environment could reach (i.e., $\varepsilon_5 = 2500$ ppm and $\varepsilon_7 = 1500\ \mu g \cdot m^{-3}$ respectively), otherwise they are set to 0. More precisely, we compute $r_{c_{CO_2}}$ and $r_{c_{VOC}}$ as follows:

$$r_{c_{CO_2}} = \begin{cases} m_{CO_2} \cdot c^i_{CO_2} + q_{CO_2} & c^i_{CO_2} < \varepsilon_4 \\ x_{CO_2} \cdot (c^i_{CO_2})^2 + y_{CO_2} \cdot c^i_{CO_2} + z_{CO_2} & \varepsilon_4 \leq c^i_{CO_2} < \varepsilon_5 \\ 0 & c^i_{CO_2} \geq \varepsilon_5 \end{cases} \tag{A.19}$$

$$r_{c_{VOC}} = \begin{cases} m_{VOC} \cdot c^i_{VOC} + q_{VOC} & c^i_{VOC} < \varepsilon_6 \\ x_{VOC} \cdot (c^i_{VOC})^2 + y_{VOC} \cdot c^i_{VOC} + z_{VOC} & \varepsilon_6 \leq c^i_{VOC} < \varepsilon_7 \\ 0 & c^i_{VOC} \geq \varepsilon_7 \end{cases} \tag{A.20}$$

with

$$m_{CO_2} = \frac{r_{max_{CO_2}} - r_{min_{CO_2}}}{c^{i,out}_{CO_2} - \varepsilon_4}, \qquad q_{CO_2} = r_{max_{CO_2}} - m_{CO_2} \cdot c^{i,out}_{CO_2}. \tag{A.21}$$

We set $r_{max_{CO_2}} = 1.0$ and $r_{min_{CO_2}} = 0.7$ since we want to find a linear equation that starts at ($c^{i,out}_{CO_2}$, $r_{max_{CO_2}}$) and finishes at (ε_4, $r_{min_{CO_2}}$).

The same method applies to m_{VOC} and q_{VOC} with the two points of the line that are ($c^{i,out}_{VOC}$, $r_{max_{VOC}}$) and (ε_6, $r_{min_{VOC}}$) with $r_{max_{VOC}} = 1$ and $r_{min_{VOC}} = 0.7$.

To model the quadratic course of $r_{c_{CO_2}}$ and $r_{c_{VOC}}$, we assume their curve start respectively at $r_{min_{CO_2}}$ and $r_{min_{VOC}}$ (both set to 0.7) and reach the 0 when $c^{i+1}_{CO_2}$ is equal to ε_5 for CO_2 and when c^{i+1}_{VOC} is equal to ε_6 for *VOC*. In particular,

$$x_{CO_2} = \frac{r_{min_{CO_2}}}{(\varepsilon_5 - \varepsilon_4)^2}, \qquad y_{CO_2} = \frac{-2 \cdot \varepsilon_5 \cdot r_{min_{CO_2}}}{(\varepsilon_5 - \varepsilon_4)^2}, \qquad z_{CO_2} = \frac{(\varepsilon_5)^2 \cdot r_{min_{CO_2}}}{(\varepsilon_5 - \varepsilon_4)^2} \tag{A.22}$$

and

$$x_{VOC} = \frac{r_{min_{VOC}}}{(\varepsilon_7 - \varepsilon_6)^2}, \qquad y_{VOC} = \frac{-2 \cdot \varepsilon_7 \cdot r_{min_{VOC}}}{(\varepsilon_7 - \varepsilon_6)^2}, \qquad z_{VOC} = \frac{(\varepsilon_7)^2 \cdot r_{min_{VOC}}}{(\varepsilon_7 - \varepsilon_6)^2} \tag{A.23}$$

Equations (A.19) and (A.20) change depending on the outdoor concentration of CO_2 ($c^{i,out}_{CO_2}$) and the outdoor concentration of VOC ($c^{i,out}_{VOC}$). In our experiments, we set the value of $c^{i,out}_{CO_2}$ to 400 ppm and of $c^{i,out}_{VOC}$ to 30 $\mu g \cdot m^{-3}$ respectively.

The value of the comfort component (r_w) is computed as a weighted mean between temperature and noise rewards when there are persons in the room, while it is set to 1 where there are no occupants.

$$r_w = \begin{cases} \frac{3 \cdot r_T + r_n}{4} & p^i > 0 \\ 1 & p^i = 0 \end{cases} \tag{A.24}$$

The temperature reward (r_T) is expressed as exponential function of the indoor temperature:

$$r_T = \exp\left(-\left(\frac{T^i_{in} - 20}{5} \right)^2 \right) \tag{A.25}$$

Finally, the noise reward (r_n) and the value of the energy consumption reward (r_E) are constant values associated to the action performed by the agent as we show in Table A.2. The higher the value of these two components (i.e., close to 1) the lower the related energy consumption and the noise pollution.

Table A.2: Energy and noise rewards for each action of the environment.

Action	r_E	r_n
ALL_OFF	1.0	1.0
WINDOW	1.0	1.0
VENT$_L$	0.7	0.8
VENT$_H$	0.1	0.2
SANITIZER	0.9	0.8
WINDOW-SANITIZER	0.9	0.8
VENT$_L$-SANITIZER	0.6	0.6
VENT$_H$-SANITIZER	0.0	0.0

Notice that the reward, and each of its components (i.e., r_q, r_w, and r_E), has a value in the range [0, 1].

A.2 Expert's Model

In this section, we give full details about the expert's model used in Chapter 5 to evaluate the methodology proposed, aiming at model learning and adaptation in MCTS. The model introduces inaccurate transition functions in the true model described in Section A.1 representing the partial knowledge of the expert about the real-world environment. In particular, the model used for the *evolution of* CO_2 *concentration* always overestimates the CO_2 decreasing and, on the other hand, always underestimates its increase. The expert's model for the *evolution of* VOC *concentration* considers the effect of the action performed in terms of ACH, unlike the true-environment model, but it underestimates the concentration of *VOC* that each person emits in the room. Moreover, it does not take into account the relative variation of CO_2 concentration. Finally, to model the *evolution of internal temperature* the expert's model does not consider the heat produced by people in the room and by the sanitizers in operation. Notice that all the other elements of the MDP are the same of the True Environment Model (see Section A.1).

A.2.1 Transition Model

The model consists of three *sub-models* used to describe the evolution of CO_2 concentration, *VOC* concentration, and indoor temperature, respectively.

Evolution of CO_2 Concentration

It considers both *stack ventilation system*, characterized by actions with $\delta_a <$ 9.97 h^{-1}, and *mechanical ventilation systems*, related to actions with $\delta_a \geq$ 9.97 h^{-1}.

