Journal of Artificial Intelligence Research 82 (2025) 2325-2388 Submitted 07/2024; published 04/2025

Composition and Zero-Shot Transfer with Lattice Structures in Reinforcement Learning

Geraud Nangue Tasse GERAUD.NANGUETASSE1@WITS.AC.ZA
Steven James STEVEN.JAMES@WITS.AC.ZA
Benjamin Rosman BENJAMIN.ROSMAN1@WITS.AC.ZA
University of the Witwatersrand
Machine Intelligence and Neural Discovery (MIND) Institute
1 Jan Smuts Avenue, Johannesburg, 2000

Abstract

An important property of long-lived agents is the ability to reuse existing knowledge to solve new tasks. An appealing approach towards obtaining such agents is by leveraging logical composition over tasks, where new tasks are defined by applying logic operators to previously-solved ones. This composition is particularly powerful since it provides a human-understandable mechanism for task specification. However, no unifying formalism for applying logic operators to tasks and generalising combinatorially over them has yet been developed. We address the problem by formally defining logical composition as operators acting on a set of tasks in a lattice structure—the algebraic structure that generalises the study of Boolean logic. This provides a theoretically rigorous method for composing tasks, allowing us to formulate new tasks in terms of the negation, disjunction, and conjunction of a set of base tasks. We prove that by learning a new type of goal-oriented value function model free, called the *world value function*, an agent can solve composite tasks involving arbitrary logical operators with no further learning. We verify our approach in high-dimensional domains—including a video game environment and continuous-control task—where an agent first learns to solve a set of base tasks, and then composes these solutions to solve a super-exponential number of new tasks.

1. Introduction

Reinforcement learning (RL) has achieved recent success in a number of difficult, high-dimensional environments [Mnih et al., 2015; Levine et al., 2016; Lillicrap et al., 2016; Silver et al., 2017]. However, these methods generally require millions of samples from the environment to learn optimal behaviours, limiting their real-world applicability. A major challenge is thus in designing sample-efficient agents that can transfer their existing knowledge to solve new tasks quickly. This is particularly important for agents in a multitask or lifelong setting, since learning to solve complex tasks from scratch is typically infeasible. More desirable would be an agent that can leverage its knowledge from previous tasks to solve new ones [Thrun, 1996; Taylor & Stone, 2009].

One approach to tackling this problem is that of *composition* [Todorov, 2009; Mendez & Eaton, 2022], where an agent leverages existing skills to build complex, novel behaviours. These newly-formed skills can then be used to immediately solve new tasks (zero-shot composition) or speed up the learning of new tasks (few-shot composition). Zero-shot composition is of particular interest, since it provides a mechanism for generalising over a

©2025 The Authors. Published by AI Access Foundation under Creative Commons Attribution License CC BY 4.0.

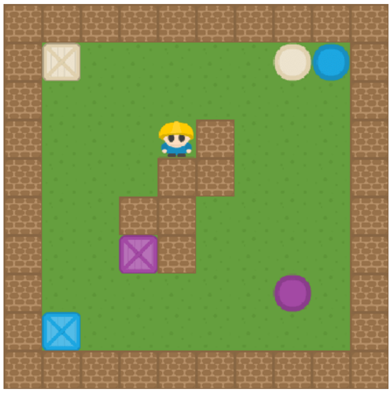


Figure 1: Video game domain with different composable tasks [van Niekerk et al., 2019]. Each task is defined by the objects the agent is required to collect.

task space without additional learning. A particularly important type of composition is the logical composition of previously solved tasks—that is, new tasks formed by applying the logic operators `OR`, `AND`, and `NOT` on past tasks. One major advantage of this form of composition is that it provides a human designer with an easy way to understand and reason about tasks, which is particularly important for explainable and interpretable reinforcement learning [Heuillet et al., 2021; Qing et al., 2022; Glanois et al., 2024]. Furthermore, while it can be difficult to specify a reward function that induces desired behaviour in an agent, logical composition provides a straightforward avenue for specifying new composite tasks with guaranteed semantics.

To illustrate, consider a domain where different tasks require an agent to collect objects of different shapes and colours (Figure 1). If an agent has learned to collect any square (*Square* task) and any blue object (*Blue* task), zero-shot logical composition would then allow the agent to solve any of the following tasks immediately:

- *Square* `AND` *Blue*: Collect blue squares.
- *Square* `AND` (`NOT` *Blue*): Collect squares that are not blue.
- (*Square* `OR` *Blue*) `AND` `NOT`(*Square* `AND` *Blue*): Collect squares or blue objects that are not blue squares.

In this work, we are hence interested in the logical combination of tasks and skills in the model-free setting—this constraint will ensure our framework is applicable even when the reward and transition functions are not known to the agent. Since logic are abstracted in lattice theory [Birkhoff, 1940], we use the algebraic structures therein to formalise the logical composition of tasks and skills in reinforcement learning. We first define a lattice algebra over the set of tasks. This formalises and unifies the disjunctive and conjunctive compositions considered by previous work [Todorov, 2009; Saxe et al., 2017; Haarnoja et al., 2018a; van Niekerk et al., 2019; Hunt et al., 2019]. We then give a formal meaning to the negation of tasks, which is used to define a De Morgan algebra over the task space. This extends previous composition problems to encompass all logic operators: conjunction, disjunction, and negation. We show that for tasks with restricted rewards, the De Morgan algebra extends to a Boolean algebra, which gives task composition the full semantics of Boolean logic.

To attain zero-shot composition, we introduce a new type of value function called the *world value function*, which can be learned model-free and encodes how to achieve various outcomes for a single task. Using this rich knowledge, and focusing our attention on goal-reaching tasks, we prove that zero-shot logical composition can be achieved by composing these world value functions. Consequently, given a set of base tasks that have been previously solved by the agent, any new task written as a logic expression can immediately be solved without further learning, resulting in a zero-shot super-exponential explosion in the agent's abilities. We summarise our main contributions as follows:

1. *Task algebra (Section 3):* We formalise the disjunction, conjunction, and negation of tasks as operators acting on a set of tasks in lattice structures. This introduces a notion of task space, and enables interpretable and trivial task specifications. This is important in complex lifelong settings, where constructing rewards for new tasks can prove difficult. A main result obtained from this formalism is the notion of *base tasks*, which are the minimal set of tasks that are sufficient to specify and generate any other task in a task space. This is essential in the lifelong setting where the task space is simply too large to construct rewards for all desired tasks.

2. *World value functions (Section 4):* This is a new type of goal-conditioned value function that we introduce to later achieve zero-shot composition. It can be learned model-free and encodes knowledge of how to achieve various goals in an environment, and is therefore suitable for transferring to new tasks within the same environment.

3. *Zero-shot composition (Section 5):* We prove that any goal-reaching task specified as the logical composition of learned tasks can be solved immediately without further learning. We demonstrate via a series of experiments that this holds not only theoretically but also practically, even when the theoretical assumptions are violated. This enables lifelong learning agents to solve a super-exponentially increasing number of tasks as the number of learned base tasks increase.

We illustrate our approach in gridworld domains, where an agent first learns to navigate to particular regions, after which it can then optimally solve any task specified as their logical combination. We then demonstrate composition in a high-dimensional video game environment, where an agent first learns to collect different objects, and then compose these abilities to solve complex tasks immediately. We also apply our approach to a high-dimensional, continuous control task, demonstrating applicability to domains with low-level continuous actions. Our results show that, even when function approximation is required, an agent can leverage its existing skills to solve new tasks without further learning.[1]

The remaining sections are organised as follows: we provide a brief introduction to reinforcement learning and lattice theory in Section 2. In Section 3, we develop a formal framework for the logical composition of tasks, and introduce the notion of a base task,

[1] This article develops and substantially extends ideas first presented in an earlier conference paper [Nangue Tasse et al., 2020] by introducing the notion of task lattices and De Morgan algebra, proving zero-shot composition for them, and demonstrating applicability to high-dimensional continuous control tasks.

which has been so far mentioned only informally in the literature. We present a new goal-conditioned value function in Section 4 which can be leveraged to solve composite tasks in a zero-shot manner. Finally, in Section 5, we demonstrate zero-shot transfer in both the tabular and function approximation setting.

2. Preliminaries

We are interested in the multitask or lifelong learning problem, where an agent is faced with a number of tasks modeled by a collection of Markov Decision Processes (MDPs) [Sutton et al., 1998]. An MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where (i) $\mathcal{S}$ is the agent's state space, (ii) $\mathcal{A}$ is its action space, (iii) $P(s'|s,a)$ is the Markov transition probability of its environment, (iv) $R(s,a,s')$ is the real-valued reward function defining the agent's task, and (v) $\gamma \in [0,1]$ is the agent's discount factor. For clarity, we will focus on MDPs with finite $\mathcal{S}$ and $\mathcal{A}$, but note that the theory also hold for continuous $\mathcal{S}$ and $\mathcal{A}$. The set of tasks is then naturally defined as follows:

Definition 1 *A set of tasks $\mathcal{M}$ is a set of MDPs $M \in \mathcal{M}$ in the same environment $(\mathcal{S}, \mathcal{A}, P, \gamma)$ that differ only in their reward functions R_M.*

An agent's goal is to learn a *policy* π that specifies the actions to take in each state to maximise the action-value function Q, which is the total sum of possibly discounted rewards it expects to receive in the future: $Q^{\pi}(s,a) := \mathbb{E}_s^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \middle| s_0 = s, a_0 = a\right]$. By defining a partial ordering over policies, and under mild assumptions over the reward functions,[2] it can be shown that there exists an optimal deterministic policy $\pi^* \geq \pi, \forall \pi$, which leads to the optimal value functions, $Q^{\pi^*}(s,a) = Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a) \;\; \forall \pi^*$. We can obtain the optimal policy by acting greedily with respect to the optimal action-value function: $\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s,a)$. Finally, since we are interested in ultimately having agents that can solve tasks in complex environments with unknown transition probabilities and reward functions, we will constrain ourselves to *model-free* RL [Sutton et al., 1998]. There are several methods for learning optimal action-value functions in this setting, such as Q-learning [Watkins, 1989].

2.1 Lattice Theory

To precisely articulate the logical composition of tasks, we will utilize the mathematical framework of lattice theory. Lattice theory is a generalisation of the study of Boolean algebras and can be interpreted through the lens of either order theory or abstract algebra [Grätzer, 2002]. The basic concepts under the order theoretic view are the least upper bounds and greatest lower bounds of elements in a partial order, which correspond to the algebraic concepts of *join* (disjunction) and *meet* (conjunction) of elements. Since we are primarily interested in the conjunction, disjunction, and negation operators, our main focus is on the algebraic interpretation as we build our way up to a Boolean algebra. However, we also use the order theoretic view whenever that seems most natural. In order theory, a lattice is a partial order $(\mathcal{L}, \leq)$ in which every pair of elements $a, b \in \mathcal{L}$ has a *least upper*

[2] When the discount factor is 1, the rewards are assumed to be such that the value functions for improper policies—those that never reach terminal states—are unbounded below [van Niekerk et al., 2019].

bound (their supremum) and a *greatest lower bound (their infimum)* [Grätzer, 2011]. The equivalent algebraic definition is as follows:

Definition 2 *A Lattice algebra $(\mathcal{L}, \vee, \wedge)$ is a set $\mathcal{L}$ equipped with the binary operators $\vee$ and $\wedge$ which satisfies the following lattice axioms for a, b, c in $\mathcal{L}$:*

(i) Idempotence: $a \wedge a = a \vee a = a$.

(ii) Commutativity: $a \wedge b = b \wedge a$ and $a \vee b = b \vee a$.

(iii) Associativity: $a \wedge (b \wedge c) = (a \wedge b) \wedge c$ and $a \wedge (b \vee c) = (a \vee b) \vee c$.

(iv) Absorption: $a \wedge (a \vee b) = a \vee (a \wedge b) = a$.

A lattice order $(\mathcal{L}, \leq)$ induces a lattice algebra $(\mathcal{L}, \vee, \wedge)$ with $\vee$ and $\wedge$ given by $a \vee b \coloneqq \sup\{a, b\}$ and $a \wedge b \coloneqq \inf\{a, b\}$. We can then call a set $\mathcal{B} \subseteq \mathcal{L}$ a *basis* of $\mathcal{L}$ if it is the smallest set that can generate all elements of $\mathcal{L}$ by applying the logical operators of $\mathcal{L}$.[3] While the lattice algebra defines conjunction and disjunction, it says nothing of the negation operator. The *De Morgan algebra*, on the other hand, provides an intuitive notion of negation.

Definition 3 *A De Morgan algebra $(\mathcal{L}, \vee, \wedge, \neg)$ is a set $\mathcal{L}$ equipped with the binary operators $\vee$ and $\wedge$, and the unary operator $\neg$ (involution, negation), which satisfies the following De Morgan algebra axioms for a, b, c in $\mathcal{L}$:*

(i) All the lattice axioms.

(ii) Distributivity: $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$ and $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$.

(iii) Identity: there exists $\mathbf{0}, \mathbf{1}$ in $\mathcal{L}$ such that $\mathbf{0} \wedge a = \mathbf{0}$, $\mathbf{0} \vee a = a$, $\mathbf{1} \wedge a = a, \mathbf{1} \vee a = \mathbf{1}$.

(iv) De Morgan involution: $\neg\neg a = a$ and $\neg(a \vee b) = \neg a \wedge \neg b$.

In other words, a De Morgan algebra is a bounded distributive lattice equipped with a De Morgan involution (a negation operator that satisfies the De Morgan laws).

Finally, notice that the De Morgan involution does not necessarily satisfy the law of the excluded middle ($a \vee \neg a = \mathbf{1}$) and the law of non-contradiction ($a \wedge \neg a = \mathbf{0}$), which are neccessary to have the familiar semantics of propositional logic and set theory [Grätzer, 2011]. A Boolean algebra gives us those semantics by enforcing said laws:

Definition 4 *A Boolean algebra $(\mathcal{L}, \vee, \wedge, \neg)$ is a set $\mathcal{L}$ equipped with the binary operators $\vee$ (disjunction) and $\wedge$ (conjunction), and the unary operator $\neg$ (negation), which satisfies the following Boolean algebra axioms for a, b, c in $\mathcal{L}$:*

(i) All the De Morgan algebra axioms.

(ii) Complements: $\forall a \in \mathcal{L}$, there exists $\neg a = a' \in \mathcal{L}$ such that $a \wedge a' = \mathbf{0}$ and $a \vee a' = \mathbf{1}$.

[3] For simplicity, we adopt the convention of Grätzer [2011] of referring to an algebraic structure (e.g., the lattice $(\mathcal{L}, \vee, \wedge)$) by the set on which it is defined ($\mathcal{L}$), when the distinction is clear given the context.

3. Compositional Tasks

Before we can construct agents capable of solving new compositional tasks, we must formally define these composite tasks.[4] Since reinforcement learning models tasks as MDPs, we require principled ways of composing these MDPs to produce new MDPs that model a desired task specification. This is done by leveraging the structure of lattice algebras since, as outlined above, they abstract the notion of disjunction, conjunction, and negation. Since the Boolean algebra is the most restrictive, we will first start with a general lattice and then increasingly constrain the task space with necessary assumptions on the road to a Boolean algebra. During this process, we will also illustrate some of the different types of semantics of task compositions that result from different lattice structures. We will use the following simple bin-packing domain as a running example.

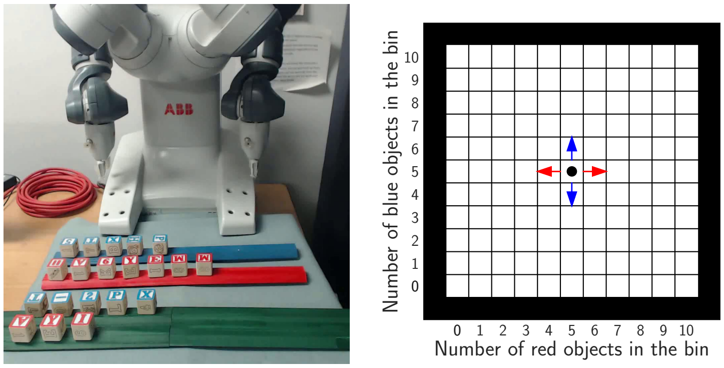


ABB
Number of blue objects in the bin
0
1
2
3
4
5
6
7
8
9
10
Number of red objects in the bin
0 1 2 3 4 5 6 7 8 9 10

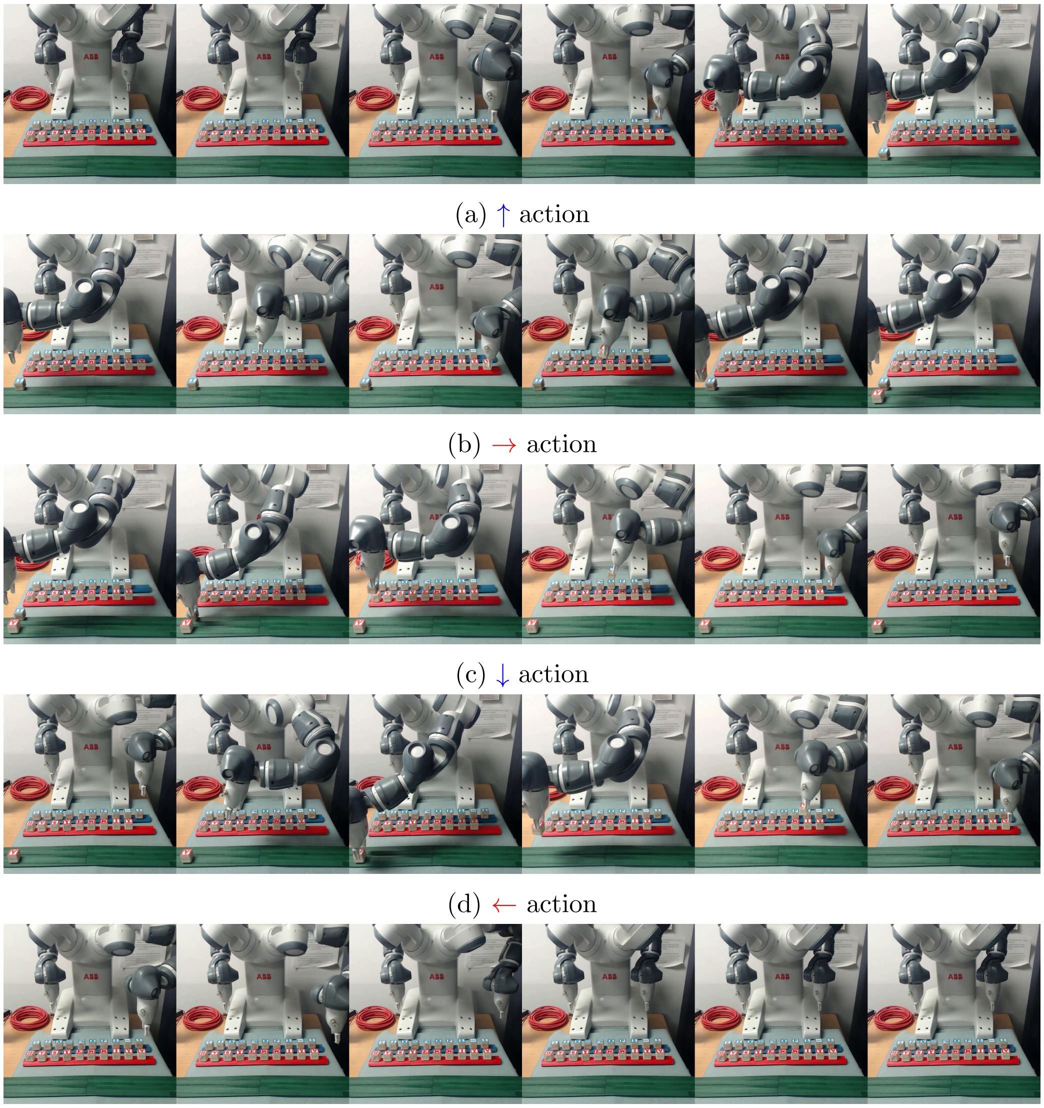
(a) Bin packing domain (b) Gridworld representation

Figure 2: Gridworld representation of the simple bin packing domain.