- *Stack ventilation system in populated room.* We compute the value of $c_{CO_2}^{i+1}$ as a function of the current value of CO_2 concentration (CO_2^i), the number of people currently in the room (p^i) and the time difference between two subsequent time-steps (t):

$$c_{CO_2}^{i+1} = c_{CO_2}^{i} + \frac{p^i}{150 \cdot \delta_a} \cdot t \tag{A.26}$$

- *Stack ventilation system in unpopulated room.* It could be either *active* (actions with $\delta_a >$ 0.10) or *inactive* (actions with $\delta_a =$ 0.10 h^{-1}). In the *active* case we also consider the current CO_2 concentration (CO_2^i), the ACH (δ_a), and the room volume (V)

$$c_{CO_2}^{i+1} = c_{CO_2}^{i} - \frac{20000 \cdot \delta_a}{V} \tag{A.27}$$

  otherwise (inactive case), the value of $c_{CO_2}^{i+1}$ depends on the current value of CO_2 concentration:

$$c_{CO_2}^{i+1} = c_{CO_2}^{i} - 100 \tag{A.28}$$

- *Mechanical ventilation system.* With mechanical ventilation compatible actions we compute $c_{CO_2}^{i+1}$ as a function of the current CO_2 concentration (CO_2^i), the ACH (δ_a), and the room volume (V):

$$c_{CO_2}^{i+1} = c_{CO_2}^{i} - \frac{30000 \cdot \delta_a}{V} \tag{A.29}$$

Notice that when $c_{CO_2}^{i+1} < c_{CO_2}^{i,out}$ we always set the value of $c_{CO_2}^{i+1} = c_{CO_2}^{i,out}$.

Evolution of VOC Concentration

To approximate the next value of *VOC* concentration (c_{VOC}^{i+1}) we use Equation (A.9) without considering the relative variation of CO_2 concentration ($\phi_{c_{CO_2}}$). Then, the value of c_{VOC}^{i+1} depends on the concentration of *VOC* that each person emits in the room (VOC_p), and the *VOC* removed by the sanitizer (VOC_r^i), the number of persons (p^i) present in the room at time i, the room volume (V) and the *VOC* concentration (c_{VOC}^i) at time i.

$$c_{VOC}^{i+1} = \frac{VOC_p \cdot p^i - VOC_r}{V} + c_{VOC}^{i} \tag{A.30}$$

where

$$VOC_p = 20 \cdot t \tag{A.31}$$

and

$$VOC_r = \begin{cases} \frac{20000 \cdot \delta_a}{7} & \delta_a > 0.1 \wedge \text{sanitizer on} \\ \frac{10000 \cdot \delta_a}{7} & \delta_a > 0.1 \wedge \text{sanitizer off} \\ 10000 & \delta_a = 0.1 \wedge \text{sanitizer on} \\ 0 & otherwise. \end{cases} \tag{A.32}$$

In Equation (A.31) t is the time difference between two subsequent time-steps, i.e., 5 minutes, and in Equation (A.32) δ_a corresponds to the ACH associated to each action (see Table A.1).
Notice that when $c_{VOC}^{i+1} < c_{VOC}^{i,out}$ we always set the value of $c_{VOC}^{i+1} = c_{VOC}^{i,out}$.

Evolution of Internal Temperature

The expert's model, to compute the next value of internal temperature (T_{in}^{i+1}) considers the following two cases

- the windows of the room are open: in this case, the course of the temperature depends on the difference between the indoor and the outdoor temperature ($\Delta T^i = T_{out}^i - T_{in}^i$) and the current indoor temperature (T_{in}^i)

$$T_{in}^{i+1} = \Delta T^i / 1.5 + T_{in}^i \tag{A.33}$$

- the windows of the room are closed: the internal temperature is assumed to be constant

$$T_{in}^{i+1} = T_{in}^i \tag{A.34}$$

Appendix B

Database search results

In this Appendix, we report the results of the search performed on the databases used, namely Scopus and Web of Science, refined by using the inclusion and exclusion criteria outlined in Section 6.2, i.e., the initial set of 181 papers. In the following table, for each paper, we indicate the authors, title, source title, and publication year in addition to the database in which a corresponding record is present. More specifically, in the "Database" column, we use the letters "S" and "W" to indicate the presence of the paper on the Scopus and W databases, respectively, while "S, W" indicates the paper's presence on both databases.

Authors	Title	Source Title	Year	Database
Sultanuddin S.J., Vibin R., Rajesh Kumar A., Behera N.R., Pasha M.J., Baseer K.K.	Development of improved reinforcement learning smart charging strategy for electric vehicle fleet	Journal of Energy Storage	2023	S, W
Ajao L.A., Apeh S.T.	Secure edge computing vulnerabilities in smart cities sustainability using petri net and genetic algorithm-based reinforcement learning	Intelligent Systems with Applications	2023	S
Wang J., Sun L.	Robust Dynamic Bus Control: a Distributional Multi-Agent Reinforcement Learning Approach	IEEE Transactions on Intelligent Transportation Systems	2023	S, W
Szoke L., Aradi S., Bécsi T.	Traffic Signal Control with Successor Feature-Based Deep Reinforcement Learning Agent	Electronics (Switzerland)	2023	S, W
Ali M.Y., Alsaeedi A., Shah S.A.A., Yafooz W.M.S., Malik A.W.	Energy Efficient Data Dissemination for Large-Scale Smart Farming Using Reinforcement Learning	Electronics (Switzerland)	2023	S, W
Yao R., Hu Y., Varga L.	Applications of Agent-Based Methods in Multi-Energy Systems—A Systematic Literature Review	Energies	2023	S, W
Kazemeini A., Swei O.	Identifying environmentally sustainable pavement management strategies via deep reinforcement learning	Journal of Cleaner Production	2023	S, W
Emamjomehzadeh O., Kerachian R., Emami-Skardi M.J., Momeni M.	Combining urban metabolism and reinforcement learning concepts for sustainable water resources management: A nexus approach	Journal of Environmental Management	2023	S
Charef N., Ben Mnaouer A., Aloqaily M., Bouachir O., Guizani M.	Artificial intelligence implication on energy sustainability in Internet of Things: A survey	Information Processing and Management	2023	S, W
Naseer F., Khan M.N., Altalbe A.	Telepresence Robot with DRL Assisted Delay Compensation in IoT-Enabled Sustainable Healthcare Environment	Sustainability (Switzerland)	2023	S
Kolat M., Kővári B., Bécsi T., Aradi S.	Multi-Agent Reinforcement Learning for Traffic Signal Control: A Cooperative Approach	Sustainability (Switzerland)	2023	S, W
Sivamayil K., Rajasekar E., Aljafari B., Nikolovski S., Vairavasundaram S., Vairavasundaram I.	A Systematic Study on Reinforcement Learning Based Applications	Energies	2023	S, W

Authors	Title	Source Title	Year	Database
Khalid M., Wang L., Wang K., Aslam N., Pan C., Cao Y.	Deep reinforcement learning-based long-range autonomous valet parking for smart cities	Sustainable Cities and Society	2023	S
Gao Y., Chang D., Chen C.-H.	A digital twin-based approach for optimizing operation energy consumption at automated container terminals	Journal of Cleaner Production	2023	S
Badakhshan S., Jacob R.A., Li B., Zhang J.	Reinforcement Learning for Intentional Islanding in Resilient Power Transmission Systems	2023 IEEE Texas Power and Energy Conference, TPEC 2023	2023	S
Zhang W., Valencia A., Chang N.	Fingerprint Networked Reinforcement Learning via Multiagent Modeling for Improving Decision Making in an Urban Food-Energy-Water Nexus	IEEE Transactions on Systems, Man, and Cybernetics: Systems	2023	S, W
Li C., Bai L., Yao L., Waller S.T., Liu W.	A bibliometric analysis and review on reinforcement learning for transportation applications	Transportmetrica B	2023	S, W
Venkataswamy V., Grigsby J., Grimshaw A., Qi Y.	RARE: Renewable Energy Aware Resource Management in Datacenters	Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)	2023	S, W
Koch L., Picerno M., Badalian K., Lee S.-Y., Andert J.	Automated function development for emission control with deep reinforcement learning	Engineering Applications of Artificial Intelligence	2023	S, W
Huo D., Sari Y.A., Kealey R., Zhang Q.	Reinforcement Learning-Based Fleet Dispatching for Greenhouse Gas Emission Reduction in Open-Pit Mining Operations	Resources, Conservation and Recycling	2023	S
Feng Y., Zhang X., Jia R., Lin F., Lu J., Zheng Z., Li M.	Intelligent Trajectory Design for Mobile Energy Harvesting and Data Transmission	IEEE Internet of Things Journal	2023	S, W
Chen M., Li Y., Zhang X., Liao R., Wang C., Bi X.	Optimization of river environmental management based on reinforcement learning algorithm: a case study of the Yellow River in China	Environmental Science and Pollution Research	2023	S, W
Gu Z., Liu Z., Wang Q., Mao Q., Shuai Z., Ma Z.	Reinforcement Learning-Based Approach for Minimizing Energy Loss of Driving Platoon Decisions	Sensors	2023	W
Daradkeh M.	Lurkers versus Contributors: An Empirical Investigation of Knowledge Contribution Behavior in Open Innovation Communities	Journal of Open Innovation: Technology, Market, and Complexity	2022	S

Authors	Title	Source Title	Year	Database
Jendoubi I., Bouffard F.	Data-driven sustainable distributed energy resources' control based on multi-agent deep reinforcement learning	Sustainable Energy, Grids and Networks	2022	S
Tomin N., Shakirov V., Kurbatsky V., Muzychuk R., Popova E., Sidorov D., Kozlov A., Yang D.	A multi-criteria approach to designing and managing a renewable energy community	Renewable Energy	2022	S, W
Adetunji K.E., Hofsajer I.W., Abu-Mahfouz A.M., Cheng L.	A novel dynamic planning mechanism for allocating electric vehicle charging stations considering distributed generation and electronic units	Energy Reports	2022	S
Li R., Zhang X., Jiang L., Yang Z., Guo W.	An adaptive heuristic algorithm based on reinforcement learning for ship scheduling optimization problem	Ocean and Coastal Management	2022	S, W
Zhang W., Xie M., Scott C., Pan C.	Sparsity-Aware Intelligent Spatiotemporal Data Sensing for Energy Harvesting IoT System	IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems	2022	S, W
Yao L., Leng Z., Jiang J., Ni F.	Large-Scale Maintenance and Rehabilitation Optimization for Multi-Lane Highway Asphalt Pavement: A Reinforcement Learning Approach	IEEE Transactions on Intelligent Transportation Systems	2022	S, W
Adetunji K.E., Hofsajer I.W., Abu-Mahfouz A.M., Cheng L.	An optimization planning framework for allocating multiple distributed energy resources and electric vehicle charging stations in distribution networks	Applied Energy	2022	S, W
Mahmud S., Abbasi A., Chakrabortty R.K., Ryan M.J.	A self-adaptive hyper-heuristic based multi-objective optimisation approach for integrated supply chain scheduling problems	Knowledge-Based Systems	2022	S, W
Musaddiq A., Ali R., Kim S.W., Kim D.-S.	Learning-Based Resource Management for Low-Power and Lossy IoT Networks	IEEE Internet of Things Journal	2022	S
Giri M.K., Majumder S.	Deep Q-learning based optimal resource allocation method for energy harvested cognitive radio networks	Physical Communication	2022	S

Authors	Title	Source Title	Year	Database
Selukar M., Jain P., Kumar T.	Inventory control of multiple perishable goods using deep reinforcement learning for sustainable environment	Sustainable Energy Technologies and Assessments	2022	S, W
Alibabaei K., Gaspar P.D., Assunção E., Alirezazadeh S., Lima T.M., Soares V.N.G.J., Caldeira J.M.L.P.	Comparison of On-Policy Deep Reinforcement Learning A2C with Off-Policy DQN in Irrigation Optimization: A Case Study at a Site in Portugal	Computers	2022	S, W
Raza A., Shah M.A., Khattak H.A., Maple C., Al-Turjman F., Rauf H.T.	Collaborative multi-agents in dynamic industrial internet of things using deep reinforcement learning	Environment, Development and Sustainability	2022	S, W
Shaw R., Howley E., Barrett E.	Applying Reinforcement Learning towards automating energy efficient virtual machine consolidation in cloud data centers	Information Systems	2022	S, W
Oubbati O.S., Atiquzzaman M., Lim H., Rachedi A., Lakas A.	Synchronizing UAV Teams for Timely Data Collection and Energy Transfer by Deep Reinforcement Learning	IEEE Transactions on Vehicular Technology	2022	S, W
Xu G., Guo F.	Sustainability-oriented maintenance management of highway bridge networks based on Q-learning	Sustainable Cities and Society	2022	S
Wang J.-J., Wang L.	A Cooperative Memetic Algorithm with Learning-Based Agent for Energy-Aware Distributed Hybrid Flow-Shop Scheduling	IEEE Transactions on Evolutionary Computation	2022	S, W
Zhang T., Gou Y., Liu J., Yang T., Cui J.-H.	UDARMF: An Underwater Distributed and Adaptive Resource Management Framework	IEEE Internet of Things Journal	2022	S, W
Zhang M., Lu Y., Hu Y., Amaitik N., Xu Y.	Dynamic Scheduling Method for Job-Shop Manufacturing Systems by Deep Reinforcement Learning with Proximal Policy Optimization	Sustainability (Switzerland)	2022	S, W
Ma Y., Kassler A., Ahmed B.S., Krakhmalev P., Thore A., Toyser A., Lindbäck H.	Using Deep Reinforcement Learning for Zero Defect Smart Forging	Advances in Transdisciplinary Engineering	2022	S
Jang J., Yang H.J.	Deep Learning-Aided User Association and Power Control with Renewable Energy Sources	IEEE Transactions on Communications	2022	S, W
Danassis P., Erden Z.D., Faltings B.	Exploiting environmental signals to enable policy correlation in large-scale decentralized systems	Autonomous Agents and Multi-Agent Systems	2022	S