Example 1 *Consider an environment where an agent needs to manipulate a robotic arm to pack (or unpack) objects into (from) a green bin (Figure 2a). There are 10 red objects and 10 blue objects in the domain, resulting in* 11×11 *states where each state* $s = (r, b)$ *corresponds to the number of red* (r) *and blue* (b) *objects in the bin. The agent has actions to command the robotic arm to put a red object into the bin* ($\rightarrow$)*, remove a red object from the bin* ($\leftarrow$)*, put a blue object into the bin* ($\uparrow$)*, and remove a blue object from the bin* ($\downarrow$)*. These actions do nothing if there are no red or blue objects inside or outside of the bin to execute them. The agent also has a fifth action* ($\bullet$) *for "done" that it chooses to terminate; a state only becomes terminal if the agent chooses the done action in it. Figure 2b illustrates the gridworld representation of this domain, and Figure 3 shows the effect of each action in it.*

Consider a task in which the robot must pack all the red objects into the bin (■). The non-terminal rewards (rewards for all non-terminal transitions) are $R_{MIN} = 0$ *and the terminal rewards range from* $R_{MIN} = 0$ *to* $R_{MAX} = 1$*. The discount factor used is* $\gamma = 0.95$*. That is, an agent receives a reward of* R_{MIN} *as it acts in the environment but when it chooses the done action at any state, it receives a reward between* R_{MIN} *and* R_{MAX} *depending on how close it is to the desired states. Figure 4 shows the terminal rewards, optimal policy, and trajectory of the robot packing red objects into the bin.*

[4] All theorems are stated here, and proofs are presented in the Appendix for clarity.



(a) ↑ action

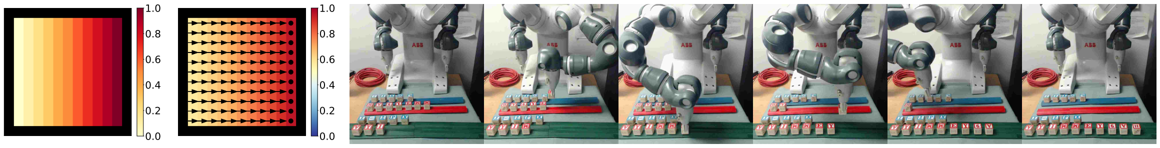
(b) → action

(c) ↓ action

(d) ← action

(e) • action

Figure 3: Example trajectories of the robotic arm for each action in the bin packing domain.



(a) $R_{\blacksquare}$

(b) $Q^*_{\blacksquare}$

(c) Trajectory starting from state (3, 5)

Figure 4: Terminal rewards (a), learned optimal policy and value function (b), and resulting robot arm trajectories for the task in which the robot must pack all the red objects into the bin. The optimal policies and value functions are obtained using Q-learning.

3.1 Task Lattice

Having described how a lattice algebra abstracts the usual concept of disjunction and conjunction, we now formalise the meaning of disjunction and conjunction of tasks. Consider the set of all tasks in an environment:

$$\mathcal{M}_{\mathbb{R}} \coloneqq \{M = (\mathcal{S}, \mathcal{A}, P, R_M, \gamma) | R_M(s, a, s') \in \mathbb{R}\}$$

Since we have constrained ourselves to the model-free setting, we require all our definitions to be usable even when the only information available to the agent from the MDP is its current state s, the next state $s' \sim P(\cdot|s, a)$ and reward $R(s, a, s')$ after taking an action a. Given that tasks differ only in their reward functions (Definition 1), we can achieve this by first defining the partial order over tasks using pointwise $\leq$ (the usual $\leq$ relation on $\mathbb{R}$) over the rewards. The resulting partially ordered set of tasks is formally stated as follows:

Proposition 1 *Let $M_1, M_2 \in \mathcal{M}_{\mathbb{R}}$ be tasks with reward functions R_{M_1} and R_{M_2} respectively. Then $(\mathcal{M}_{\mathbb{R}}, \leq)$ is a partially ordered set with the relation $\leq$ given by*

$$M_1 \leq M_2 \textit{ if } R_{M_1}(s, a, s') \leq R_{M_2}(s, a, s') \textit{ for all } (s, a, s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}.$$

■

Since the reward functions are real-valued, every pair of rewards has a least upper bound (sup) and a greatest lower bound (inf). The resulting real functions after pointwise inf and sup are then clearly still valid task reward functions. Hence the partially-ordered set of tasks $(\mathcal{M}_{\mathbb{R}}, \leq)$ has a least upper bound $\sup\{M_1, M_2\} \in \mathcal{M}_{\mathbb{R}}$ and a greatest lower bound $\inf\{M_1, M_2\} \in \mathcal{M}_{\mathbb{R}}$ for any pair of task $M_1, M_2 \in \mathcal{M}_{\mathbb{R}}$ (since they only differ on their rewards). The lattice $(\mathcal{M}_{\mathbb{R}}, \vee, \wedge)$ induced by this partial order trivially follows with the binary operators $\vee$ and $\wedge$ given by $M_1 \vee M_2 \coloneqq \sup\{M_1, M_2\} \in \mathcal{M}_{\mathbb{R}}$ and $M_1 \wedge M_2 \coloneqq \inf\{M_1, M_2\} \in \mathcal{M}_{\mathbb{R}}$. We define these operators formally as follows:

Definition 5 *The join $\vee : \mathcal{M}_{\mathbb{R}} \times \mathcal{M}_{\mathbb{R}} \to \mathcal{M}_{\mathbb{R}}$ and meet $\wedge : \mathcal{M}_{\mathbb{R}} \times \mathcal{M}_{\mathbb{R}} \to \mathcal{M}_{\mathbb{R}}$ operators over tasks are given by*

$$\vee(M_1, M_2) \coloneqq (\mathcal{S}, \mathcal{A}, P, R_{M_1 \vee M_2}, \gamma), \textit{ where } R_{M_1 \vee M_2} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R} \\ (\cdot) \mapsto \sup\{R_{M_1}(\cdot), R_{M_2}(\cdot)\},$$

$$\wedge(M_1, M_2) \coloneqq (\mathcal{S}, \mathcal{A}, P, R_{M_1 \wedge M_2}, \gamma), \textit{ where } R_{M_1 \wedge M_2} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R} \\ (\cdot) \mapsto \inf\{R_{M_1}(\cdot), R_{M_2}(\cdot)\}.$$

In fact, $(\mathcal{M}_{\mathbb{R}}, \vee, \wedge)$ forms a *distributive lattice* since *inf* and *sup* with the usual $\leq$ in $\mathbb{R}$ is distributive:

Proposition 2 $(\mathcal{M}_{\mathbb{R}}, \vee, \wedge)$ *is a distributive lattice.* ■

This allows us to jointly apply disjunction and conjuction operators to sets of tasks. Given a non-empty finite set $\mathcal{O}$ of lower bounded subsets of tasks $\mathcal{N} \subset \mathcal{M}_{\mathbb{R}}$, the task lattice $(\mathcal{M}_{\mathbb{R}}, \vee, \wedge)$ gives us the principled way of specifying the disjunction of conjunctions:

$$\bigvee_{\mathcal{N}\in\mathcal{O}}\left(\bigwedge_{N\in\mathcal{N}} N\right) = (\mathcal{S},\mathcal{A},P,R_{\underset{\mathcal{O}}{\vee}\underset{\mathcal{N}}{\wedge}},\gamma), \quad \text{where} \quad R_{\underset{\mathcal{O}}{\vee}\underset{\mathcal{N}}{\wedge}}(s,a,s') \coloneqq \sup_{\mathcal{N}\in\mathcal{O}}\left(\inf_{N\in\mathcal{N}} R_N(s,a,s')\right).$$

Similarly, given a non-empty finite set $\mathcal{O}$ of upper bounded subsets of tasks $\mathcal{N} \subset \mathcal{M}_{\mathbb{R}}$, the conjunction of disjunctions is given by,

$$\bigwedge_{\mathcal{N}\in\mathcal{O}}\left(\bigvee_{N\in\mathcal{N}} N\right) = (\mathcal{S},\mathcal{A},P,R_{\underset{\mathcal{O}}{\wedge}\underset{\mathcal{N}}{\vee}},\gamma), \quad \text{where} \quad R_{\underset{\mathcal{O}}{\wedge}\underset{\mathcal{N}}{\vee}} : (s,a,s') \coloneqq \inf_{\mathcal{N}\in\mathcal{O}}\left(\sup_{N\in\mathcal{N}} R_N(s,a,s')\right).$$

Example 2 *Consider the bin packing domain introduced in Example 1. Further consider the specification of two tasks, 🟥 and 🟦, in which the robot must pack all the red and blue objects into the bin respectively. We can use a Hasse diagram to visualise the task sub-lattice generated by all combinations of disjunction and conjunction of 🟥 and 🟦 (Figure 5). A Hasse diagram illustrates a lattice $\mathcal{M}$ by drawing an edge between $M_1 \in \mathcal{M}$ and $M_2 \in \mathcal{M}$ if they are comparable, and M_1 is drawn below M_2 if $M_1 \leq M_2$ [Grätzer, 2011]. Hence, 🟥 $\wedge$ 🟦 is connected to 🟥 and 🟦 directly below them because it is their greatest lower bound. Similarly, 🟥 $\vee$ 🟦 is connected to 🟥 and 🟦 directly above them because it is their least upper bound.*

Finally, Figure 6 shows the terminal rewards (all non-terminal rewards are 0), optimal policy and value function for all these tasks. We can observe how the sup and inf of task rewards indeed result in composite task rewards with the correct semantics for the disjunctive and conjunctive tasks respectively. To learn the corresponding optimal policies and value functions, we use Q-learning where the transition rewards are obtained using $\sup\{R_{🟥}(s,a,s'), R_{🟦}(s,a,s')\}$ *for their disjunction and* $\inf\{R_{🟥}(s,a,s'), R_{🟦}(s,a,s')\}$ *for their conjunction.*

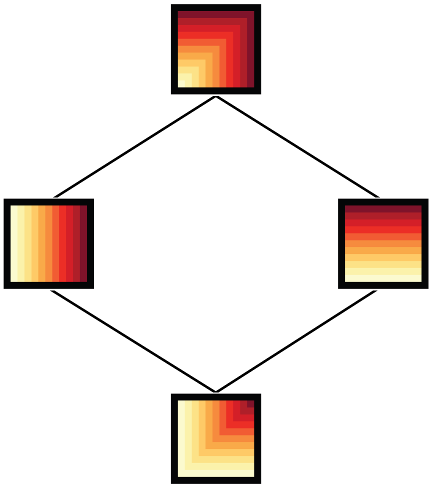
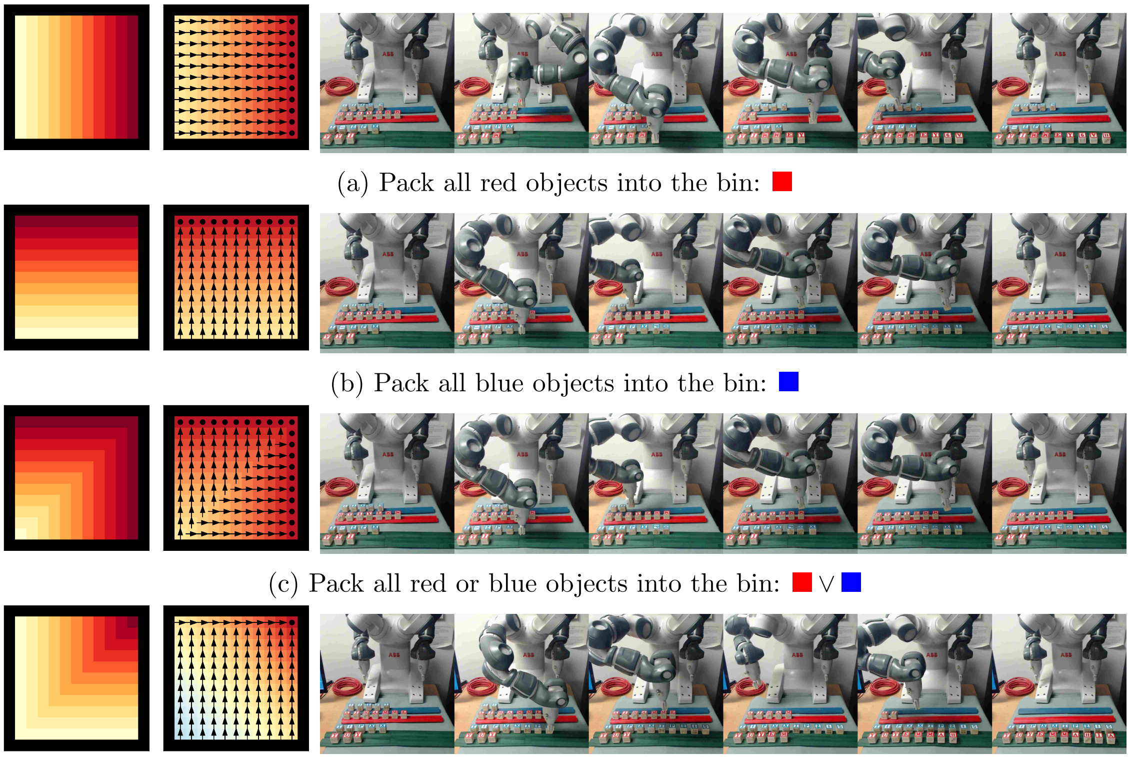


Figure 5: Hasse diagram of the task sub-lattice with basis {🟥,🟦}.



(a) Pack all red objects into the bin: 🟥

(b) Pack all blue objects into the bin: 🟦

(c) Pack all red or blue objects into the bin: 🟥 $\vee$ 🟦

(d) Pack all red and blue objects into the bin: 🟥 $\wedge$ 🟦

Figure 6: Terminal rewards (left column heat maps), optimal value functions (middle column heat maps), optimal policies (middle column arrows), and sample robot trajectories (right column images) for the disjunction (c) and conjunction (d) of tasks (a-b) in the bin packing domain. The optimal policies and value functions are obtained using Q-learning.

3.2 De Morgan Task Algebra

Having formalised the meaning of task disjunction and conjunction, we next turn our attention to the negation of tasks. As discussed in Section 2, the De Morgan algebra allows us to define this operator by adding the minimal required properties that encapsulate the desired semantics of a negation. In particular, we only need the set of tasks to be bounded by some tasks $\mathcal{M}_{INF}, \mathcal{M}_{SUP} \in \mathcal{M}_{\mathbb{R}}$:

$$\mathcal{M}_{[\mathcal{M}_{INF},\mathcal{M}_{SUP}]} \coloneqq \{M = (\mathcal{S}, \mathcal{A}, P, R_M, \gamma) \mid R_M(s,a,s') \in [R_{\mathcal{M}_{INF}}(s,a,s'), R_{\mathcal{M}_{SUP}}(s,a,s')]\}$$

We can now define the negation of a task as follows:

Definition 6 *Define the negation operator* $\neg : \mathcal{M}_{[\mathcal{M}_{INF},\mathcal{M}_{SUP}]} \to \mathcal{M}_{[\mathcal{M}_{INF},\mathcal{M}_{SUP}]}$ *as*

$$\neg(M) \coloneqq (\mathcal{S}, \mathcal{A}, P, R_{\neg M}, \gamma), \text{ where } R_{\neg M} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$$
$$(\cdot) \mapsto (R_{\mathcal{M}_{SUP}}(\cdot) + R_{\mathcal{M}_{INF}}(\cdot)) - R_M(\cdot).$$

The above definition captures the intuition behind negation. For example, if an agent takes an action at a given state and receives the smallest reward for the resulting transition, then the agent acting in the opposite task should receive the highest reward for that same transition. Note how for $M \in \mathcal{M}_{[\mathcal{M}_{INF}, \mathcal{M}_{SUP}]}$, $(\mathcal{S}, \mathcal{A}, P, R_{\neg M}, \gamma) \in \mathcal{M}$ because $R_{\neg M}(s, a, s')$ is also bounded by $[R_{\mathcal{M}_{INF}}(s, a, s'), R_{\mathcal{M}_{SUP}}(s, a, s')]$. Hence, $\neg$ is closed in $\mathcal{M}_{[\mathcal{M}_{INF}, \mathcal{M}_{SUP}]}$.

Finally, we formalise the interaction of the negation of tasks with the conjunction and disjunction of tasks as follows:

Proposition 3 $(\mathcal{M}_{[\mathcal{M}_{INF}, \mathcal{M}_{SUP}]}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ *is a De Morgan algebra.* ■

We can now specify arbitrary disjunction, conjunction, and negation of tasks.

Example 3 *In the bin packing environment, consider the De Morgan lattice bounded by the tasks where all non-terminal rewards are 0, and all terminal rewards are 0 (🟨) for the lower bound and 1 (🟫) for the upper bound. Further consider the specification of two tasks, 🟥 and 🟦, in which the robot must pack all the red and blue objects into the bin respectively (the same ones from Example 2). Figure 7 shows the rewards, optimal policies, optimal value functions, and resulting robot trajectories for sample task compositions. We can observe how the negation of tasks, and its interaction with the disjunction and conjunction of tasks, results in composite tasks with desired semantics. For example, the negations ¬🟥 and ¬🟦 specify the tasks in which the robot must remove all red objects and all blue objects from the bin respectively. Figure 8 shows the De Morgan sub-lattice generated by all combinations of disjunction, conjunction, and negation of 🟥 and 🟦.*

Finally, notice that the task bounds are what define the semantic meaning of the negation operator, with different task bounds leading to different semantics. For example, when using the task bounds {🟨, 🟫}, the negation of the task in which the robot must pack or unpack all red objects into (from) the bin (▥) is to pack or unpack exactly half of the red objects into (from) the bin (¬▥ = ▯). However, when using the task bounds {⊞, ▣}, the negation is instead to pack or unpack all the blue objects into (from) the bin (¬▥ = ▤).

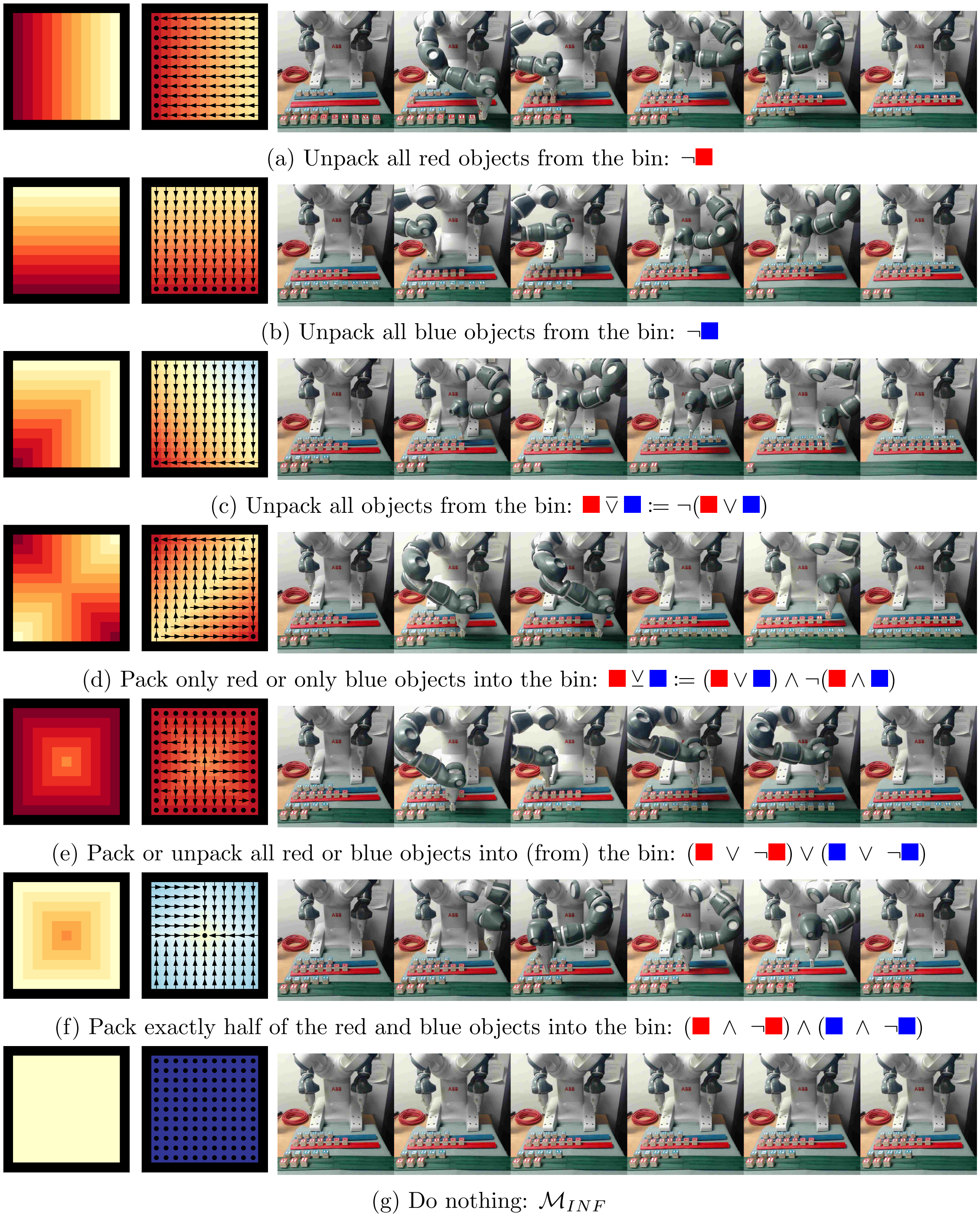
3.3 Boolean Task Algebra

While the De Morgan task algebra allows for logical composition of tasks with arbitrary bounded rewards, it provides no guarantees on certain desired properties. In particular, these task compositions do not always satisfy the laws of the excluded middle ($M_1 \vee \neg M_1 = \mathcal{M}_{SUP}$), and of non-contradiction ($M_1 \wedge \neg M_1 = \mathcal{M}_{INF}$). This can clearly be seen in Figure 7f, where the agent needs to *pack all red objects into the bin and remove all red objects from the bin* and *pack all blue objects into the bin and remove all blue objects from the bin.* In this case, the choice of rewards produces a meaningful task—*pack exactly half of the red and blue objects into the bin*— but in general, we may want to guarantee that contradicting task specifications are meaningless. To achieve this, we need to restrict the set of tasks to those with *binary rewards*—binary here means the rewards are either $R_{\mathcal{M}_{INF}}(s, a, s')$ or $R_{\mathcal{M}_{SUP}}(s, a, s')$—which ensures that tasks have a Boolean nature:

$$\mathcal{M}_{\{\mathcal{M}_{INF}, \mathcal{M}_{SUP}\}} \coloneqq \{M = (\mathcal{S}, \mathcal{A}, P, R_M, \gamma) \mid R_M(s, a, s') \in \{R_{\mathcal{M}_{INF}}(s, a, s'), R_{\mathcal{M}_{SUP}}(s, a, s')\}\}$$

We can now formalise a Boolean logic on the set of tasks.

Proposition 4 $(\mathcal{M}_{\{\mathcal{M}_{INF}, \mathcal{M}_{SUP}\}}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ *is a Boolean algebra.* ■



(a) Unpack all red objects from the bin: $\neg$🟥

(b) Unpack all blue objects from the bin: $\neg$🟦

(c) Unpack all objects from the bin: 🟥 $\bar{\vee}$ 🟦 $:= \neg($🟥 $\vee$ 🟦$)$

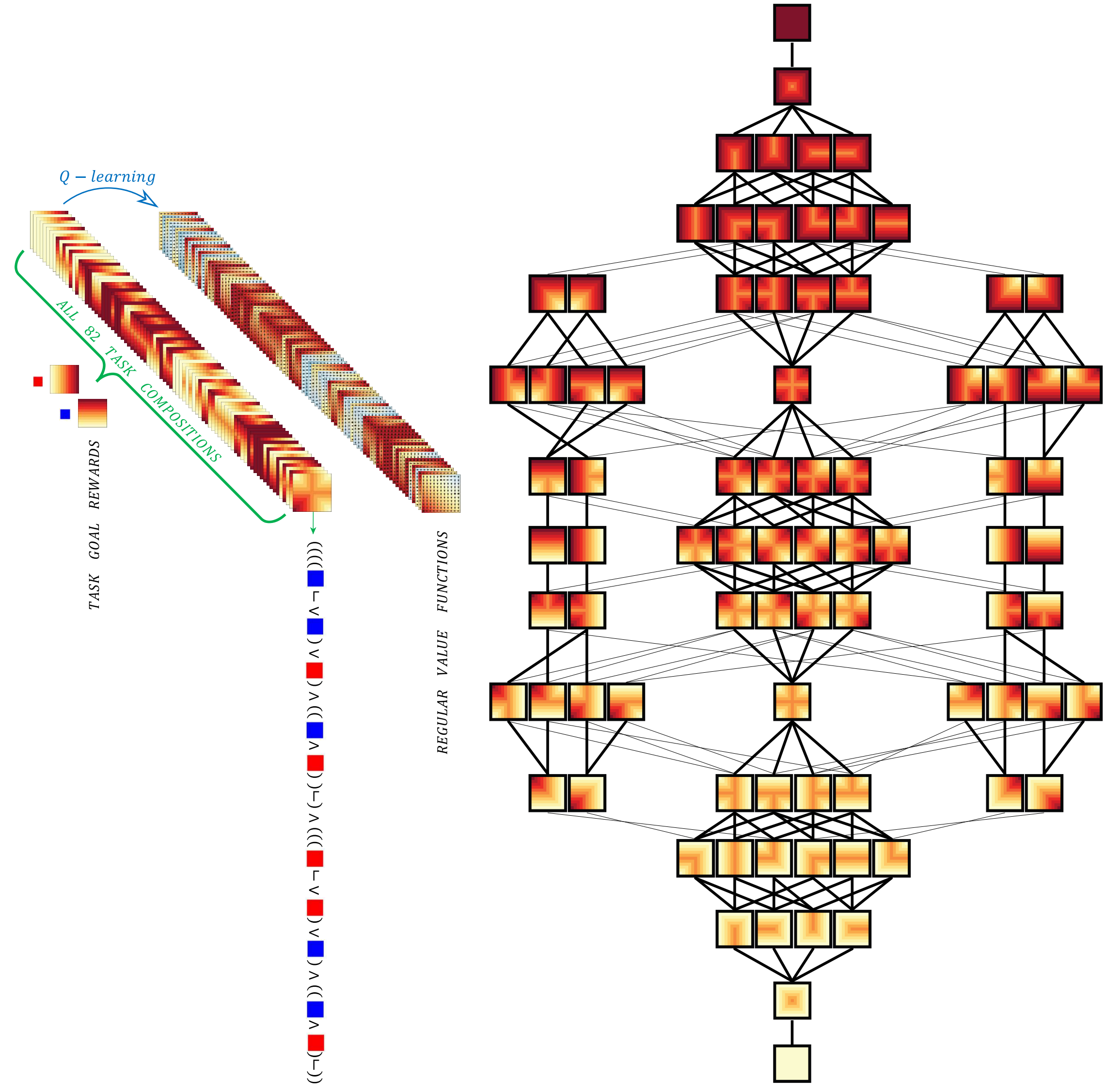
(d) Pack only red or only blue objects into the bin: 🟥 $\veebar$ 🟦 $:= ($🟥 $\vee$ 🟦$) \wedge \neg($🟥 $\wedge$ 🟦$)$

(e) Pack or unpack all red or blue objects into (from) the bin: $($🟥 $\vee$ $\neg$🟥$) \vee ($🟦 $\vee$ $\neg$🟦$)$

(f) Pack exactly half of the red and blue objects into the bin: $($🟥 $\wedge$ $\neg$🟥$) \wedge ($🟦 $\wedge$ $\neg$🟦$)$

(g) Do nothing: $\mathcal{M}_{INF}$

Figure 7: Terminal rewards (left column heat maps), optimal value functions (middle column heat maps), optimal policies (middle column arrows), and sample robot trajectories (left column images) for the lower bound task $\mathcal{M}_{INF}$ and compositions of tasks (🟥, 🟦) in the bin packing domain. Optimal policies and value functions are obtained using Q-learning.



(a) All task compositions of 🟥 and 🟦

(b) Hasse diagram

Figure 8: Illustration of the De Morgan sub-lattice with basis {🟥, 🟦}. (a) Illustrates how the rewards for any logical task specification (82 unique ones in this case) can be generated by simply composing the rewards of the basis tasks, with guaranteed semantics. (b) Shows the relationship between all the 82 unique tasks that can be specified by composing the red and blue tasks.

Finally, we summarise the main differences between the developed task algebras in Table 1. While the Boolean task algebra necessitates the most assumptions, it is also the most powerful. In the next section, we will show that an additional benefit of Boolean task algebra is its basis can be obtained directly from the task bounds.

	Assumptions			Benefits		
	Same environment	Same reward bounds	Binary rewards	Union and intersection	Negation	Full Boolean logic
Task lattice	✓			✓		
De Morgan task algebra	✓	✓		✓	✓	
Boolean task algebra	✓	✓	✓	✓	✓	✓

Table 1: Trade-offs between the necessary assumptions and benefits of each lattice structure.

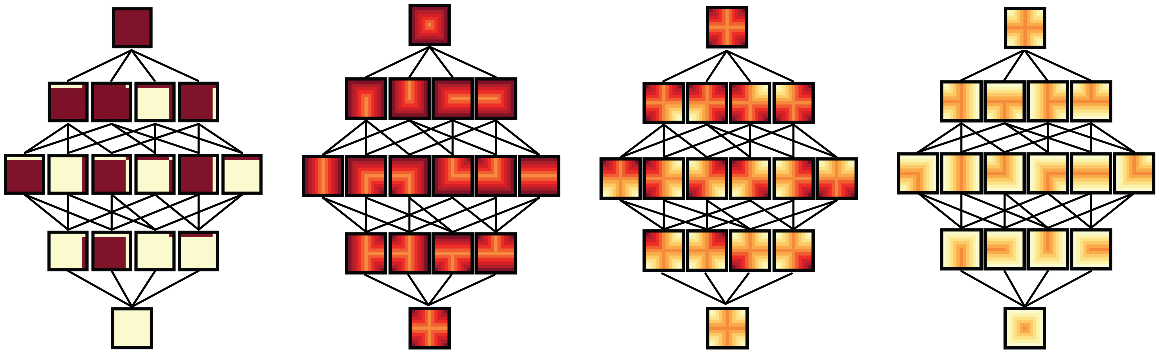
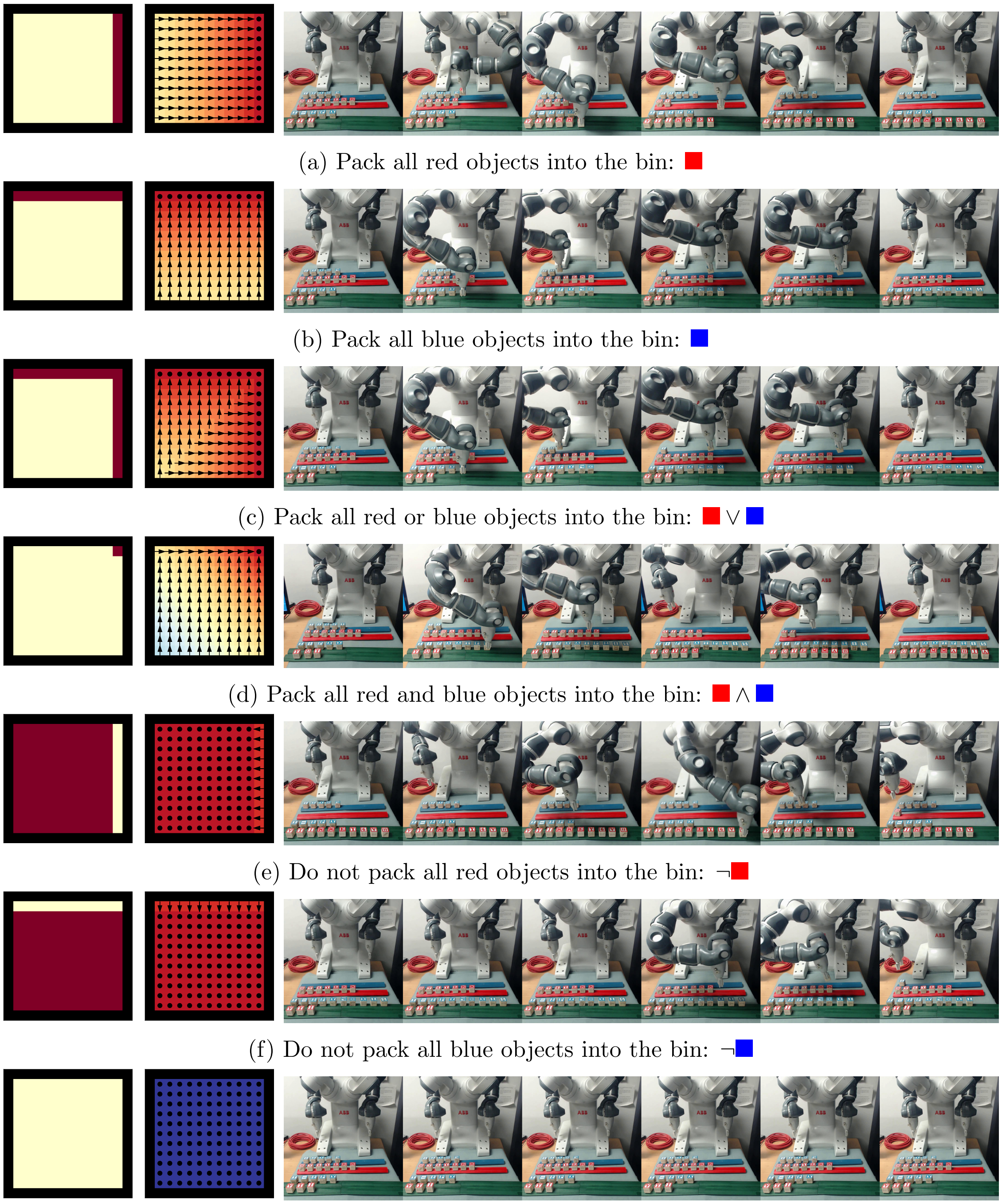


Figure 9: Examples of Boolean task sub-algebra with basis {□,□}, {▨,▨}, {▨,▨}, {▨,▨} and task space bounds {□,■}, {▨,■}, {▨,▨}, {▨,▨} respectively.

Example 4 *In the bin packing domain, consider the Boolean algebra bounded by the tasks where all non-terminal rewards are 0, and all terminal rewards are 0 (□) for the lower bound and 1 (■) for the upper bound. Consider the specification of two tasks, 🟥 and 🟦, in which the robot must pack all the red and blue objects into the bin respectively. Unlike before, their respective rewards are binary for this example: □,□. Figure 10 shows the rewards, optimal policies, optimal value functions, and resulting robot trajectories for sample logical compositions. Note how the binary rewards result in semantically different compositions. For example, the negation of the 🟥 task here now means "do not pack all red objects in the bin". Also, meaningless compositions like (🟥 ∧ ¬🟥) ∧ (🟦 ∧ ¬🟦) now produce the lower bound task □, where all states have 0 rewards.*

Figure 9 shows the Boolean sub-algebra generated by all logical compositions of {🟥,🟦} (leftmost), and other examples resulting from different choices of task space bounds and basis. This illustrates an interesting result: It is not necessary to restrict all transition rewards to 0 or 1—that is, use the task bounds {□,■}—to obtain Boolean logic over tasks. As it turns out, any transition reward can be used as long as they are the transition rewards of some choice of task bounds. All such rewards are valid, with different task bounds simply leading to different operator semantics.



(a) Pack all red objects into the bin: 🟥

(b) Pack all blue objects into the bin: 🟦

(c) Pack all red or blue objects into the bin: 🟥 ∨ 🟦

(d) Pack all red and blue objects into the bin: 🟥 ∧ 🟦

(e) Do not pack all red objects into the bin: ¬🟥

(f) Do not pack all blue objects into the bin: ¬🟦

(g) Pack and do not pack all red and blue objects into the bin: (🟥 ∧ ¬🟥) ∧ (🟦 ∧ ¬🟦)

Figure 10: Terminal rewards (left), optimal values and policies (middle) and sample robot trajectories (right) for the compositions of Boolean tasks in the bin packing domain. Optimal policies and value functions are obtained using Q-learning.

3.4 Constructing a Basis Set of Tasks

In the previous sections, we demonstrated how lattice structures can formally define the logical composition of tasks with varying semantics, enabling the specification of new tasks as compositions of a set of known ones. However, an arbitrary set of tasks may contain redundant elements—tasks that are compositions of others within the set—which limits the number of uniquely specifiable tasks and, consequently, the ability to represent every task in a given task space. Hence, a natural question arises: how can we construct a basis set of tasks—a minimal set capable of generating every task in the space?