Authors	Title	Source Title	Year	Database
Manchella K., Haliem M., Aggarwal V., Bhargava B.	PassGoodPool: Joint Passengers and Goods Fleet Management with Reinforcement Learning Aided Pricing, Matching, and Route Planning	IEEE Transactions on Intelligent Transportation Systems	2022	S, W
Yk S., Wu J., Song S.	Research on Autonomous Driving Decision Based on Improved Deep Deterministic Policy Algorithm	SAE Technical Papers	2022	S
Neumann M., Palkovits D.S.	Reinforcement Learning Approaches for the Optimization of the Partial Oxidation Reaction of Methane	Industrial and Engineering Chemistry Research	2022	S, W
Luo M., Du B., Klemmer K., Zhu H., Wen H.	Deployment Optimization for Shared e-Mobility Systems with Multi-Agent Deep Neural Search	IEEE Transactions on Intelligent Transportation Systems	2022	S, W
Dey S., Saha S., Singh A.K., McDonald-Maier K.	SmartNoshWaste: Using Blockchain, Machine Learning, Cloud Computing and QR Code to Reduce Food Waste in Decentralized Web 3.0 Enabled Smart Cities	Smart Cities	2022	S
Kim J., Park J., Cho K.	Continuous Autonomous Ship Learning Framework for Human Policies on Simulation	Applied Sciences (Switzerland)	2022	S, W
Wang T., Liu L., Ding T.	Optimized Sustainable Strategy in Aerial Terrestrial IoT Network	Proceedings - 2022 18th International Conference on Mobility, Sensing and Networking, MSN 2022	2022	S, W
Heo S., Mayer P., Magno M.	Predictive Energy-Aware Adaptive Sampling with Deep Reinforcement Learning	ICECS 2022 - 29th IEEE International Conference on Electronics, Circuits and Systems, Proceedings	2022	S, W
Rampini L., Re Cecconi F.	Artificial Intelligence in Construction Asset Management: A Review of Present Status, Challenges and Future Opportunities	Journal of Information Technology in Construction	2022	S, W
Dusparic I.	Reinforcement Learning for Sustainability: Adapting in large-scale heterogeneous dynamic environments	Proceedings - 2022 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion, ACSOS-C 2022	2022	S, W

Authors	Title	Source Title	Year	Database
Amadi K.W., Iyalla I., Radhakrishna P., Al Saba M.T., Waly M.M.	Continuous Dynamic Drill-Off Test Whilst Drilling Using Reinforcement Learning in Autonomous Rotary Drilling System	Society of Petroleum Engineers - ADIPEC 2022	2022	S
Baumgart U., Burger M.	Optimal Control of Traffic Flow Based on Reinforcement Learning	Communications in Computer and Information Science	2022	S
Andersen P.-A., Goodwin M., Granmo O.-C.	CaiRL: A High-Performance Reinforcement Learning Environment Toolkit	IEEE Conference on Computatonal Intelligence and Games, CIG	2022	S
Sabet S., Farooq B.	Green Vehicle Routing Problem: State of the Art and Future Directions	IEEE Access	2022	S, W
Korecki M., Helbing D.	Analytically Guided Reinforcement Learning for Green It and Fluent Traffic	IEEE Access	2022	S
Ounoughi C., Touibi G., Yahia S.B.	EcoLight: Eco-friendly Traffic Signal Control Driven by Urban Noise Prediction	Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)	2022	S, W
Tang Y., Deng X., Yi L., Xia Y., Yang L.T., Tang A.X.	Collaborative Intelligent Confident Information Coverage Node Sleep Scheduling for 6G-Empowered Green IoT	IEEE Transactions on Green Communications and Networking	2022	S, W
Paul S., Chowdhury S.	A Graph-based Reinforcement Learning Framework for Urban Air Mobility Fleet Scheduling	AIAA Aviation 2022 Forum	2022	S
Isufaj R., Sebastia D.A., Piera M.A.	Toward Conflict Resolution with Deep Multi-Agent Reinforcement Learning	Journal of Air Transportation	2022	S
Eriksson K., Ramasamy S., Zhang X., Wang Z., Danielsson F.	Conceptual framework of scheduling applying discrete event simulation as an environment for deep reinforcement learning	Procedia CIRP	2022	S
Zhang W., Liu H., Xiong H., Xu T., Wang F., Xin H., Wu H.	RLCharge: Imitative Multi-Agent Spatiotemporal Reinforcement Learning for Electric Vehicle Charging Station Recommendation	IEEE Transactions on Knowledge and Data Engineering	2022	S, W
Zhang W., Zhang J., Xie M., Liu T., Wang W., Pan C.	M2M-Routing: Environmental Adaptive Multi-agent Reinforcement Learning based Multi-hop Routing Policy for Self-Powered IoT Systems	Proceedings of the 2022 Design, Automation and Test in Europe Conference and Exhibition, DATE 2022	2022	S, W

Authors	Title	Source Title	Year	Database
Mao Z., Fang Z., Li M., Fan Y.	EvadeRL: Evading PDF Malware Classifiers with Deep Reinforcement Learning	Security and Communication Networks	2022	S
Maree C., Omlin C.W.	Balancing Profit, Risk, and Sustainability for Portfolio Management	2022 IEEE Symposium on Computational Intelligence for Financial Engineering and Economics, CIFEr 2022 - Proceedings	2022	S, W
Lee J., Sun Y.G., Sim I., Kim S.H., Kim D.I., Kim J.Y.	Non-Technical Loss Detection Using Deep Reinforcement Learning for Feature Cost Efficiency and Imbalanced Dataset	IEEE Access	2022	S, W
Liu Y., Yang M., Guo Z.	Reinforcement learning based optimal decision making towards product lifecycle sustainability	International Journal of Computer Integrated Manufacturing	2022	S, W
Alanne K., Sierla S.	An overview of machine learning applications for smart buildings	Sustainable Cities and Society	2022	S
Harrold D.J.B., Cao J., Fan Z.	Data-driven battery operation for energy arbitrage using rainbow deep reinforcement learning	Energy	2022	S, W
Liu Q., Sun S., Rong B., Kadoch M.	Intelligent Reflective Surface Based 6G Communications for Sustainable Energy Infrastructure	IEEE Wireless Communications	2021	S, W
Muhammad G., Hossain M.S.	Deep-Reinforcement-Learning-Based Sustainable Energy Distribution for Wireless Communication	IEEE Wireless Communications	2021	S
Grzelczak M., Duch P.	Deep reinforcement learning algorithms for path planning domain in grid-like environment	Applied Sciences (Switzerland)	2021	S
Gao A., Wang Q., Liang W., Ding Z.	Game Combined Multi-Agent Reinforcement Learning Approach for UAV Assisted Offloading	IEEE Transactions on Vehicular Technology	2021	S, W
Atli İ., Ozturk M., Valastro G.C., Asghar M.Z.	Multi-objective uav positioning mechanism for sustainable wireless connectivity in environments with forbidden flying zones	Algorithms	2021	S
Guo L., Li Z., Outbib R.	Reinforcement Learning based Energy Management for Fuel Cell Hybrid Electric Vehicles	IECON Proceedings (Industrial Electronics Conference)	2021	S, W
Kővári B., Szőke L., Bécsi T., Aradi S., Gáspár P.	Traffic signal control via reinforcement learning for reducing global vehicle emission	Sustainability (Switzerland)	2021	S, W