In this section, we show how a Boolean task algebra provides an answer to this question. To do so, we first establish an equivalence between a Boolean task algebra and a power set algebra. We note that the assumption of binary rewards induces a bijection F between the set of tasks $\mathcal{M}_{\{\mathcal{M}_{INF},\mathcal{M}_{SUP}\}}$ and the power-set $2^{\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$, given by:

$$\begin{aligned} F : 2^{\mathcal{S}\times\mathcal{A}\times\mathcal{S}} &\to \mathcal{M}_{\{\mathcal{M}_{INF},\mathcal{M}_{SUP}\}} \\ \mathcal{H} &\mapsto (\mathcal{S}, \mathcal{A}, P, R_{\mathcal{H}}, \gamma), \text{ where } R_{\mathcal{H}} : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to \mathbb{R} \\ &\qquad (s,a,s') \mapsto \begin{cases} R_{\mathcal{M}_{SUP}}(s,a,s'), & \text{if } (s,a,s') \in \mathcal{H} \\ R_{\mathcal{M}_{INF}}(s,a,s'), & \text{if } (s,a,s') \notin \mathcal{H}. \end{cases} \end{aligned}$$

The Boolean task algebra together with the bijection between tasks $\mathcal{M}_{\{\mathcal{M}_{INF},\mathcal{M}_{SUP}\}}$ and the power-set $2^{\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$ gives us the following result:

Proposition 5 *The Boolean task algebra on $\mathcal{M}_{\{\mathcal{M}_{INF},\mathcal{M}_{SUP}\}}$ is isomorphic to the power set Boolean algebra on $2^{\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$.* ■

As a consequence, all results that hold for power set Boolean algebras now also hold for Boolean task algebras. In particular, consider the Boolean algebra on a set of tasks $\mathcal{M}$ where rewards are binary and depend only on a subset of the state space $\mathcal{G} \subseteq \mathcal{S}$ (such as Example 4). The corresponding Boolean task algebra is isomorphic to the power set algebra on $\mathcal{G}$, with the isomorphism $F : 2^{\mathcal{G}} \to \mathcal{M}$ given by:

$$\begin{aligned} \mathcal{H} &\mapsto (\mathcal{S}, \mathcal{A}, P, R_{\mathcal{H}}, \gamma), \text{ where } R_{\mathcal{H}} : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to \mathbb{R} \\ &\qquad (s,a,s') \mapsto \begin{cases} R_{\mathcal{M}_{SUP}}(s,a,s'), & \text{if } s \in \mathcal{H} \\ R_{\mathcal{M}_{INF}}(s,a,s'), & \text{if } s \notin \mathcal{H}. \end{cases} \end{aligned}$$

This means that for $\mathcal{M}$ with finite $\mathcal{G}$, we require only a logarithmic number of basis tasks (minimal generators) $\lceil \log_2 |\mathcal{G}| \rceil$ (for $|\mathcal{G}| > 1$) to specify an exponential number of composed tasks $|\mathcal{M}| = 2^{|\mathcal{G}|}$. For example, in the bin packing domain with binary rewards (Example 4), all $|\mathcal{M}| = 2^{121}$ tasks can be specified by composing only $\lceil \log_2 121 \rceil = 7$ basis tasks (Figure 11). The following example demonstrates how one can construct such a basis using a Boolean table.

Example 5 *Consider the Four Rooms domain [Sutton et al., 1999], where an agent must navigate a gridworld to particular rooms. The agent can move in any of the four cardinal directions at each timestep, but colliding with a wall leaves the agent in the same location. There is a fifth action for "done" that the agent chooses to achieve goals $\mathcal{G}$ (center of a*

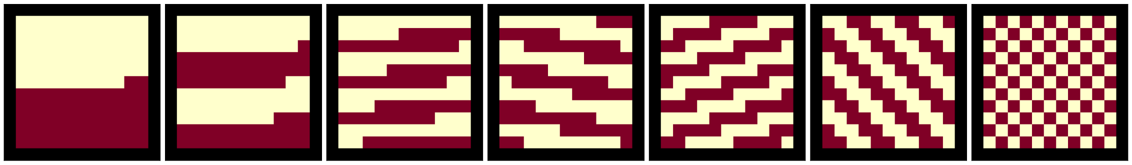
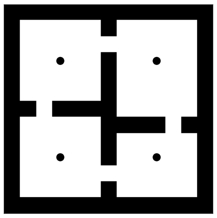


Figure 11: Basis (minimal generators) for the Boolean task algebra bounded by {□, ■}.



(a) Four Rooms domain. The circles indicate goals the agent must reach ($\mathcal{G}$).

Goals ($\mathcal{G}$)	M_D	M_R
top-left	0	0
top-right	0	1
bottom-left	1	0
bottom-right	1	1

(b) Goals labeled by the well order $\leq$ given by: `top-left` $\leq$ `top-right` $\leq$ `bottom-left` $\leq$ `bottom-right`.

Goals ($\mathcal{G}$)	M_T	M_L
bottom-right	0	0
bottom-left	0	1
top-right	1	0
top-left	1	1

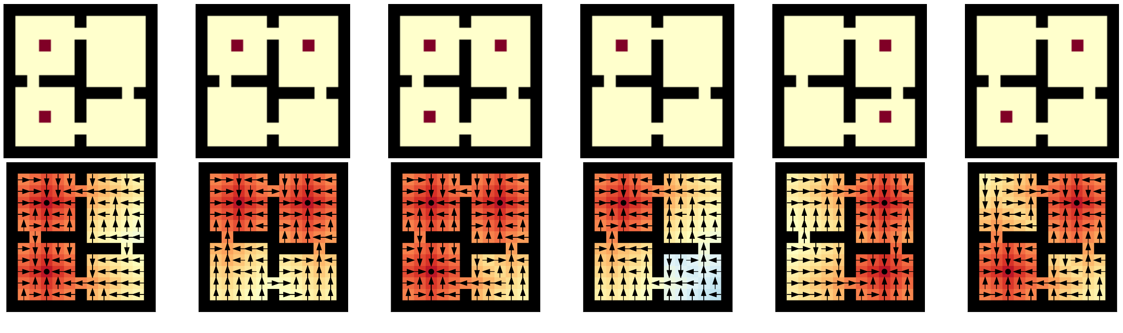
(c) Goals labeled by the well order $\leq$ given by: `bottom-right` $\leq$ `bottom-left` $\leq$ `top-right` $\leq$ `top-left`.

Figure 12: Basis tasks in the Four Rooms domain induced by various well orders on $\mathcal{G}$. Each column in (b)-(c) represents a basis task, where **0** or **1** for goal g on task M means a reward of $R_M(g, a, s') = R_{\text{MIN}}$ or $R_M(g, a, s') = R_{\text{MAX}}$ $\forall a \in \mathcal{A}$ and s' absorbing, respectively. The goal rewards are sufficient to specify the tasks since the non-terminal rewards are R_{MIN}.

room). A goal position only becomes terminal if the agent chooses the done action in it. The non-terminals rewards are $R_{MIN} = -0.1$ and the goal rewards (rewards on the terminal set) are binary ($R_{MIN} = -0.1$ or $R_{MAX} = 2$). The discount factor used is $\gamma = 1$. Figure 12a illustrates the layout of the environment and the goals the agent must reach.

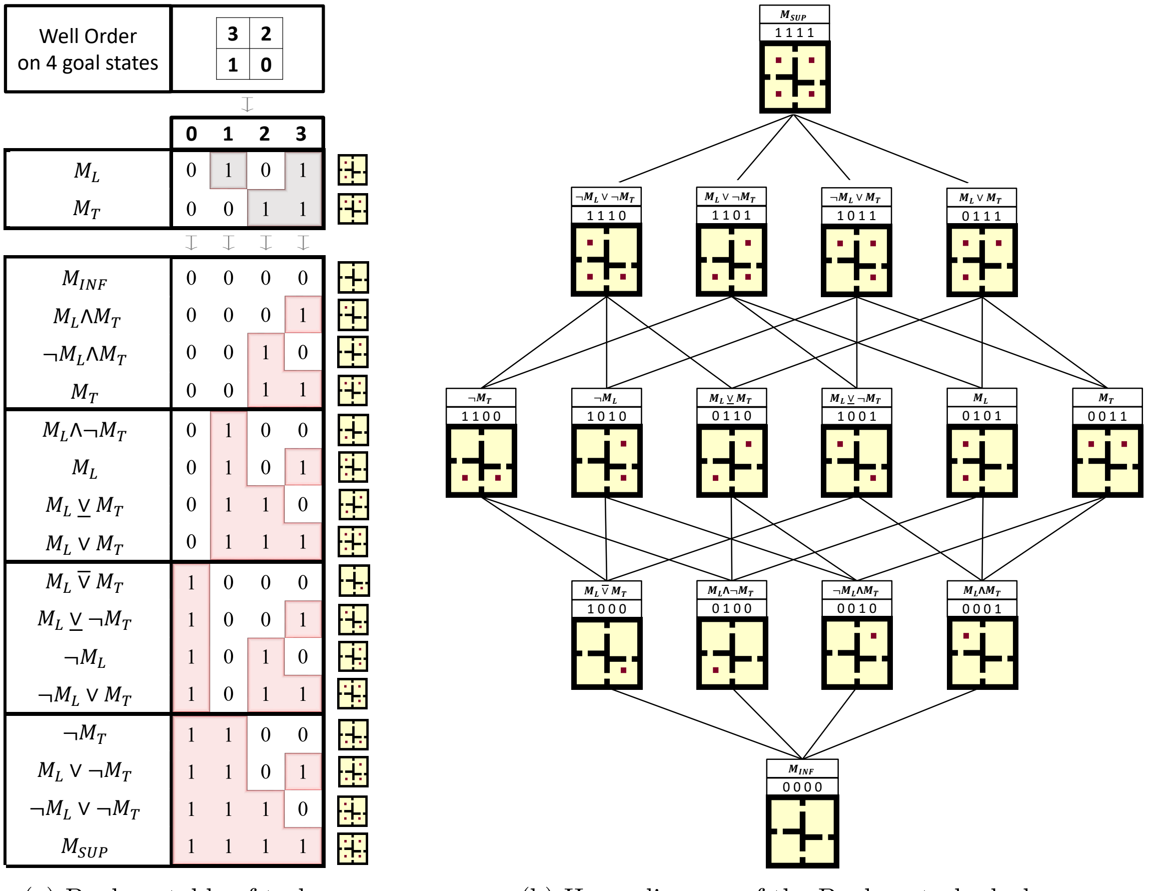
We can select a minimal set of generator tasks (basis tasks) by assigning each goal a binary number, and then using the columns of the table to select the tasks. Since the set of achievable goals (the terminal set) is finite, this assignment can be done using a Boolean table. We first assign labels to the individual goals by defining a well order over the set $\mathcal{G}$. Since there are four goals, the number of basis tasks induced by this well order is $\lceil |\log_2 \mathcal{G}| \rceil = 2$. Tables 12b-12c illustrates how different well orders on $\mathcal{G}$ leads to different choices of basis tasks. Consider, for example, the well order on $\mathcal{G}$ shown in Table 12c. The basis tasks induced are M_L and M_T, in which an agent must navigate to the two left rooms and the two top rooms respectively. Figure 13 shows the rewards and optimal policies of the tasks specified by some of their logical compositions. Figure 14 shows the Boolean table and Hasse diagram for all $2^{2^k} = 16$ tasks generated by the $k = 2$ basis, which spans the whole set of tasks, $|\mathcal{M}| = 2^4$.

This demonstrates how for any Boolean task algebra $\mathcal{M}$, we can construct a task basis for it given the reward bounds $\{R_{\mathcal{M}_{INF}}, R_{\mathcal{M}_{SUP}}\}$ and the subset of states $\mathcal{G}$ (or state transitions) where those reward bounds differ. Interestingly, even in lifelong learning setting where the agent is not given the reward bounds and full set of goals upfront, it can still gradually learn these quantities throughout its lifetime while building a basis set of tasks [Tasse et al., 2022].



(a) M_L (b) M_T (c) $M_L \vee M_T$ (d) $M_L \wedge M_T$ (e) $\neg M_L$ (f) $M_L \veebar M_T$

Figure 13: Compositions of basis tasks $\{M_L, M_T\}$ in the Four Rooms domain—where $\veebar$ represents exclusive disjunction. (top) Rewards for the basis and composed tasks, (bottom) The optimal policies and value functions obtained using Q-learning.



Well Order on 4 goal states	3 2 1 0			
	0	**1**	**2**	**3**
M_L	0	1	0	1
M_T	0	0	1	1
M_{INF}	0	0	0	0
$M_L \wedge M_T$	0	0	0	1
$\neg M_L \wedge M_T$	0	0	1	0
M_T	0	0	1	1
$M_L \wedge \neg M_T$	0	1	0	0
M_L	0	1	0	1
$M_L \veebar M_T$	0	1	1	0
$M_L \vee M_T$	0	1	1	1
$M_L \overline{\vee} M_T$	1	0	0	0
$M_L \veebar \neg M_T$	1	0	0	1
$\neg M_L$	1	0	1	0
$\neg M_L \vee M_T$	1	0	1	1
$\neg M_T$	1	1	0	0
$M_L \vee \neg M_T$	1	1	0	1
$\neg M_L \vee \neg M_T$	1	1	1	0
M_{SUP}	1	1	1	1

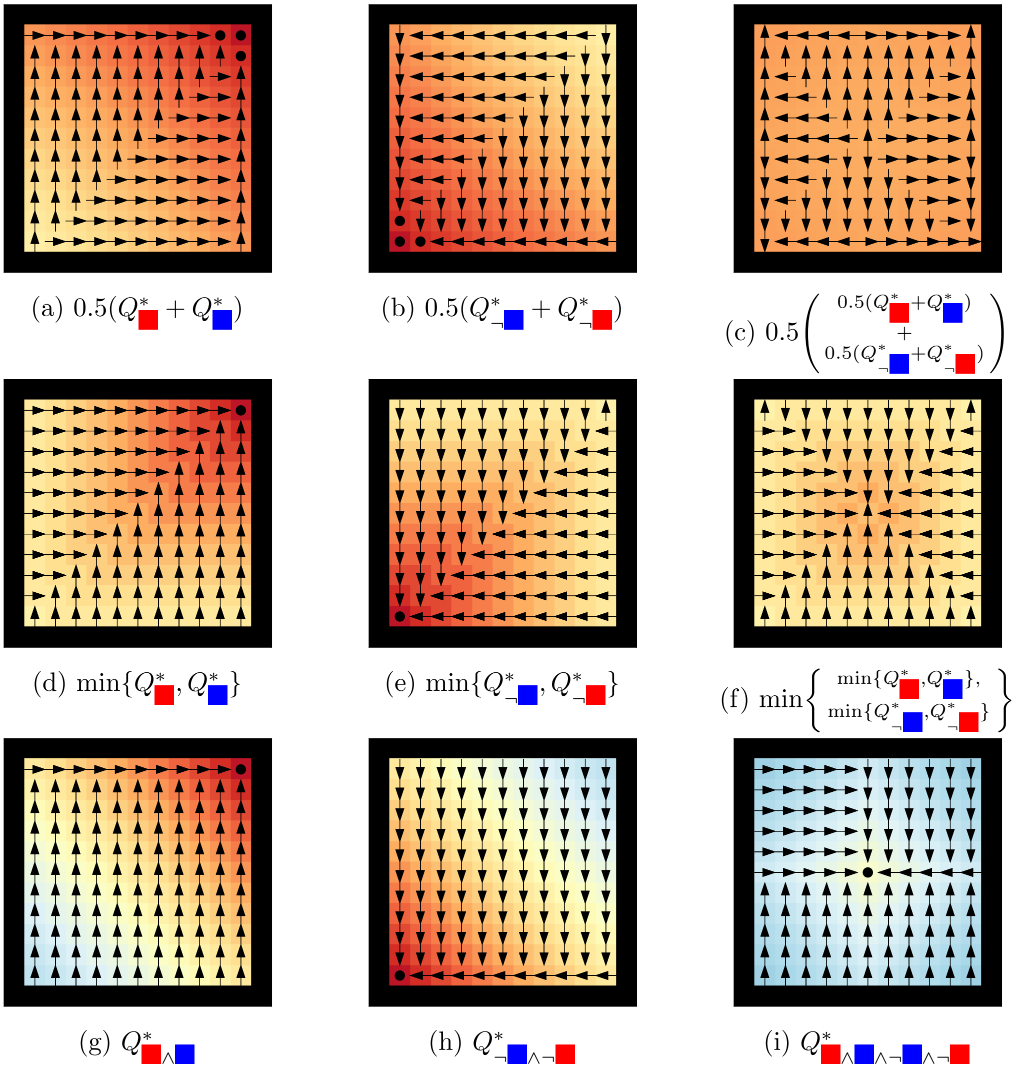
(a) Boolean table of tasks.

(b) Hasse diagram of the Boolean task algebra.

Figure 14: Boolean table and Hasse diagram for the Four Rooms domain. (a) Illustration of a well order on $\mathcal{G}$ that labels the goals, the induced basis tasks, the Boolean values and rewards for all 16 compositions of the basis tasks. (b) Relationship between all tasks in $\mathcal{M}$.

4. World Value Functions

In the previous section, we formally established the logical composition of tasks. The question now is whether we can leverage knowledge gained in a set of tasks to immediately solve novel composite tasks in a zero-shot manner. Unfortunately, while we can combine value functions using pointwise *max* to solve a disjunction of tasks [van Niekerk et al., 2019], the conjunction operator poses major problems. For example, Figure 15 shows how taking the pointwise *average* [Haarnoja et al., 2018a; van Niekerk et al., 2019] or pointwise *min* of standard value functions does not always solve the corresponding task composition.



(a) $0.5(Q^*_{\text{🟥}} + Q^*_{\text{🟦}})$

(b) $0.5(Q^*_{\neg\text{🟦}} + Q^*_{\neg\text{🟥}})$

(c) $0.5\left(\begin{matrix}0.5(Q^*_{\text{🟥}}+Q^*_{\text{🟦}})\\+\\0.5(Q^*_{\neg\text{🟦}}+Q^*_{\neg\text{🟥}})\end{matrix}\right)$

(d) $\min\{Q^*_{\text{🟥}}, Q^*_{\text{🟦}}\}$

(e) $\min\{Q^*_{\neg\text{🟦}}, Q^*_{\neg\text{🟥}}\}$

(f) $\min\left\{\begin{matrix}\min\{Q^*_{\text{🟥}},Q^*_{\text{🟦}}\},\\\min\{Q^*_{\neg\text{🟦}},Q^*_{\neg\text{🟥}}\}\end{matrix}\right\}$

(g) $Q^*_{\text{🟥}\wedge\text{🟦}}$

(h) $Q^*_{\neg\text{🟦}\wedge\neg\text{🟥}}$

(i) $Q^*_{\text{🟥}\wedge\text{🟦}\wedge\neg\text{🟦}\wedge\neg\text{🟥}}$

Figure 15: Consider the four tasks, 🟥, ¬🟥, 🟦, and ¬🟦 in the bin packing domain. (a-c) Approximate conjunction using the pointwise average of their respective optimal Q-functions, (d-f) approximate conjunction using the pointwise *min* of their respective optimal Q-functions, and (g-i) optimal conjunction learned using Q-learning.