Authors	Title	Source Title	Year	Database
Rangel-Martinez D., Nigam K.D.P., Ricardez-Sandoval L.A.	Machine learning on sustainable energy: A review and outlook on renewable energy systems, catalysis, smart grid and energy storage	Chemical Engineering Research and Design	2021	S, W
Shani P., Chau S., Swei O.	All roads lead to sustainability: Opportunities to reduce the life-cycle cost and global warming impact of U.S. roadways	Resources, Conservation and Recycling	2021	S
Jia R., Zhang X., Feng Y., Wang T., Lu J., Zheng Z., Li M.	Long-Term Energy Collection in Self-Sustainable Sensor Networks: A Deep Q-Learning Approach	IEEE Internet of Things Journal	2021	S, W
Zhao J., Rodriguez M.A., Buyya R.	A Deep Reinforcement Learning Approach to Resource Management in Hybrid Clouds Harnessing Renewable Energy and Task Scheduling	IEEE International Conference on Cloud Computing, CLOUD	2021	S, W
Kathirgamanathan A., Mangina E., Finn D.P.	Development of a Soft Actor Critic deep reinforcement learning approach for harnessing energy flexibility in a Large Office building	Energy and AI	2021	S, W
Mabina P., Mukoma P., Booysen M.J.	Sustainability matchmaking: Linking renewable sources to electric water heating through machine learning	Energy and Buildings	2021	S
Chen K., Wang H., Valverde-Pérez B., Zhai S., Vezzaro L., Wang A.	Optimal control towards sustainable wastewater treatment plants based on multi-agent reinforcement learning	Chemosphere	2021	S, W
Sacco A., Flocco M., Esposito F., Marchetto G.	Supporting Sustainable Virtual Network Mutations with Mystique	IEEE Transactions on Network and Service Management	2021	S
Munir M.S., Tran N.H., Saad W., Hong C.S.	Multi-Agent Meta-Reinforcement Learning for Self-Powered and Sustainable Edge Computing Systems	IEEE Transactions on Network and Service Management	2021	S
Pérez-Pons M.E., Alonso R.S., García O., Marreiros G., Corchado J.M.	Deep q-learning and preference based multi-agent system for sustainable agricultural market	Sensors	2021	S
Zhang X., Manogaran G., Muthu B.	IoT enabled integrated system for green energy into smart cities	Sustainable Energy Technologies and Assessments	2021	S, W
Emami-Skardi M.J., Momenzadeh N., Kerachian R.	Social learning diffusion and influential stakeholders identification in socio-hydrological environments	Journal of Hydrology	2021	S, W

Authors	Title	Source Title	Year	Database
Almalki A.J., Alsofyani M., Alghuried A., Wocjan P., Wang L.	Model-based variational autoencoders with autoregressive flows	Proceedings of the 2021 5th World Conference on Smart Trends in Systems Security and Sustainability, WorldS4 2021	2021	S
Liu B., Han W., Wang E., Ma X., Xiong S., Qiao C., Wang J.	An efficient message dissemination scheme for cooperative drivings via multi-agent hierarchical attention reinforcement learning	Proceedings - International Conference on Distributed Computing Systems	2021	S, W
Ghosh S., De S., Chatterjee S., Portmann M.	Learning-Based Adaptive Sensor Selection Framework for Multi-Sensing WSN	IEEE Sensors Journal	2021	S
Li L., Luo Y., Pu L.	Q-learning Enabled Intelligent Energy Attack in Sustainable Wireless Communication Networks	IEEE International Conference on Communications	2021	S, W
Sacco A., Esposito F., Marchetto G., Montuschi P.	Sustainable Task Offloading in UAV Networks via Multi-Agent Reinforcement Learning	IEEE Transactions on Vehicular Technology	2021	S
Razack A.J., Ajith V., Gupta R.	A Deep Reinforcement Learning Approach to Traffic Signal Control	2021 IEEE Conference on Technologies for Sustainability, SusTech 2021	2021	S
Zhang W., Liu H., Wang F., Xu T., Xin H., Dou D., Xiong H.	Intelligent electric vehicle charging recommendation based on multi-agent reinforcement learning	The Web Conference 2021 - Proceedings of the World Wide Web Conference, WWW 2021	2021	S, W
Park J., Lee J., Kim T., Ahn I., Park J.	Co-evolution of predator-prey ecosystems by reinforcement learning agents	Entropy	2021	S, W
Eyni A., Skardi M.J.E., Kerachian R.	A regret-based behavioral model for shared water resources management: Application of the correlated equilibrium concept	Science of the Total Environment	2021	S, W
Raeisi M., Mahboob A.S.	Intelligent Control of Urban Intersection Traffic Light Based on Reinforcement Learning Algorithm	26th International Computer Conference, Computer Society of Iran, CSICC 2021	2021	S, W
Piovesan N., Lopez-Perez D., Miozzo M., Dini P.	Joint Load Control and Energy Sharing for Renewable Powered Small Base Stations: A Machine Learning Approach	IEEE Transactions on Green Communications and Networking	2021	S

Authors	Title	Source Title	Year	Database
Leng J., Ruan G., Song Y., Liu Q., Fu Y., Ding K., Chen X.	A loosely-coupled deep reinforcement learning approach for order acceptance decision of mass-individualized printed circuit board manufacturing in industry 4.0	Journal of Cleaner Production	2021	S, W
Baumgart U., Burger M.	A Reinforcement Learning Approach for Traffic Control	International Conference on Vehicle Technology and Intelligent Transport Systems, VEHITS - Proceedings	2021	S, W
Dinh T.H.L., Kaneko M., Wakao K., Kawamura K., Moriyama T., Takatori Y.	Towards an Energy-Efficient DQN-based User Association in Sub6GHz/mmWave Integrated Networks	Proceedings - 2021 17th International Conference on Mobility, Sensing and Networking, MSN 2021	2021	S
Barth A., Zhang L., Ma O.	Cooperation of a team of heterogeneous swarm robots for space exploration	Proceedings of the International Astronautical Congress, IAC	2021	S
Al-Jawad A., Comsa I.-S., Shah P., Gemikonakli O., Trestian R.	REDO: A Reinforcement Learning-based Dynamic Routing Algorithm Selection Method for SDN	2021 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN 2021 - Proceedings	2021	S, W
Cloud J.M., Nieves R.J., Duke A.K., Muller T.J., Janmohamed N.A., Buckles B.C., Dupuis M.A.	Towards autonomous lunar resource excavation via deep reinforcement learning	Accelerating Space Commerce, Exploration, and New Discovery conference, ASCEND 2021	2021	S
Serrano J.C., Mula J., Poler R.	Digital Twin for Supply Chain Master Planning in Zero-Defect Manufacturing	IFIP Advances in Information and Communication Technology	2021	S
Chaudhuri R., Mukherjee K., Narayanam R., Vallam R.D.	Collaborative Reinforcement Learning Framework to Model Evolution of Cooperation in Sequential Social Dilemmas	Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)	2021	S, W
Zheng Z., Yan P., Chen Y., Cai J., Zhu F.	Increasing Crop Yield Using Agriculture Sensing Data in Smart Plant Factory	Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)	2021	S