To understand why, consider two tasks that have multiple different goals, but where at least one of those goals is common. Clearly, there is a meaningful conjunction between them—namely, to achieve the common goal. Now consider an agent that learns standard value functions for both tasks, and which is then required to solve their conjunction with no further learning. Notice how this is impossible in general, since the standard value function for each task only represents the value of each state with respect to the *best* goal. That is, for all states where the best goal for each task is *not* the common goal, the agent has no information about that common goal. The top right corner of Figure 15e shows an example of how this can lead to incorrect actions.

Hence, for an agent to be capable of solving new tasks without additional learning, it needs to have gained sufficient information from its experience when learning to solve previous tasks. Thus, its goal during learning should not be to learn an optimal policy or standard value function, since it only encodes how to maximise the current task rewards. Instead, it may need to learn an optimal general value function (GVF) [Sutton et al., 2011] that encodes how to maximise both the current task rewards as well as all task rewards in a task space. Precisely, it may need to learn $|\mathcal{M}|$ value functions. However, this is clearly impractical since the number of tasks is potentially infinite ($\mathcal{M}_{\mathbb{R}}$ for example). Hence, we introduce a type of general value function called a *world value function* (WVF) which can be learned model-free, and for some (but not all) task spaces encodes how to maximise a sufficient number of task rewards corresponding to reaching all goal states ($\mathcal{G}$) in the environment. Veeriah et al. [2018] refers to this kind of knowledge as having "mastery" of the environment. Precisely, a WVF is a set of $|\mathcal{G}| \leq |\mathcal{S}|$ value functions. We will show that they are sufficient for *goal-reaching tasks*—a slight generalisation of the shortest path tasks considered in prior zero-shot composition work [van Niekerk et al., 2019; Nangue Tasse et al., 2020]—defined as follows:

Definition 7 *We define goal-reaching tasks as tasks in a deterministic environment, where the rewards across tasks differ only at terminal transitions (transitions into absorbing states), and there exists an optimal policy that always reaches a terminal transition.*

Examples of such goal-reaching tasks are discounted tasks ($\gamma \in [0, 1)$) with zero non-terminal rewards—such as the bin-packing grid-world examples—and undiscounted tasks ($\gamma = 1$) with strictly negative non-terminal rewards [van Niekerk et al., 2019; Nangue Tasse et al., 2020]—such as the four-rooms example.

4.1 Theory for World Value Function

Let $M = (\mathcal{S}, \mathcal{A}, P, R_M, \gamma)$ be a given task from some task space $\mathcal{M}$. We first define the internal goal space $\mathcal{G} \subseteq \mathcal{S}$ of an agent as all states where it experiences a terminal transition. Unlike other goal-conditioned approaches where the goal state to achieve in each episode is given by the environment, here the agent chooses the goal state it wishes to achieve. The agent's aim now is to simultaneously solve the current task, while also learning how to achieve its own internal goals. To do so, the agent can define its own goal-conditioned reward function $\bar{R}_M$, which extends R_M to penalise itself for achieving goals it did not intend to:

Definition 8 *For a task M with reward function R_M bounded by $[R_{MIN}, R_{MAX}] \subset \mathbb{R}$ for non-terminal transitions, the extended reward function $\bar{R}_M : \mathcal{S} \times \mathcal{G} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is given by*

$$\bar{R}_M(s, g, a, s') \coloneqq \begin{cases} \bar{R}_{MIN} & \textit{if } s' \textit{ is absorbing and } s \neq g, \\ R_M(s, a, s') & \textit{otherwise}, \end{cases} \tag{1}$$

where $\bar{R}_{MIN} \leq \min\{R_{MIN}, (R_{MIN} - R_{MAX})D\}$ *and* D *is the highest expected number of timesteps to reach* $\mathcal{G}$.[5]

This new reward function represents the idea that if an agent terminates in a state ($s \in \mathcal{G}$) that is not the goal state it was trying to reach ($s \neq g$), it should receive the smallest reward possible ($\bar{R}_{\text{MIN}}$). Intuitively, the penalty $\bar{R}_{\text{MIN}}$ adds one bit of information to the agent's rewards, and we will later prove this is sufficient for the agent to learn the value of achieving its internal goals in a goal-reaching task. The agent must now compute a *world Markov policy* $\bar{\pi} : \mathcal{S} \times \mathcal{G} \to Pr(\mathcal{A})$ that optimally reaches any reachable goal. A given world policy $\bar{\pi}$ is characterised by a *world value function* defined as follows:

Definition 9 *For a task M, the world value function* $\bar{Q}^{\bar{\pi}}_M : \mathcal{S} \times \mathcal{G} \times \mathcal{A} \to \mathbb{R}$ *is given by*

$$\bar{Q}^{\bar{\pi}}_M(s, g, a) \coloneqq \mathbb{E}^{\bar{\pi}}_s \left[\sum_{t=0}^{\infty} \gamma^t \bar{R}_M(s_t, g, a_t, s_{t+1}) \middle| s_0 = s, a_0 = a \right]. \tag{2}$$

This specifies the expected return obtained by executing a from s, and thereafter following $\bar{\pi}$ to reach g. Since for each g these WVFs are equivalent to standard value functions, it follows that all standard results on standard value functions also hold for WVFs by extension. This can be shown by simply noting that each $g \in \mathcal{G}$ corresponds to a well defined MDP $M_g := (\mathcal{S}, \mathcal{A}, P, R_g, \gamma)$ with reward function $R_{M_g}(s, a, s') := \bar{R}_M(s, g, a, s')$. In particular, we have that there exists an optimal deterministic world policy $\bar{\pi}^*$, and unique optimal WVF $\bar{Q}^*_M$, such that $\bar{Q}^*_M(s, g, a) := \bar{Q}^{\bar{\pi}^*}_M(s, g, a) = \max_{\bar{\pi}} \bar{Q}^{\bar{\pi}}_M(s, g)$. Similarly to standard value functions, we want to still be able to extract the standard policy that solves the current task by acting greedily over action-values: $\pi^*_M(s) \in \arg\max_a \max_g \bar{Q}^*_M(s, g, a)$. Theorem 1 shows that this is possible, at least in the case of goal-reaching tasks.

Theorem 1 *Let* R_M, $\bar{R}_M$, Q^*_M *and* $\bar{Q}^*_M$ *be the standard reward function, extended reward function, optimal value function and optimal world value function respectively for a goal-reaching task* M. *Then for all* (s, a) *in* $\mathcal{S} \times \mathcal{A}$, *we have*

$$R_M(s, a, s') = \max_{g \in \mathcal{G}} \bar{R}_M(s, g, a, s') \textit{ and } Q^*_M(s, a) = \max_{g \in \mathcal{G}} \bar{Q}^*_M(s, g, a).$$

■

Theorem 1 is critical: despite changing the standard RL objective (the standard reward function, value function, and policy), an agent can always recover these original objects, and can also solve the current task by simply maximising over goals. WVFs are therefore a strict generalisation of regular value functions for this type of task. However, we note that this is not necessarily true for all other types of tasks. For example, consider a stochastic

[5] $D = \max_{s \in \mathcal{S} \setminus \mathcal{G}, g \in \mathcal{G}} \max_{\pi} \mathbb{E}\left[T(g|\pi, s)\right]$, where T is the number of timesteps required to first reach g from s under a proper policy π.

Actions	$P(g_1)$	$P(g_2)$	$P(g_3)$	$\bar{Q}^*_M(g_1)$	$\bar{Q}^*_M(g_2)$	$\bar{Q}^*_M(g_3)$	$\max_g \bar{Q}^*_M(g)$	Q^*
a_1	0.6	0.2	0.2	0.6	0.2	0	0.6	0.8
a_2	0.2	0.6	0.2	0.2	0.6	0	0.6	0.8
a_3	0.5	0.5	0.0	0.5	0.5	0	0.5	1

Table 2: Counter example in a stochastic MDP. We omit s and a in the notation for clarity.

MDP with three goal states g_1, g_2, g_3, an initial state w with three actions a_1, a_2, a_3, non-terminal rewards of 0, and goal rewards of $1, 1, 0$ respectively. Table 2 shows the transition probabilities, WVF at each goal, the maximisation over the WVF, and the regular value function for each action in state W. Clearly for this task, acting greedily over the WVF is suboptimal compared to the standard value function.

Having established WVFs as a task-specific general value function, we next prove in Theorem 2 that those of goal-reaching tasks have mastery—that is, they learn the value of reaching all achievable goal states in the world. We define mastery as follows:

Definition 10 *Let $\bar{Q}^*_M$ be the optimal world value function for a task M in $\mathcal{M}$. Then $\bar{Q}^*_M$ has mastery if for all $g \in \mathcal{G}$ reachable from $s \in \mathcal{S} \setminus \mathcal{G}$, there exists an optimal world policy $\bar{\pi}^*(s, g) \in \arg\max_{a \in \mathcal{A}} \bar{Q}^*_M(s, g, a)$ that maximises the probability of reaching g from s.*

Theorem 2 *For all goal-reaching tasks M, $\bar{Q}^*_M$ has mastery.* ■

These results are important as they show that a WVF encodes the optimal policy for the current task (Theorem 1)—meaning that we can henceforth focus only WVFs—while also encoding the optimal policy for achieving any goal in the environment and the value of achieving said goals in the current task (Theorem 2)—which will later be useful for zero-shot composition.

4.2 Learning WVFs

Having formally established WVFs, we now propose ways of learning them using model-free algorithms. We note that while learning methods in the large body of work on goal-conditioned value functions can be used to learn WVFs, they require the reward functions or desired goal states to be available to the learning methods [Kaelbling, 1993; Veeriah et al., 2018; Andrychowicz et al., 2017; Colas et al., 2019; Foster & Dayan, 2002; Mirowski et al., 2017; Moore et al., 1999]. Here we wish to learn WVFs only with rewards per state-action obtained by interacting with the environment. Furthermore, the agent should also learn the best goal states to achieve for the current task. We hence extend DG learning in the tabular case [Kaelbling, 1993] and function approximation case [Veeriah et al., 2018] to learn WVFs with task-dependent reward functions.

4.2.1 Tabular Case

DG learning uses a modified version of Q-learning to learn goal-conditioned action-value functions from a single stream of experience. The algorithm differs from standard Q-learning in several ways: it keeps track of the set of terminating states seen so far (line 6), and at each timestep updates the goal-conditioned value function with respect to both the current state and action, as well as all goals encountered so far (lines 7–10). We extend the algorithm to use the definition of the extended rewards (Equation 1) for the task-dependent rewards (line 8). The full pseudocode is shown in Algorithm 1.

Algorithm 1: Q-learning for WVFs

```
   Initialise: WVF $\bar{Q}(s, g, a) = 0$, goal buffer $\mathcal{G} = \{\emptyset\}$
1  foreach episode do
2      Observe initial state $s \in \mathcal{S}$ and sample a goal $g \in \mathcal{G}$
3      while episode is not done do
4          $a \leftarrow \begin{cases} \arg\max_{a \in \mathcal{A}} \bar{Q}(s, g, a) & \text{w.p. } 1 - \varepsilon \\ \text{a random action} & \text{w.p. } \varepsilon \end{cases}$
5          Execute $a$, observe reward $r$ and next state $s'$
6          if $s'$ is absorbing then $\mathcal{G} \leftarrow \mathcal{G} \cup \{s\}$
7          for $g' \in \mathcal{G}$ do
8              $\bar{r} \leftarrow \bar{R}_{\text{MIN}}$ if $g' \neq s$ and $s \in \mathcal{G}$ else $r$
9              $\bar{Q}(s, g', a) \xleftarrow{\alpha} \left(\bar{r} + \gamma \max_{a'} \bar{Q}(s', g', a')\right) - \bar{Q}(s, g', a)$
10         $s \leftarrow s'$
```

4.2.2 Function Approximation Case

Veeriah et al. [2018] modify deep Q-learning in a similar way to Kaelbling [1993] to learn goal-conditioned values. Their approach differs from standard deep Q-learning in that they use universal value function approximators (UVFAs) [Schaul et al., 2015], which take both states and goals as input. Additionally, when a terminal state is encountered, it is added to the collection of goals seen so far, and when learning updates occur, these goals are sampled randomly from a goal buffer. Similarly to our extension of DG-learning in the tabular case, we also extend this approach to use the definition of extended rewards (Equation 1) for the task-dependent rewards. The full pseudocode is shown in Algorithm 2 in the appendix, where we note that our approach is agnostic to the learning algorithm used.

4.3 Experiments

In this section, we empirically verify that WVFs can be learned with a single stream of experience using standard model-free algorithms, and validate their properties described above.

4.3.1 Tabular Case

Consider the bin packing domain introduced in Example 1 where the non-terminal rewards (rewards for all non-terminal states) are 0 and the goal rewards (rewards for terminal states) range from 0 to 1. We use a discount factor of $\gamma = 0.95$.

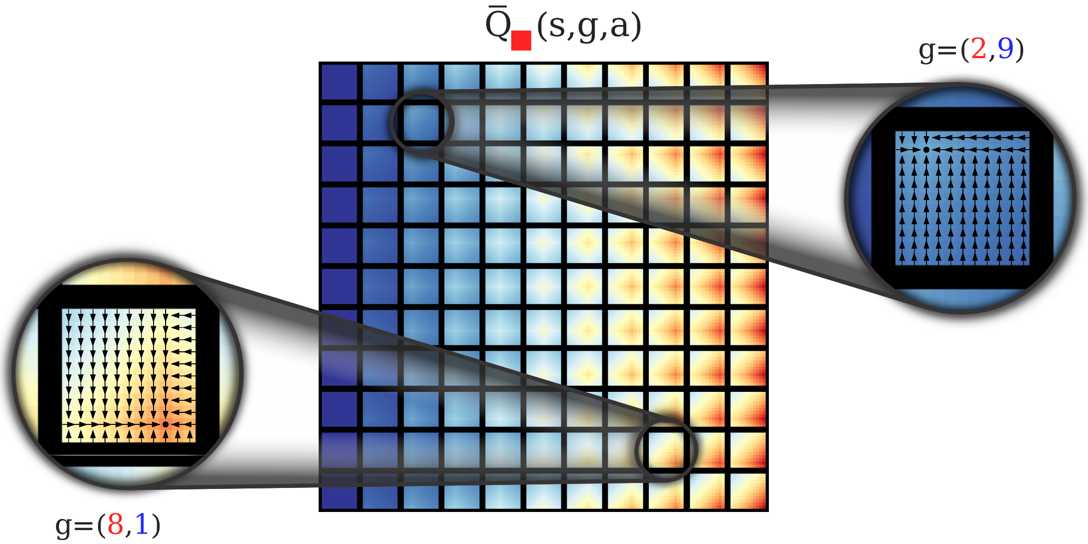


Figure 16: Learned WVF for the task of packing all the red objects into the bin. Each square shows the value of all states (s) with respect to the goal (g) at that position.

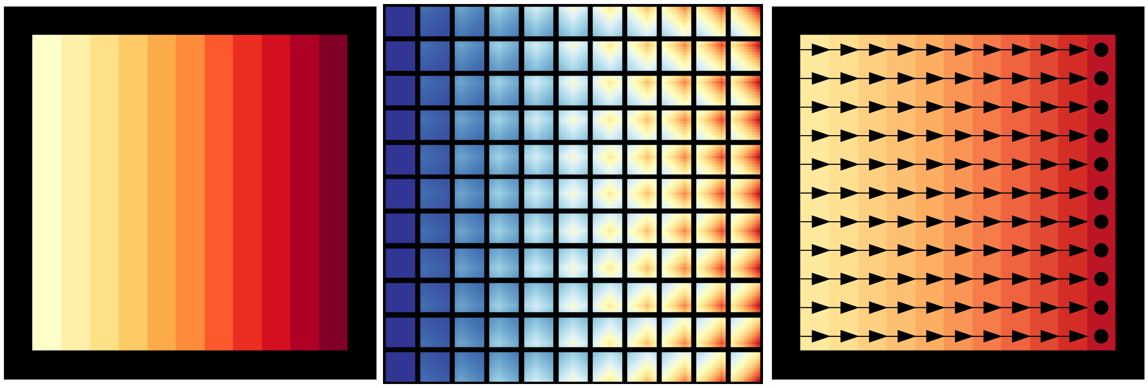
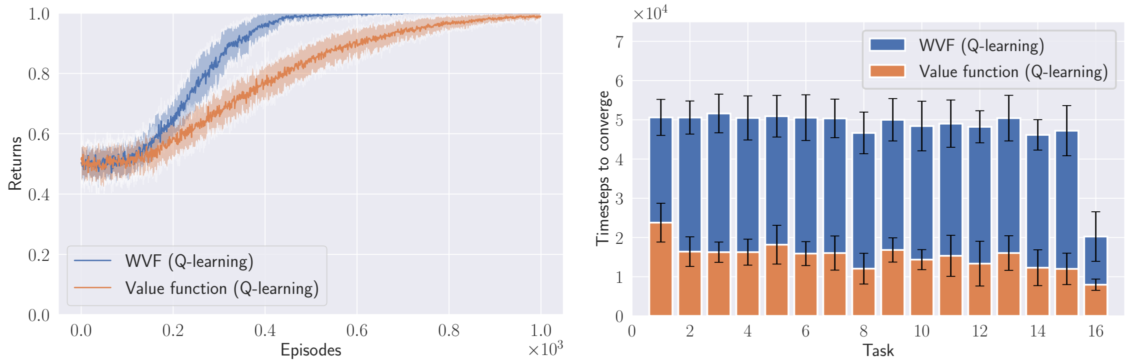


Figure 17: Illustrating the standard value function and policy (right) for the task of packing all the red objects into the bin (left), obtained by maximising over the goal values of the learned WVF $\bar{Q}^*_{\blacksquare}$ (middle).

We use Algorithm 1 to train an agent on the task 🟥, in which the robot must pack all the red objects into the bin. Figure 16 shows the learned WVF. The figure is generated by first plotting the value functions for all goal states, then displaying each of them at their respective position in the gridworld representation of the domain. We can observe from the value gradients of the plots that the learned WVF does indeed have mastery as it encodes how to achieve all desirable goals, demonstrating the results proven in Theorem 2. Notice how for the goal states corresponding to no red object in the bin (the leftmost column of the plot), the WVF has zero values everywhere since the agent receives no reward at those goals. As we proved in Theorem 1, we can also maximise over goals to obtain the standard state-value function and policy as shown in Figures 17.



(a) Returns during training of the WVF and standard value function for the 🟥 task. Returns are calculated by greedy evaluation at the end of each episode. Mean and standard deviation over 25 random seeds are shown.

(b) Number of samples required to learn optimal WVFs and standard value functions for the 16 tasks in Figure 14. Error bars represent standard deviations over 25 random seeds.