Authors	Title	Source Title	Year	Database
Musaddiq A., Ali R., Choi J.-G., Kim B.-S., Kim S.-W.	Collision observation-based optimization of low-power and lossy IoT network using reinforcement learning	Computers, Materials and Continua	2021	S, W
Ballis H., Dimitriou L.	Evaluating the Performance of Reinforcement Learning Signalling Strategies for Sustainable Urban Road Networks	Advances in Intelligent Systems and Computing	2021	S
Surovik D., Wang K., Vespignani M., Bruce J., Bekris K.E.	Adaptive tensegrity locomotion: Controlling a compliant icosahedron with symmetry-reduced reinforcement learning	International Journal of Robotics Research	2021	S, W
Mandhare P., Yadav J., Kharat V., Patil C. Y.	Control and Coordination of Self-Adaptive Traffic Signal Using Deep Reinforcement Learning	International Journal of Next-Generation Computing	2021	W
Tiwari T., Shastry N., Nandi A.	Deep learning based lateral control system	Proceedings - 2020 IEEE International Symposium on Sustainable Energy, Signal Processing and Cyber Security, iSSSC 2020	2020	S
Chemingui Y., Gastli A., Ellabban O.	Reinforcement learning-based school energy management system	Energies	2020	S, W
Ballis H., Dimitriou L.	Evaluation of Reinforcement Learning Traffic Signalling Strategies for Alternative Objectives: Implementation in the Network of Nicosia, Cyprus	Transport and Telecommunication	2020	S, W
Guliyev H.B., Tomin N.V., Ibrahimov F.S.	Methods of intelligent protection from asymmetrical conditions in electric networks	E3S Web of Conferences	2020	S
Liu H., Zhang C., Guo Q.	Data-driven robust voltage/var control using PV inverters in active distribution networks	Proceedings - 2020 International Conference on Smart Grids and Energy Systems, SGES 2020	2020	S, W
Nakamoto Y., Kumalija E., Zhang M.	Toward autonomous adaptive embedded systems for sustainable services using reinforcement learning (WiP report)	Proceedings - 2020 8th International Symposium on Computing and Networking Workshops, CANDARW 2020	2020	S
Han M., Del Castillo L.A., Khairy S., Chen X., Cai L.X., Lin B., Hou F.	Multi-agent Reinforcement Learning for Green Energy Powered IoT Networks with Random Access	IEEE Vehicular Technology Conference	2020	S, W

Authors	Title	Source Title	Year	Database
Tan Z., Karakose M.	Comparative Study for Deep Reinforcement Learning with CNN, RNN, and LSTM in Autonomous Navigation	2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy, ICDABI 2020	2020	S
Lee S., Cho Y., Lee Y.H.	Injection mold production sustainable scheduling using deep reinforcement learning	Sustainability (Switzerland)	2020	S, W
Han M., Duan J., Khairy S., Cai L.X.	Enabling Sustainable Underwater IoT Networks with Energy Harvesting: A Decentralized Reinforcement Learning Approach	IEEE Internet of Things Journal	2020	S, W
Opalic S.M., Goodwin M., Jiao L., Nielsen H.K., Lal Kolhe M.	A Deep Reinforcement Learning scheme for Battery Energy Management	2020 5th International Conference on Smart and Sustainable Technologies, SpliTech 2020	2020	S
Dawn S., Saraogi U., Thakur U.S.	Agent-based Learning for Auto-Navigation within the Virtual City	2020 International Conference on Computational Performance Evaluation, ComPE 2020	2020	S, W
Xi F., Ruan X.	Influence of Intelligent Environmental Art based on Reinforcement Learning on the Regionality of Architectural Design	Proceedings of the International Conference on Electronics and Sustainable Communication Systems, ICESC 2020	2020	S
Skardi M.J.E., Kerachian R., Abdolhay A.	Water and treated wastewater allocation in urban areas considering social attachments	Journal of Hydrology	2020	S, W
Banerjee P.S., Mandal S.N., De D., Maiti B.	RL-Sleep: Temperature Adaptive Sleep Scheduling using Reinforcement Learning for Sustainable Connectivity in Wireless Sensor Networks	Sustainable Computing: Informatics and Systems	2020	S, W
Piovesan N., Miozzo M., Dini P.	Modeling the environment in deep reinforcement learning: The case of energy harvesting base stations	ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings	2020	S, W
Radenkovic M., Ha Huynh V.S.	Energy-Aware Opportunistic Charging and Energy Distribution for Sustainable Vehicular Edge and Fog Networks	2020 5th International Conference on Fog and Mobile Edge Computing, FMEC 2020	2020	S, W

Authors	Title	Source Title	Year	Database
Ma D., Lan G., Hassan M., Hu W., Das S.K.	Sensing, Computing, and Communications for Energy Harvesting IoTs: A Survey	IEEE Communications Surveys and Tutorials	2020	S, W
Miozzo M., Piovesan N., Dini P.	Coordinated Load Control of Renewable Powered Small Base Stations Through Layered Learning	IEEE Transactions on Green Communications and Networking	2020	S
Bouhamed O., Ghazzai H., Besbes H., Massoud Y.	A UAV-Assisted Data Collection for Wireless Sensor Networks: Autonomous Navigation and Scheduling	IEEE Access	2020	S, W
Elavarasan D., Durairaj Vincent P.M.	Crop Yield Prediction Using Deep Reinforcement Learning Model for Sustainable Agrarian Applications	IEEE Access	2020	S
Yang T., Zhao L., Li W., Zomaya A.Y.	Reinforcement learning in sustainable energy and electric systems: a survey	Annual Reviews in Control	2020	S, W
Temesgene D.A., Miozzo M., Dini P.	Dynamic control of functional splits for energy harvesting virtual small cells: A distributed reinforcement learning approach	Computer Communications	2019	S, W
Lin K., Lin B., Chen X., Lu Y., Huang Z., Mo Y.	A time-driven workflow scheduling strategy for reasoning tasks of autonomous driving in edge environment	Proceedings - 2019 IEEE Intl Conf on Parallel and Distributed Processing with Applications, Big Data and Cloud Computing, Sustainable Computing and Communications, Social Computing and Networking, ISPA/BDCloud/SustainCom/SocialCom 2019	2019	S
Bhargavi K., Sathish Babu B.	Load Balancing Scheme for the Public Cloud using Reinforcement Learning with Raven Roosting Optimization Policy (RROP)	CSITSS 2019 - 2019 4th International Conference on Computational Systems and Information Technology for Sustainable Solution, Proceedings	2019	S
Strnad F.M., Barfuss W., Donges J.F., Heitzig J.	Deep reinforcement learning in World-Earth system models to discover sustainable management strategies	Chaos	2019	S, W

Authors	Title	Source Title	Year	Database
Firdausiyah N., Taniguchi E., Qureshi A.G.	Impacts of Urban Consolidation Centres for Sustainable City Logistics Using Adaptive Dynamic Programming Based Multi-Agent Simulation	IOP Conference Series: Earth and Environmental Science	2019	S
Xu T., Wang N., Lin H., Sun Z.	UAV Autonomous Reconnaissance Route Planning Based on Deep Reinforcement Learning	Proceedings of the 2019 IEEE International Conference on Unmanned Systems, ICUS 2019	2019	S
Alizadeh Shabestray S.M., Abdulhai B.	Multimodal iNtelligent Deep (MiND) Traffic Signal Controller	2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019	2019	S, W
Ebell N., Gütlein M., Pruckner M.	Sharing of Energy among Cooperative Households Using Distributed Multi-Agent Reinforcement Learning	Proceedings of 2019 IEEE PES Innovative Smart Grid Technologies Europe, ISGT-Europe 2019	2019	S, W
Vo N.N.Y., He X., Liu S., Xu G.	Deep learning for decision making and the optimization of socially responsible investments and portfolio	Decision Support Systems	2019	S, W
Chang S., Saha N., Castro-Lacouture D., Yang P.P.-J.	Multivariate relationships between campus design parameters and energy performance using reinforcement learning and parametric modeling	Applied Energy	2019	S, W
Shabana Anjum S., Md Noor R., Ahmedy I., Anisi M.H.	Energy optimization of sustainable Internet of Things (IoT) systems using an energy harvesting medium access protocol	IOP Conference Series: Earth and Environmental Science	2019	S
Do Q.V., Koo I.	Dynamic Bandwidth Allocation Scheme for Wireless Networks with Energy Harvesting Using Actor-Critic Deep Reinforcement Learning	1st International Conference on Artificial Intelligence in Information and Communication, ICAIIC 2019	2019	S, W
Blad C., Koch S., Ganeswarathas S., Kallesøe C.S., Bøgh S.	Control of HVAC-systems with slow thermodynamic using reinforcement learning	Procedia Manufacturing	2019	S
Mikhail M., Yacout S., Ouali M.-S.	Optimal preventive maintenance strategy using reinforcement learning	Proceedings of the International Conference on Industrial Engineering and Operations Management	2019	S
Chen H., Zhao T., Li C., Guo Y.	Green Internet of Vehicles: Architecture, Enabling Technologies, and Applications	IEEE Access	2019	S, W

Authors	Title	Source Title	Year	Database
Chaudhuri R., Vallam R.D., Garg S., Mukherjee K., Kumar A., Singh S., Narayanam R., Mathur A., Parija G.	Collaborative reinforcement learning model for sustainability of cooperation in sequential social dilemmas	Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS	2019	S, W
Park J.Y., Nagy Z.	The Influence of Building Design, Sensor Placement, and Occupant Preferences on Occupant Centered Lighting Control	Computing in Civil Engineering 2019: Smart Cities, Sustainability, and Resilience - Selected Papers from the ASCE International Conference on Computing in Civil Engineering 2019	2019	S
Saifuddin M. R. B., Logenthiran T., Naayagi R. T., Woo W. L.	A Nano-Biased Energy Management Using Reinforced Learning Multi-Agent on Layered Coalition Model: Consumer Sovereignty	IEEE Access	2019	W
Temesgene D.A., Miozzo M., Dini P.	Dynamic Functional Split Selection in Energy Harvesting Virtual Small Cells Using Temporal Difference Learning	IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC	2018	S, W
Prauzek M., Mourcet N.R.A., Hlavica J., Musilek P.	Q-Learning Algorithm for Energy Management in Solar Powered Embedded Monitoring Systems	2018 IEEE Congress on Evolutionary Computation, CEC 2018 - Proceedings	2018	S, W
Ghanshala K.K., Sharma S., Mohan S., Nautiyal L., Mishra P., Joshi R.C.	Self-Organizing Sustainable Spectrum Management Methodology in Cognitive Radio Vehicular Adhoc Network (CRAVENET) Environment: A Reinforcement Learning Approach	ICSCCC 2018 - 1st International Conference on Secure Cyber Computing and Communications	2018	S, W
Aziz H.M.A., Zhu F., Ukkusuri S.V.	Learning-based traffic signal control algorithms with neighborhood information sharing: An application for sustainable mobility	Journal of Intelligent Transportation Systems: Technology, Planning, and Operations	2018	S
Ganapathi Subramanian S., Crowley M.	Combining MCTS and A3C for prediction of spatially spreading processes in forest wildfire settings	Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)	2018	S

Authors	Title	Source Title	Year	Database
Miozzo M., Giupponi L., Rossi M., Dini P.	Switch-On/Off Policies for Energy Harvesting Small Cells through Distributed Q-Learning	2017 IEEE Wireless Communications and Networking Conference Workshops, WCNCW 2017	2017	S, W
Lindkvist E., Ekeberg Ö., Norberg J.	Strategies for sustainable management of renewable resources during environmental change	Proceedings of the Royal Society B: Biological Sciences	2017	S
Perolat J., Leibo J.Z., Zambaldi V., Beattie C., Tuyls K., Graepel T.	A multi-agent reinforcement learning model of common-pool resource appropriation	Advances in Neural Information Processing Systems	2017	S, W
Sheikhi A., Rayati M., Ranjbar A.M.	Dynamic load management for a residential customer; Reinforcement Learning approach	Sustainable Cities and Society	2016	S, W
Chen H., Li X., Zhao F.	A reinforcement learning-based sleep scheduling algorithm for desired area coverage in solar-powered wireless sensor networks	IEEE Sensors Journal	2016	S, W
De Gracia A., Fernández C., Castell A., Mateu C., Cabeza L.F.	Control of a PCM ventilated facade using reinforcement learning techniques	Energy and Buildings	2015	S
Soares I.B., De Hauwere Y.-M., Januarius K., Brys T., Salvant T., Nowe A.	Departure MANagement with a Reinforcement Learning Approach: Respecting CFMU Slots	IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC	2015	S
Jin J., Ma X.	Adaptive Group-Based Signal Control Using Reinforcement Learning with Eligibility Traces	IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC	2015	S
Hsu R.C., Lin T.-H., Chen S.-M., Liu C.-T.	Dynamic energy management of energy harvesting wireless sensor nodes using fuzzy inference system with reinforcement learning	Proceeding - 2015 IEEE International Conference on Industrial Informatics, INDIN 2015	2015	S, W
Miozzo M., Giupponi L., Rossi M., Dini P.	Distributed Q-learning for energy harvesting Heterogeneous Networks	2015 IEEE International Conference on Communication Workshop, ICCW 2015	2015	S
Comşa I.S., Aydin M., Zhang S., Kuonen P., Wagen J.-F., Lu Y.	Scheduling policies based on dynamic throughput and fairness tradeoff control in LTE-A networks	Proceedings - Conference on Local Computer Networks, LCN	2014	S, W
Hsu R.C., Liu C.-T., Wang H.-L.	A reinforcement learning-based ToD provisioning dynamic power management for sustainable operation of energy harvesting wireless sensor node	IEEE Transactions on Emerging Topics in Computing	2014	S

Authors	Title	Source Title	Year	Database
Urieli D., Stone P.	TacTex'13: A champion adaptive power trading agent	13th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2014	2014	S
Lindkvist E., Norberg J.	Modeling experiential learning: The challenges posed by threshold dynamics for sustainable renewable resource management	Ecological Economics	2014	S
Crowley M.	Using equilibrium policy gradients for spatiotemporal planning in forest ecosystem management	IEEE Transactions on Computers	2014	S
Bielskis A.A., Guseinoviene E., Zutautas L., Drungilas D., Dzemydiene D., Gricius G.	Modeling of Ambient Comfort Affect Reward based on multi-agents in cloud interconnection environment for developing the sustainable home controller	2013 8th International Conference and Exhibition on Ecological Vehicles and Renewable Energies, EVER 2013	2013	S
Bielskis A.A., Guseinoviene E., Dzemydiene D., Drungilas D., Gricius G.	Ambient lighting controller based on reinforcement learning components of multi-agents	Elektronika ir Elektrotechnika	2012	S, W
Sabbadin R., Spring D., Bergonnier E.	A reinforcement-learning application to biodiversity conservation in Costa-Rican forest	MODSIM07 - Land, Water and Environmental Management: Integrated Systems for Sustainability, Proceedings	2007	S
Chadès I., Martin T.G., Curtis J.M.R., Barreto C.	Managing interacting species: A reinforcement learning decision theoretic approach	MODSIM07 - Land, Water and Environmental Management: Integrated Systems for Sustainability, Proceedings	2007	S

Appendix C

Application Domains of Environmental Sustainability

In this Appendix, we provide full details about the keywords used to answer research question RQ4 in Section 6.3.1 about the application domain of environmental sustainability issues. To answer this research question, we analyze the index keywords of the 181 papers that were not excluded by applying the inclusion and exclusion criteria, as well as the authors' keywords for works with no index keywords. We group the keywords into macro areas, e.g. in the macro area "Energy" we include keywords such as "Energy", "Energy Conservation", "Energy Consumption", etc. In the following, we represent each macro area with a table, in which we report all keywords belonging to this macro area.