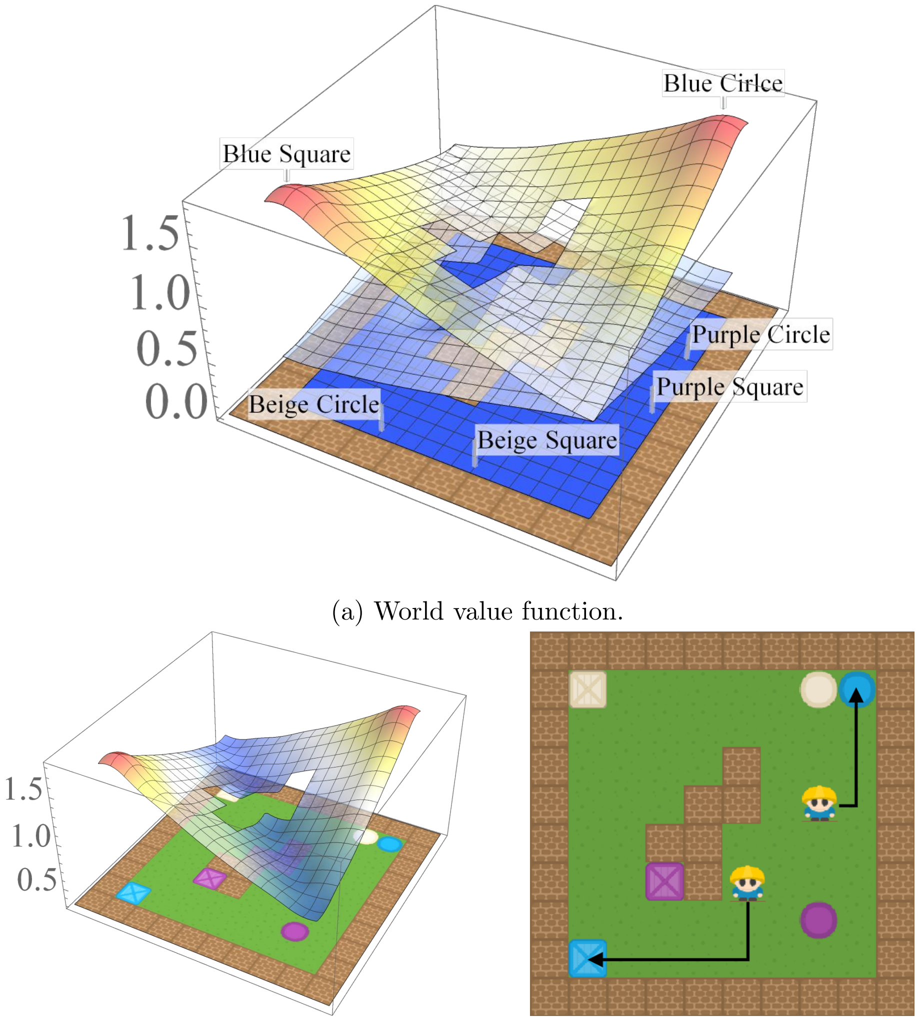
Figure 18: Sample complexity for training WVFs and standard value functions.

By learning WVFs, an agent learns a large number of diverse solutions to a single task. However, the upfront cost of learning is likely to be higher since we must learn not only the single value function, but rather the value function with respect to every goal. We investigate the sample complexity of learning WVFs (using Algorithm 1) and learning standard value functions (using standard Q-learning) in Figure 18. As expected, we observe that the number of samples required to learn optimal WVFs is greater than that for learning optimal standard value functions (Figure 18b). Interestingly, we also observe that it is more sample efficient to learn WVFs if we only care about the performance of the resulting policy instead of the optimality of the action-values—despite the fact that WVFs have an additional dimension that must be learned (Figure 18a). We conjecture that this is due to the induced goal-directed exploration of Algorithm 1, similarly to the results obtained by Kaelbling [1993].

We show in Section 5 how we can leverage the WVFs to improve transfer in a multi-task setting, which amortises the upfront cost over multiple tasks.

4.3.2 Function Approximation Case

Finally, we demonstrate that our modified learning method can also be used to learn WVFs in high-dimensional domains where function approximation is required. Consider the same video game environment as van Niekerk et al. [2019], where an agent must navigate a 2D world and collect objects of different shapes and colours. The state space is an 84×84 RGB image, and the agent is able to move in any of the four cardinal directions. The agent also possesses a `pick-up` action, which allows it to collect an object when standing on top of it. There are two shapes (squares and circles) and three colours (blue, beige and purple) for a total of six unique objects. The positions of the agent and objects are randomised at the start of each episode. Finally, the rewards are 2 for picking-up a desired object and 0 everywhere else, and the discount factor is $\gamma = 0.95$.



(a) World value function.

(b) Standard value function obtained by maximising over goals on the WVF.

(c) Sample trajectories of the agent solving the task, obtained by acting greedily over the learned WVF.

Figure 19: Learned WVF for the task of collecting blue objects in the video game domain. The WVF shows the value function with respect to each goal plotted on the same axis.

We train the agent on the *Blue* task, which requires the agent to navigate to blue objects and collect them. See appendix B for all the training hyperparameters and neural network architecture. Figure 19 shows the learned WVF and the result of maximising its values over goals. To generate the plot, we place the agent at every location and compute the maximum output of the network over all actions for each goal. We then interpolate between the points to smooth the graph. We observe that the learned WVF encodes how to achieve all desirable goals in the environment, even when using function approximation.

5. Composing World Value Functions

In the previous sections, we formalised the logical composition of tasks, showed how standard value functions are insufficient to solve them zero-shot, and introduced WVFs as a richer type of value function that may encode sufficient information about learned tasks to solve new ones without further learning. In this section we formally show that WVFs of goal-reaching tasks are indeed sufficient to solve the logical composition of tasks zero-shot.[6] We do this by first formalising their composition under the relevant algebraic structures, just as was done with task compositions. This gives us mathematical tools which can be used to formally prove zero-shot composition and further explore some additional properties of the established structures.

5.1 WVF Lattice

Similarly to how we formalised the disjunction and conjunction of tasks using the lattice algebraic structure, we now formalise the disjunction and conjunction of WVFs. Since WVFs are real valued functions, a natural partial order on them is pointwise $\leq$ (the usual $\leq$ on $\mathbb{R}$). We state the resulting poset formally as follows:

Proposition 6 *Let $(\mathcal{M}, \vee, \wedge)$ be a lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then $(\bar{\mathcal{Q}}^*, \leq)$ is a partially ordered set with the relation $\leq$ given by*

$$\bar{Q}^*_{M_1} \leq \bar{Q}^*_{M_2} \text{ if } \bar{Q}^*_{M_1}(s,g,a) \leq \bar{Q}^*_{M_2}(s,g,a) \text{ for all } (s,g,a) \in \mathcal{S} \times \mathcal{G} \times \mathcal{A}.$$

■

Given that all goal-reaching tasks share the same non-terminal rewards, the WVF for each goal state is partially ordered based only on the terminal reward at that goal state (since the terminal reward at the other goal states is set to R_{MIN} by the extended reward function). Hence, since $(\mathcal{M}, \vee, \wedge)$ is a lattice based on the usual $\leq$ over rewards, every pair of optimal WVFs has a supremum and an infimum in $\bar{\mathcal{Q}}^*$ respectively given by simply applying pointwise *sup* and *inf*:

Proposition 7 *Let $(\mathcal{M}, \vee, \wedge)$ be a lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then for all $M_1, M_2 \in \mathcal{M}$,*
(i) $\sup\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\} = \bar{Q}^*_{M_1 \vee M_2} \in \bar{\mathcal{Q}}^*$, *(ii)* $\inf\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\} = \bar{Q}^*_{M_1 \wedge M_2} \in \bar{\mathcal{Q}}^*$. ■

This means that the set of WVFs $\bar{\mathcal{Q}}^*$ forms a lattice $(\bar{\mathcal{Q}}^*, \vee, \wedge)$ with $\vee$ and $\wedge$ given by $\bar{Q}^*_{M_1} \vee \bar{Q}^*_{M_2} := \sup\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\}$ and $\bar{Q}^*_{M_1} \wedge \bar{Q}^*_{M_2} := \inf\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\}$ respectively. We define these formally as follows:

Definition 11 *Let $(\mathcal{M}, \vee, \wedge)$ be a lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. The join $\vee : \bar{\mathcal{Q}}^* \times \bar{\mathcal{Q}}^* \to \bar{\mathcal{Q}}^*$ and meet $\wedge : \bar{\mathcal{Q}}^* \times \bar{\mathcal{Q}}^* \to \bar{\mathcal{Q}}^*$ operators are given by the mappings $(\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}) \mapsto \bar{Q}^*_{M_1} \vee \bar{Q}^*_{M_2}$ and $(\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}) \mapsto \bar{Q}^*_{M_1} \wedge \bar{Q}^*_{M_2}$ respectively, where*

[6] While we focus on the WVFs of goal-reaching tasks (Definition 7), we leave our proofs as general as possible, making it clear which parts actually make use of this constraint.

$$\begin{aligned}\bar{Q}^*_{M_2} \vee \bar{Q}^*_{M_2} :\ & \mathcal{S} \times \mathcal{G} \times \mathcal{A} \to \mathbb{R} \\ & (s, g, a) \mapsto \sup\{\bar{Q}^*_{M_1}(s, g, a), \bar{Q}^*_{M_2}(s, g, a)\},\end{aligned}$$

$$\begin{aligned}\bar{Q}^*_{M_1} \wedge \bar{Q}^*_{M_2} :\ & \mathcal{S} \times \mathcal{G} \times \mathcal{A} \to \mathbb{R} \\ & (s, g, a) \mapsto \inf\{\bar{Q}^*_{M_1}(s, g, a), \bar{Q}^*_{M_2}(s, g, a)\}.\end{aligned}$$

In fact $(\bar{\mathcal{Q}}^*, \vee, \wedge)$ forms a distributive lattice, following from the distributivity of *inf* and *sup* on real numbers. We state this as follows:

Proposition 8 *Let $(\mathcal{M}, \vee, \wedge)$ be a lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then $(\bar{\mathcal{Q}}^*, \vee, \wedge)$ is a distributive lattice.* ■

Given a non-empty finite set $\mathcal{O}$ of lower bounded subsets of WVFs $\mathcal{N} \subset \bar{\mathcal{Q}}^*$, the WVF lattice $(\bar{\mathcal{Q}}^*, \vee, \wedge)$ gives us a principled way of specifying the disjunction of conjunctions:

$$\bigvee_{\mathcal{N} \in \mathcal{O}} \left(\bigwedge_{N \in \mathcal{N}} N \right)(s, g, a) = \sup_{\mathcal{N} \in \mathcal{O}} \left(\inf_{N \in \mathcal{N}} \bar{Q}^*_N(s, g, a) \right).$$

Similarly, given a non-empty finite set $\mathcal{O}$ of upper bounded subsets of WVFs $\mathcal{N} \subset \bar{\mathcal{Q}}^*$, the conjunction of disjunctions is given by

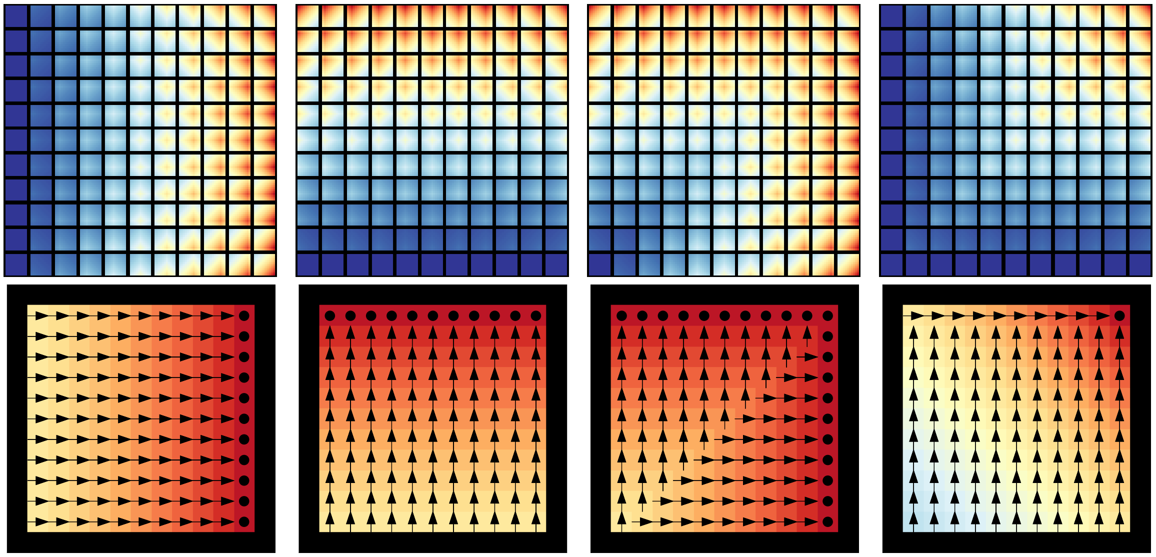
$$\bigwedge_{\mathcal{N} \in \mathcal{O}} \left(\bigvee_{N \in \mathcal{N}} N \right)(s, g, a) = \inf_{\mathcal{N} \in \mathcal{O}} \left(\sup_{N \in \mathcal{N}} \bar{Q}^*_N(s, g, a) \right).$$

Having established a lattice algebra over tasks and WVFs, we show that there exists an equivalence between the two. As a result, if we can specify a task under the lattice algebra, we can immediately obtain the optimal WVF for the task. This homomorphism follows from the fact that $\sup\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\} = \bar{Q}^*_{M_1 \vee M_2}$ and $\inf\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\} = \bar{Q}^*_{M_1 \wedge M_2}$ in Proposition 7.

Theorem 3 *Let $(\mathcal{M}, \vee, \wedge)$ be a lattice of goal-reaching tasks, and let $(\bar{\mathcal{Q}}^*, \vee, \wedge)$ be the corresponding lattice of WVFs. Let $H : \mathcal{M} \to \bar{\mathcal{Q}}^*$ be any map from $\mathcal{M}$ to $\bar{\mathcal{Q}}^*$ such that $H(M) = \bar{Q}^*_M$ for all M in $\mathcal{M}$. Then H is a homomorphism.* ■

Experiment 5.1 *Consider the bin packing domain introduced in Example 1 where non-terminal rewards are 0 and the goal rewards range from 0 to 1. The discounting is $\gamma = 0.95$.*

We train an agent on the tasks 🟥 and 🟦, in which the robot must respectively pack all the red and blue objects into the bin. Figure 20 shows the learned WVFs $\bar{Q}^*_{\text{🟥}}$ and $\bar{Q}^*_{\text{🟦}}$ together with their disjunction and conjunction. Notice how the composition of WVFs exhibits the same semantics as that of the corresponding rewards (Figure 6). This highlights the homomorphism proven in Theorem 3, showing the structural similarity between the task space and value function space. Finally, Figure 21 shows the Hasse diagram of the WVF sub-lattice generated by all combinations of disjunction and conjunction of $\bar{Q}^*_{\text{🟥}}$ and $\bar{Q}^*_{\text{🟦}}$.



(a) $\bar{Q}^*_{\text{🟥}}$

(b) $\bar{Q}^*_{\text{🟦}}$

(c) $\bar{Q}^*_{\text{🟥}} \vee \bar{Q}^*_{\text{🟦}}$

(d) $\bar{Q}^*_{\text{🟥}} \wedge \bar{Q}^*_{\text{🟦}}$

Figure 20: Showing the disjunction and conjunction of the learned WVFs ($\bar{Q}^*_{\text{🟥}}, \bar{Q}^*_{\text{🟦}}$) in the bin packing domain. The top row shows the WVFs, and the bottom one shows the value functions and policies obtained by acting greedily over their values per goal.

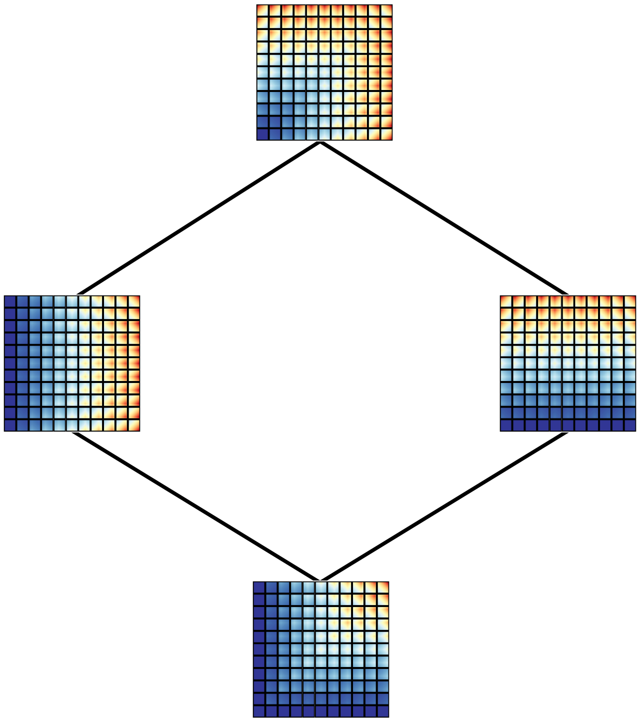


Figure 21: Hasse diagram of the WVF lattice generated by $\bar{Q}^*_{\text{🟥}}$ and $\bar{Q}^*_{\text{🟦}}$.

5.2 De Morgan WVF Algebra

Having formalised the meaning of disjunction and conjunction, we next formalise the meaning of the negation of WVFs. A De Morgan algebra enables us to define this by adding the minimal required properties that encapsulate the desired semantics of negation. Specifically, we assume that the set of tasks is bounded and that the WVF of each task has mastery. We then define the negation of an WVF as follows:

Definition 12 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a De Morgan lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Define $\bar{Q}^*_{INF}, \bar{Q}^*_{SUP} \in \bar{\mathcal{Q}}^*$ to be the optimal $\bar{Q}$-functions for the task bounds $\mathcal{M}_{INF}, \mathcal{M}_{SUP} \in \mathcal{M}$. Then the negation operator is given by*

$$\begin{aligned} \neg :\ & \bar{\mathcal{Q}}^* \to \bar{\mathcal{Q}}^* \\ & \bar{Q}^* \mapsto \neg\bar{Q}^*, \text{ where } \neg\bar{Q}^* : \ \mathcal{S} \times \mathcal{G} \times \mathcal{A} \to \mathbb{R} \\ & \qquad\qquad\qquad\qquad (s,g,a) \mapsto \left(\bar{Q}^*_{SUP}(s,g,a) + \bar{Q}^*_{INF}(s,g,a)\right) - \bar{Q}^*(s,g,a). \end{aligned}$$

We now show that the negation of $\bar{Q}^* \in \bar{\mathcal{Q}}^*$ is indeed in $\bar{\mathcal{Q}}^*$ for WVFs with mastery:

Proposition 9 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a De Morgan lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then $\neg\bar{Q}^*_M = \bar{Q}^*_{\neg M} \in \bar{\mathcal{Q}}^*$ for all $M \in \mathcal{M}$.* ■

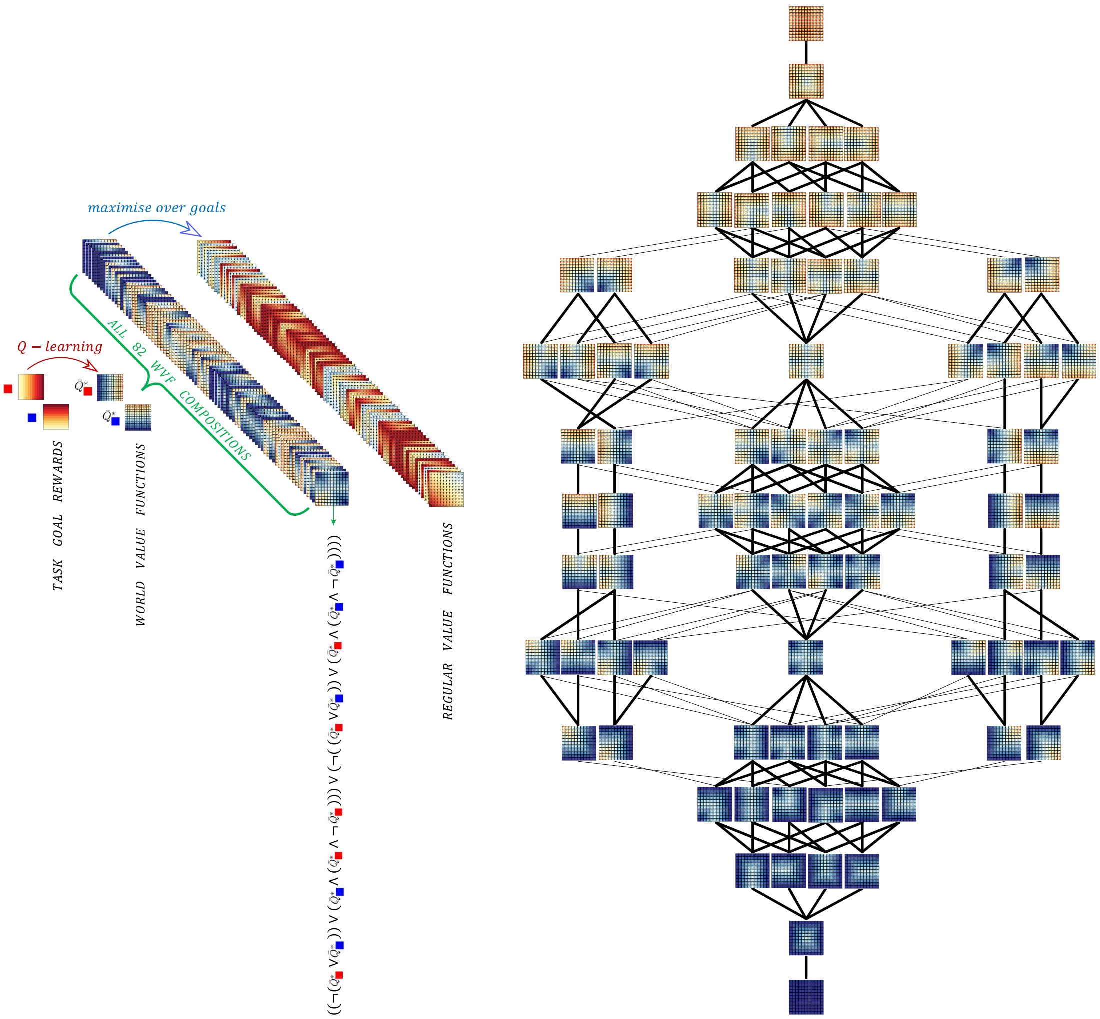
We now formalise the interaction of the negation of WVFs with the conjunction and disjunction of WVFs as follows:

Proposition 10 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a De Morgan lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then $(\bar{\mathcal{Q}}^*, \vee, \wedge, \neg, \bar{Q}^*_{SUP}, \bar{Q}^*_{INF})$ is a De Morgan algebra.* ■

Having established a De Morgan algebra over tasks and WVFs, we show that there exists an equivalence between the two. As a result, if we can specify a task under the De Morgan algebra, we can immediately derive the optimal WVF for the task.

Theorem 4 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a De Morgan lattice of goal-reaching tasks, and let $(\bar{\mathcal{Q}}^*, \vee, \wedge, \neg, \bar{Q}^*_{SUP}, \bar{Q}^*_{INF})$ be the corresponding De Morgan lattice over WVFs. Let $H\ : \mathcal{M} \to \bar{\mathcal{Q}}^*$ be any map from $\mathcal{M}$ to $\bar{\mathcal{Q}}^*$ such that $H(M) = \bar{Q}^*_M$ for all M in $\mathcal{M}$. Then H is a homomorphism.* ■

Experiment 5.2 *Consider the bin packing domain introduced in Example 1. We use the WVFs learned in Experiment 5.1 for the tasks* 🟥 and 🟦*. Figure 22 shows the De Morgan sub-lattice generated by all combinations of their disjunction, conjunction, and negation. Figure 23 shows sample WVF compositions, and the optimal policies and optimal value functions obtained from them by maximising over goals. For example, their negations $\neg\bar{Q}^*_{🟥}$ and $\neg\bar{Q}^*_{🟦}$, result in policies in which the agent determines how to unpack all the red and blue objects from the bin, without further learning. This shows that the negation defined above does indeed have the expected semantics. The figure also shows that arbitrary disjunction, conjunction, and negation of WVFs also produce WVFs with the desired semantics.*



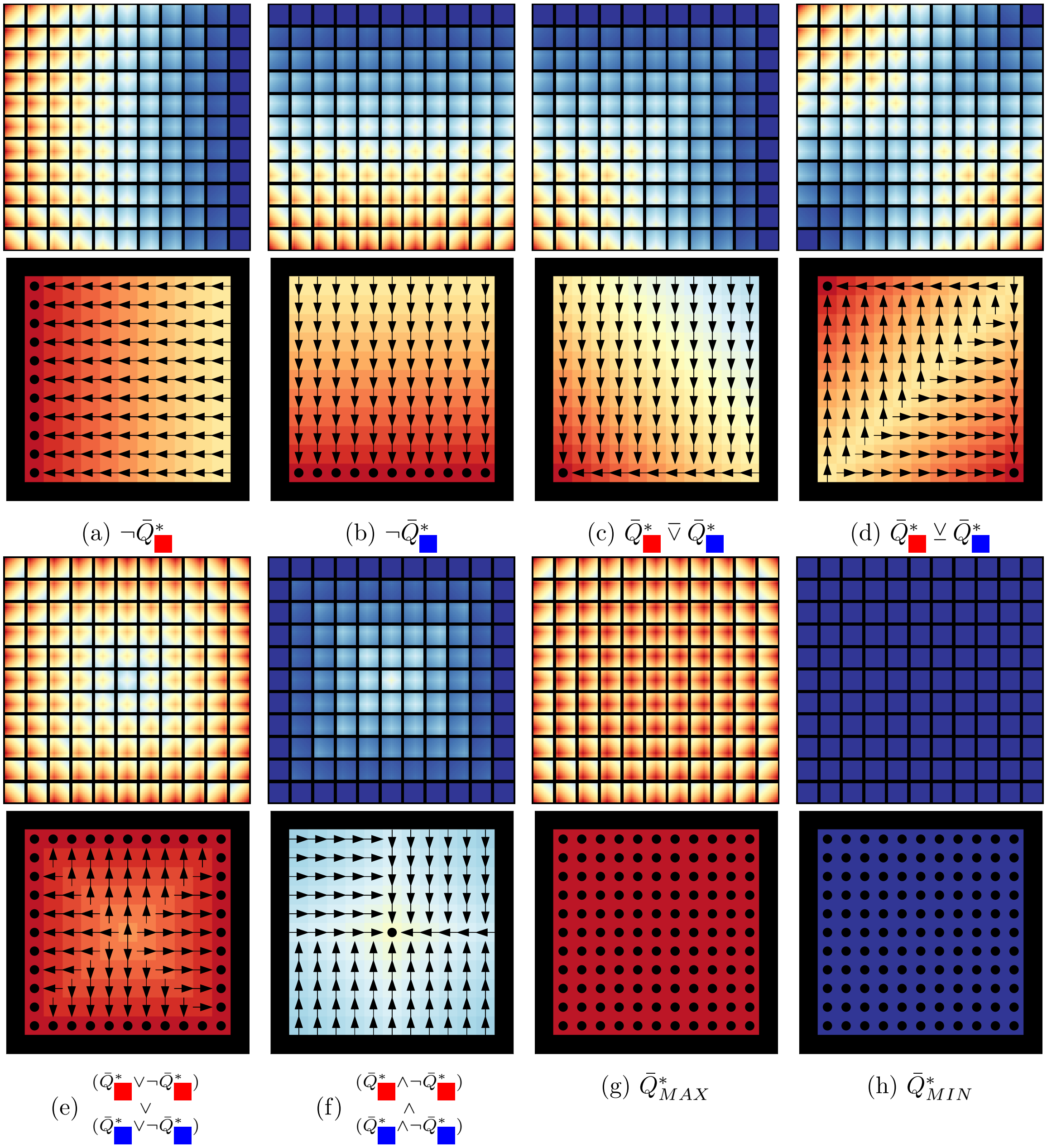
(a) Zero-shot WVF compositions

(b) Hasse diagram

Figure 22: Illustration of the De Morgan sub-lattice generated by composing $\bar{Q}^*_{\color{red}\blacksquare}$ and $\bar{Q}^*_{\color{blue}\blacksquare}$. (a) Illustrates how after learning the WVF for only 2 tasks, we can compose them to solve a combinatorially large number of unique tasks (82 tasks in this case) without further learning. (b) Shows the relationship between all the WVFs generated by composing the WVF of the red and blue tasks.

5.3 Boolean WVF Algebra

While the De Morgan WVF algebra allows us to specify arbitrary disjunction, conjunction, and negation of WVFs, it does not in general represent the full desired properties of logic. In particular, the WVF compositions do not always satisfy the laws of the excluded middle and of non-contradiction. This is because the De Morgan WVF algebra allows for WVFs obtained from non-Boolean tasks—that is, tasks with non-binary rewards. In this section, we show that by restricting the WVFs to those of Boolean tasks, we obtain the full logic on WVFs.



(a) $\neg\bar{Q}^*_{\text{🟥}}$

(b) $\neg\bar{Q}^*_{\text{🟦}}$

(c) $\bar{Q}^*_{\text{🟥}} \mathbin{\bar{\vee}} \bar{Q}^*_{\text{🟦}}$

(d) $\bar{Q}^*_{\text{🟥}} \mathbin{\underline{\vee}} \bar{Q}^*_{\text{🟦}}$

(e) $\begin{matrix}(\bar{Q}^*_{\text{🟥}} \vee \neg\bar{Q}^*_{\text{🟥}}) \\ \vee \\ (\bar{Q}^*_{\text{🟦}} \vee \neg\bar{Q}^*_{\text{🟦}})\end{matrix}$

(f) $\begin{matrix}(\bar{Q}^*_{\text{🟥}} \wedge \neg\bar{Q}^*_{\text{🟥}}) \\ \wedge \\ (\bar{Q}^*_{\text{🟦}} \wedge \neg\bar{Q}^*_{\text{🟦}})\end{matrix}$

(g) $\bar{Q}^*_{MAX}$

(h) $\bar{Q}^*_{MIN}$

Figure 23: The learned boundary WVFs ($\bar{Q}^*_{MAX}, \bar{Q}^*_{MIN}$) and the composition of learned WVFs ($\bar{Q}^*_{\text{🟥}}, \bar{Q}^*_{\text{🟦}}$). The top rows in each figure show the WVFs and the bottom ones show the value functions and policies obtained by acting greedily over goals.

Proposition 11 *Let* $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ *be a Boolean algebra of goal-reaching tasks, and let* $\bar{\mathcal{Q}}^*$ *be the set of optimal WVFs of tasks in* $\mathcal{M}$*.*
Then $(\bar{\mathcal{Q}}^*, \vee, \wedge, \neg, \bar{Q}^*_{SUP}, \bar{Q}^*_{INF})$ *is a Boolean WVF algebra.* ■

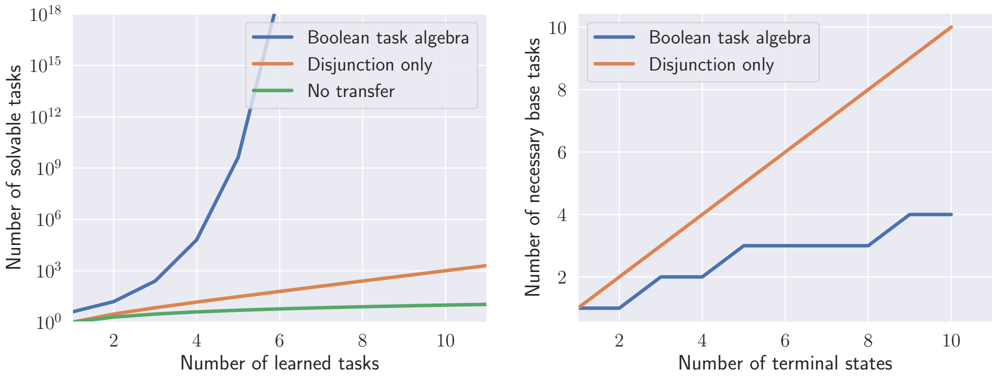
Having established a Boolean algebra over tasks and WVFs, we show that they are homomorphic. As a result, if we can specify a task under the Boolean algebra, we can immediately obtain the optimal WVF for the task.

Theorem 5 *Let* $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ *be a Boolean algebra of goal-reaching tasks, and let* $(\bar{\mathcal{Q}}^*, \vee, \wedge, \neg, \bar{Q}^*_{SUP}, \bar{Q}^*_{INF})$ *be the corresponding Boolean algebra of WVFs. Let* $H : \mathcal{M} \to \bar{\mathcal{Q}}^*$ *be any map from* $\mathcal{M}$ *to* $\bar{\mathcal{Q}}^*$ *such that* $H(M) = \bar{Q}^*_M$ *for all* M *in* $\mathcal{M}$*. Then* H *is a homomorphism.* ■

Combining Theorem 5 with the notion of base tasks and task specifications obtained in Section 3.4, we obtain the much desired super-exponential explosion of skills (see Figure 24).

Experiment 5.3 *Consider the simple bin packing domain used in Example 4 where the goal rewards are restricted to* $R_{MIN} = 0$ *or* $R_{MAX} = 1$*. The non-terminal rewards are* 0 *and the goal rewards are* 0 *for undesired terminal states and* 1 *for desired ones. The discounting used is* $\gamma = 0.95$*.*

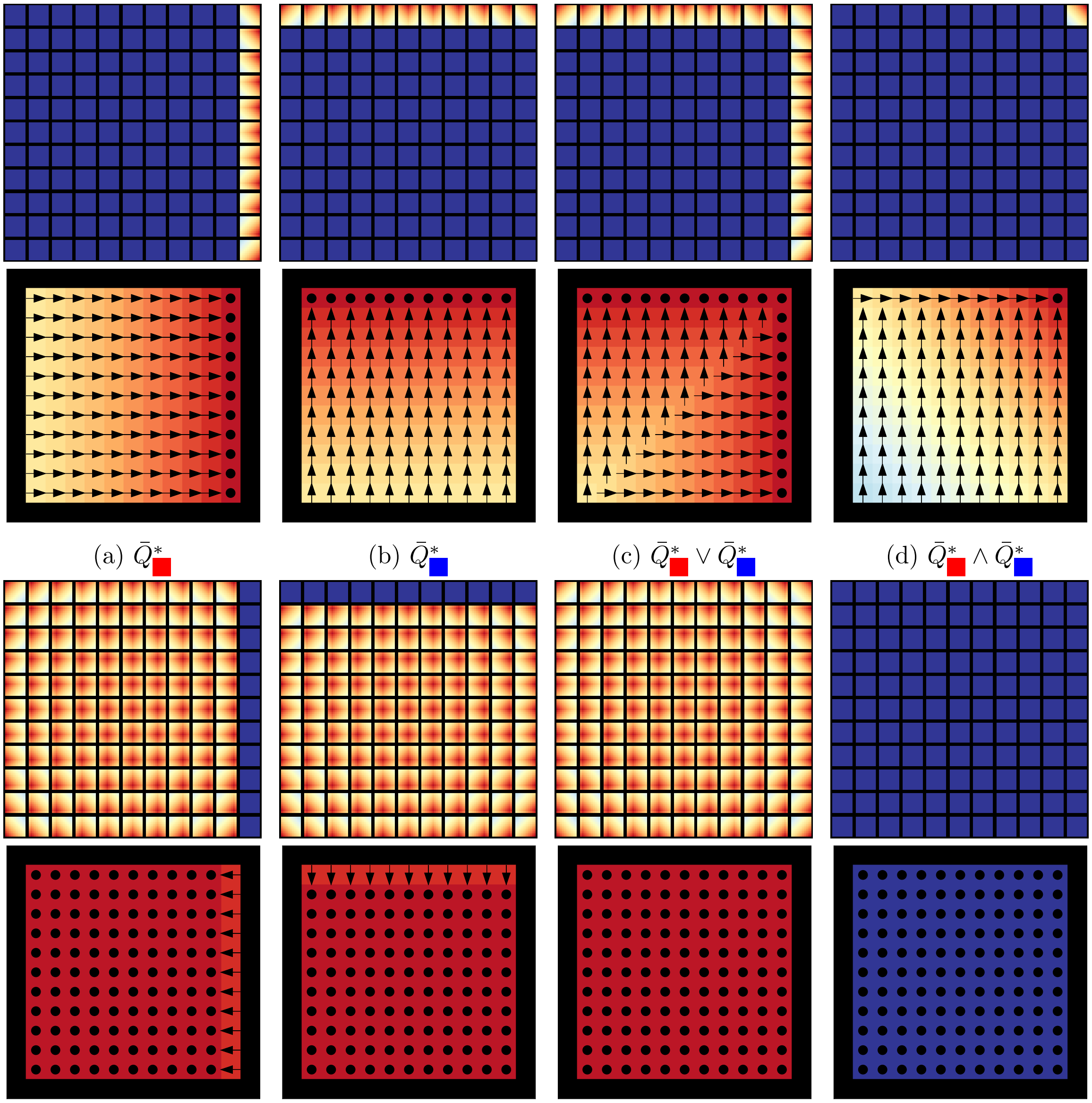
We train an agent on the tasks 🟥 *and* 🟦 *(as described in Example 4), producing the respective WVFs* $\bar{Q}^*_{🟥}$ *and* $\bar{Q}^*_{🟦}$*. We are now able to perform zero-shot composition of any logical combination of* 🟥 *and* 🟦*. This is demonstrated in Figure 25. As usual, we observe that the WVFs have the same structure as the rewards in the terminal set and encode how to achieve them. Notice how for the lower bound WVF (the result of meaningless composition), the optimal policy is to achieve any terminal state (Figure 25f). This is because all terminal states are equally undesirable (they have the lowest values). Figure 26 shows the Hasse diagram of the Boolean sub-lattice generated by* $\bar{Q}^*_{🟥}$ *and* $\bar{Q}^*_{🟦}$*.*



(a) Number of tasks that can be solved as a function of the number of existing tasks solved. Results are plotted on a log-scale.