Table C.1: Keywords grouped in the "Energy" macro area

Energy		
energy	energy aware	energy conservation
energy consumption	energy distributions	energy efficiency
energy harvesting	energy management	energy management systems
energy resource	energy source	energy storage
energy sustainability	energy systems	energy utilization
distributed energies	dynamic energy	energy allocations
energy arbitrages	energy availability	energy consumption balances
energy flexibility	energy infrastructures	energy management strategy
energy market	energy neutrality	energy optimal scheduling
energy routing	energy savings	energy storage system
energy theft	energy transfer	energy usage
energy use	energy-awareness	energy-constrained networks
energy-saving strategies	harvesting energies	intelligent energy management
non-renewable energy	total energy consumption	wireless energy transfers
energy trading		

Table C.2: Keywords grouped in the "Electric energy" macro area

Electric energy	
electric energy storage	electric load dispatching
electric power transmission networks	electric power utilization
dynamic energy managements	dynamic loads
dynamic power management	electric load management
electric loads	electric power system control
electric power transmission	electrical networks
electricity demands	electricity loss
micro grid	smart grid
smart power grids	electricity grids
grid resilience	power grids
electricity grids	smart grid communications
grid resilience	distributed power generation
power management	power supply
low power electronics	dynamic power management
electric power system control	electric power transmission
inductive power transmission	low power
low power and lossy network (lln)	low power networks
low-power consumption	low-power devices
power	power allocations
power dispatch	power grids
power harvesting	power limitations
power system simulator for engineering	power system simulators
power traces	power control
power transmission systems	reactive power
reactive power output	power management (telecommunication)

Table C.3: Keywords grouped in the "Urban Traffic & Transportation" macro area

Urban Traffic & Transportation		
street traffic control	sustainable mobility	traffic congestion
traffic emission	traffic flow	traffic light control
traffic management	traffic signal control	traffic signals
adaptive traffic signal control	intelligent traffic controls	optimal traffic control
traffic	traffic conditions	traffic environment
traffic light	traffic management strategies	traffic scheduling
urban traffic	highway administration	motor transportation
transportation	transportation system	urban transportation
bus bunching	bus transportation	sustainable transportation
intelligent transportation	transport systems	transportation network
transportation planning		

Table C.4: Keywords grouped in the "Water resources" macro area

Water resources		
aquifer	ground water	wastewater
wastewater treatment	water management	water quality
water resources	water resources management	water supply
water treatment	surface water	waste water management
waste water recycling	sustainable wastewater treatments	wastewater treatment plant
water metabolism	water purification	water resources systems
water treatment plants	groundwater resources	water level
water quantity	watersheds	

Table C.5: Keywords grouped in the "Renewable/sustainable energies" macro area

Renewable/sustainable energies		
renewable energies	renewable energy resources	renewable energy source
renewable resource	renewables	alternative energy
solar energy	green energy	smart renewable energy
use of renewable energies	sustainable energy	solar power generation
renewable power generation	tidal power	wind power
hydroelectric power plants	hydropower	

Table C.6: Keywords grouped in the "Emissions/Pollution" macro area

Emissions/Pollution		
carbon emission	carbon footprint	emission control
gas emissions	greenhouse gas	greenhouse gas emissions
acoustic noise	air pollution monitoring	atmospheric pollution
carbon abatement strategy	carbon sequestration	co2 emissions
greenhouse emissions	greenhouse gas emission reduction	groundwater pollution
noise pollution	pollution control	vehicular emission
water pollution	air quality	

Table C.7: Keywords grouped in the "Mobile & Wireless communication" macro area

Mobile & Wireless communication	
5G mobile communication systems	mobile telecommunication systems
6g mobile communication	mobile communications
base stations	small cells
wireless communication links	wireless communications
wireless telecommunication systems	sustainable wireless communication network
wireless communications networks	communication network
heterogeneous networks	wireless powered communication network

Table C.8: Keywords grouped in the "Data" macro area

Data		
data acquisition	data handling	data transfer
digital storage	big data	data aggregation
data aggregation and fusion	data analytics	data distribution
data logger	data mining	data sensing
data-driven approach	data-driven design	database technology
datacenter	distributed database	next generation data centers
data transfer	data-communication	real-time data

Table C.9: Keywords grouped in the "Wireless sensor network" macro area

Wireless sensor network	
wireless sensor network	wireless sensor node
heterogeneous wireless sensor networks	solar-powered wireless sensor networks
rechargeable sensor networks	wireless smart sensors
sensor nodes	integrating sensors
sensor networks	wireless smart sensors
sensor networks	smart sensors
adaptive sensor selections	mobile sensors
sensor	sensor payloads
sleep scheduling	

Table C.10: Keywords grouped in the "Autonomous vehicles" macro area

Autonomous vehicles	
autonomous driving	autonomous navigation
autonomous vehicles	unmanned aerial vehicles (uav)
autonomous unmanned aerial vehicles	uav networks
unmanned aerial vehicle	uav positioning
autonomous vehicle control	autonomous ship
auto-navigation	automated vehicles

Table C.11: Keywords grouped in the "Batteries" macro area

Batteries		
battery energy storage systems	battery management systems	battery storage
charging (batteries)	secondary batteries	battery operation
electric batteries	battery capacity	lead acid batteries
lithium batteries	residual battery	

Table C.12: Keywords grouped in the "Manufacturing" macro area

Manufacturing	
manufacture	manufacturing
production control	supply chains
sustainable manufacturing	distributed manufacturing systems
manufacturing environments	large scale manufacturing systems
manufacturing industries	manufacturing sector
printed circuit board manufacturing	process manufacturing
sustainable manufacturing engineering and resource-efficient production	

Table C.13: Keywords grouped in the "Electric vehicles" macro area

Electric vehicles	
charging station	electric vehicle charging
electric vehicle charging station	electric vehicles
electric vehicle charging station recommendation	e mobilities
fuel cell hybrid electric vehicles	vehicle-to-grid

Table C.14: Keywords grouped in the "IoT" macro area

IoT		
internet of things	internet of underwater things	green internet of thing
industrial internet of thing		

Table C.15: Keywords grouped in the "Agriculture" macro area

Agriculture		
agricultural robots	agriculture	crops
agricultural productions	agricultural products	crop productivity
smart agricultures	sustainable agricultural	sustainable agricultural system
sustainable agriculture		

Table C.16: Keywords grouped in the "Vehicles" macro area

Vehicles			
intelligent vehicle highway systems	vehicles	transit vehicles	vehicle dispatch
vehicle fleets			

Table C.17: Keywords grouped in the "Buildings" macro area

Buildings		
building	building coverage ratios	building energy
building energy flexibility	building energy management systems	building stocks
college buildings	building energy managements	intelligent buildings
office buildings		