(b) Number of base tasks that need to be solved to span all tasks as a function of number of terminal states. Results are plotted on a log-scale.

Figure 24: Results in comparison to the disjunctive composition of van Niekerk et al. [2019]. (a) The WVFs allow us to solve exponentially more tasks than the disjunctive approach without further learning. (b) With WVFs the number of base tasks required to solve all tasks is logarithmic in the number of achievable goals.



(a) $\bar{Q}^*_{🟥}$

(b) $\bar{Q}^*_{🟦}$

(c) $\bar{Q}^*_{🟥} \vee \bar{Q}^*_{🟦}$

(d) $\bar{Q}^*_{🟥} \wedge \bar{Q}^*_{🟦}$

(e) $\neg\bar{Q}^*_{🟥}$

(f) $\neg\bar{Q}^*_{🟦}$

(g) $\begin{matrix}(\bar{Q}^*_{🟥} \vee \neg\bar{Q}^*_{🟥}) \\ \vee \\ (\bar{Q}^*_{🟦} \vee \neg\bar{Q}^*_{🟦})\end{matrix}$

(h) $\begin{matrix}(\bar{Q}^*_{🟥} \wedge \neg\bar{Q}^*_{🟥}) \\ \wedge \\ (\bar{Q}^*_{🟦} \wedge \neg\bar{Q}^*_{🟦})\end{matrix}$

Figure 25: The Boolean composition (top row) of learned WVFs in the bin packing domain. The corresponding value functions and policies in the bottom row are obtained by acting greedily over their Q-values per goal.

Finally, we show that while optimal WVFs are more expensive to learn than optimal standard value functions, the trade-off with the compositional explosion of skills justifies this cost. We demonstrate this in the bin packing domain where we need to learn only 7 base tasks (Figure 27 shows the learned WVFs), as opposed to 121 for the disjunctive *case (since there are 121 goals). Figure 28 shows results in comparison to the disjunctive composition of van Niekerk et al. [2019].*

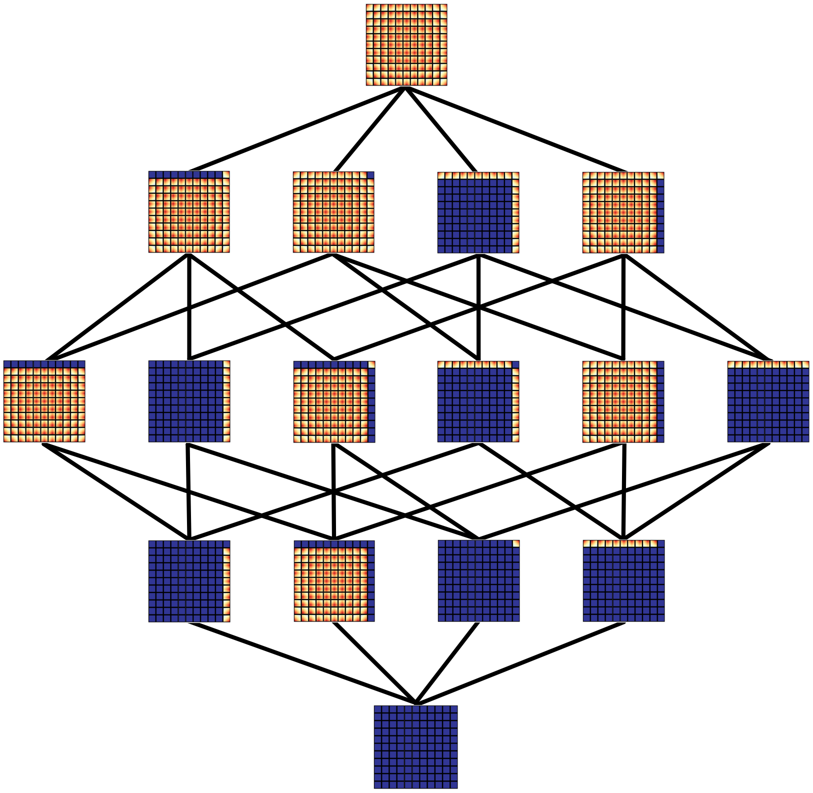


Figure 26: Hasse diagram of the Boolean sub-algebra generated by $\bar{Q}^*_{\text{🟥}}$ and $\bar{Q}^*_{\text{🟦}}$.

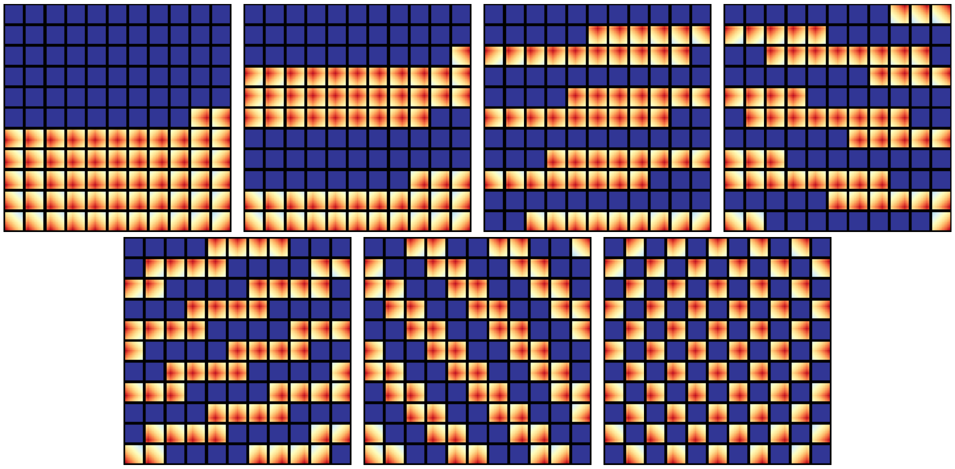


Figure 27: Base WVFs (generators) for the full Boolean algebra WVF space

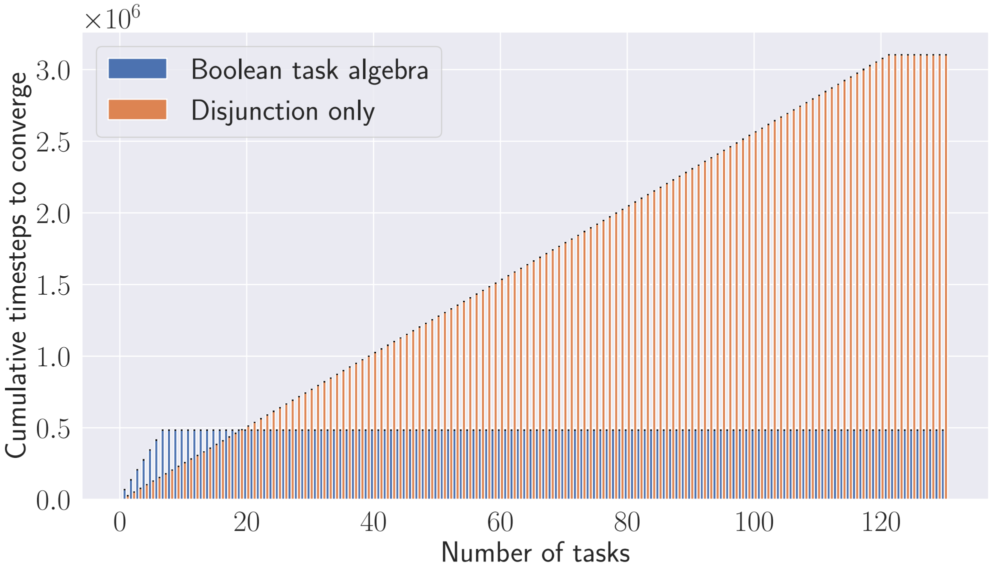


Figure 28: Cumulative number of samples required to solve tasks in the bin packing domain. Error bars represent standard deviations over 100 runs, but are small relative to the y-axis.

5.4 Constructing WVFs from Task Goals

In the previous section, we showed that the algebraic structures of tasks and WVFs are homomorphic. As a result, any task that is specified according to a task algebra can be immediately solved according to the WVF algebra. In this section we show that for the case of a Boolean WVF algebra, we can immediately construct optimal WVFs directly from a set of desired task goals. We do this by showing that the WVF and power set Boolean algebras are in fact isomorphic. Consider the mapping F between the set of WVFs $\bar{\mathcal{Q}}^*$ and the power-set $P(\mathcal{G})$, given by

$$\begin{aligned} F:\ & P(\mathcal{G}) \to \bar{\mathcal{Q}}^* \\ & \mathcal{H} \mapsto \bar{Q}^*_{\mathcal{H}},\ \text{where } \bar{Q}^*_{\mathcal{H}}:\ \mathcal{S}\times\mathcal{G}\times\mathcal{A} \to \mathbb{R} \\ & \qquad\qquad (s,g,a) \mapsto \begin{cases} \bar{Q}^*_{SUP}(s,g,a), & \text{if } g \in \mathcal{H} \\ \bar{Q}^*_{INF}(s,g,a), & \text{otherwise.} \end{cases} \end{aligned}$$

F is clearly an isomorphism between $P(\mathcal{G})$ and $\bar{\mathcal{Q}}^*$, since $g \in \mathcal{H}$ if and only if $\bar{Q}^*_{\mathcal{H}}(s,g',a) = \bar{Q}^*_{SUP}(s,g',a)$ for all $(s,a) \in \mathcal{S}\times\mathcal{A}$. This gives us the following important result.

Theorem 6 *Let $(\mathcal{M},\vee,\wedge,\neg,\mathcal{M}_{SUP},\mathcal{M}_{INF})$ be a Boolean algebra of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then the Boolean algebra on $\bar{\mathcal{Q}}^*$ is isomorphic to the Boolean algebra on $\mathcal{M}$.* ■

This illustrates how the base knowledge an agent needs to act optimally in an environment for any future task can be constructed rather than learned (if there is an efficient way of doing that construction). All that is required to be learned are two WVFs: the lower bound WVF $\bar{Q}^*_{INF}$ and upper bound WVF $\bar{Q}^*_{SUP}$. Note how this is true for any cardinality of $\mathcal{G}$. If $\mathcal{G}$ is finite and relatively small, then this can easily be done with a Boolean table.

Experiment 5.4 *Consider the Four Rooms domain described in Section 3.4, where an agent must navigate to the center of particular rooms. The non-terminal rewards are* -0.1 *and the goal rewards (rewards on the terminal set) are* -0.1 *for undesirable goals and* 2 *for desirable ones. The discount factor used is* $\gamma = 1$.

Just as in the Boolean task algebra, we can select a minimal set of base WVFs (generator WVFs) by assigning each goal a binary number, and then using the columns of the table to generate the WVFs. Since the terminal set is finite, we can do this assignment using a Boolean table. Similarly to Section 3.4, we label each goal by defining a well order over $\mathcal{G}$. *The number of base WVFs induced are then* $\lceil |\log_2 \mathcal{G}| \rceil = 2$ *since* $|\mathcal{G}| = 4$. *Table 3 illustrates how different well orders on* $\mathcal{G}$ *leads to different choices of base WVFs.*

Consider the well order on $\mathcal{G}$ *given by Table 3b. The base WVFs induced are* $\bar{Q}^*_L$ *and* $\bar{Q}^*_T$ *which are the WVFs of the tasks* M_L *and* M_T *respectively. Figure 29 illustrates these WVFs and some of their logical compositions. Note that the resulting optimal value function*

Goals	$\bar{Q}^*_D$	$\bar{Q}^*_R$
top-left	0	0
top-right	0	1
bottom-left	1	0
bottom-right	1	1

(a) Goals labelled by the well order $\leq$ given by: `top-left` $\leq$ `top-right` $\leq$ `bottom-left` $\leq$ `bottom-right`.

Goals	$\bar{Q}^*_T$	$\bar{Q}^*_L$
bottom-right	0	0
bottom-left	0	1
top-right	1	0
top-left	1	1

(b) Goals labelled by the well order $\leq$ given by: `bottom-right` $\leq$ `bottom-left` $\leq$ `top-right` $\leq$ `top-left`.

Table 3: Base WVFs induced by various well orders on $\mathcal{G}$. Each column represents a base WVF, where **0** or **1** for goal g on WVF $\bar{Q}^*$ means respectively WVF of $\bar{Q}^*(s,g,a) = \bar{Q}^*_{SUP}(s,g,a)$ or $\bar{Q}^*(s,g,a) = \bar{Q}^*_{INF}(s,g,a)$ $\forall(s,a) \in \mathcal{S} \times \mathcal{A}$.

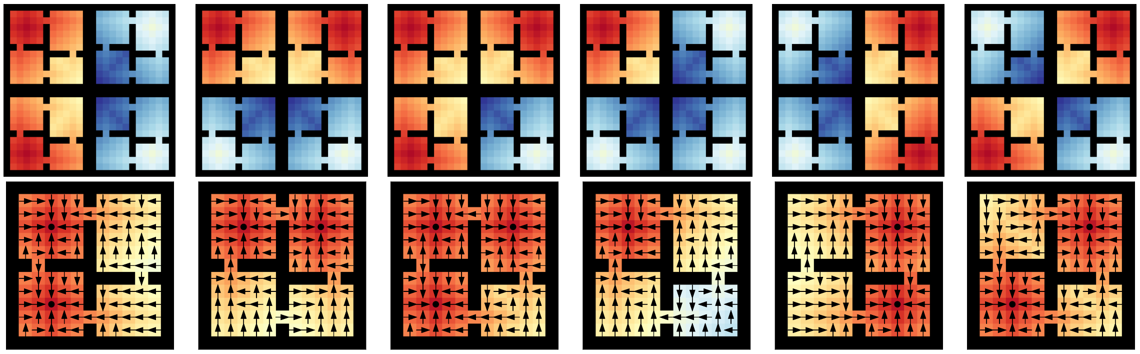
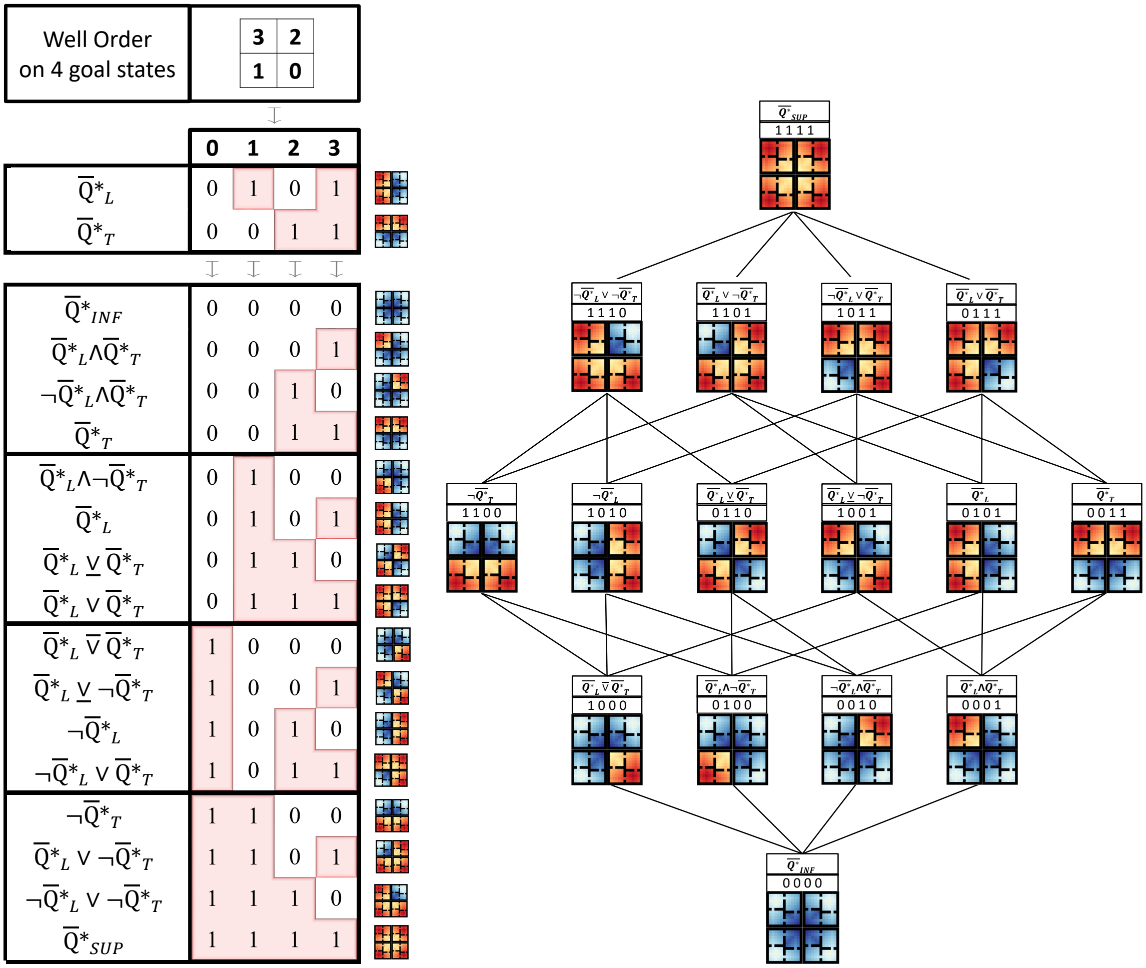

(a) $\bar{Q}^*_L$ (b) $\bar{Q}^*_T$ (c) $\bar{Q}^*_L \vee \bar{Q}^*_T$ (d) $\bar{Q}^*_L \wedge \bar{Q}^*_T$ (e) $\neg\bar{Q}^*_L$ (f) $\bar{Q}^*_L \veebar \bar{Q}^*_T$

Figure 29: An example of Boolean algebraic composition using the generated WVFs of base tasks. Arrows represent the optimal action in a given state.



Well Order on 4 goal states		
	3	2
	1	0

	0	1	2	3
$\bar{Q}^*_L$	0	1	0	1
$\bar{Q}^*_T$	0	0	1	1
$\bar{Q}^*_{INF}$	0	0	0	0
$\bar{Q}^*_L \wedge \bar{Q}^*_T$	0	0	0	1
$\neg\bar{Q}^*_L \wedge \bar{Q}^*_T$	0	0	1	0
$\bar{Q}^*_T$	0	0	1	1
$\bar{Q}^*_L \wedge \neg\bar{Q}^*_T$	0	1	0	0
$\bar{Q}^*_L$	0	1	0	1
$\bar{Q}^*_L \underline{\vee} \bar{Q}^*_T$	0	1	1	0
$\bar{Q}^*_L \vee \bar{Q}^*_T$	0	1	1	1
$\bar{Q}^*_L \overline{\vee} \bar{Q}^*_T$	1	0	0	0
$\bar{Q}^*_L \underline{\vee} \neg\bar{Q}^*_T$	1	0	0	1
$\neg\bar{Q}^*_L$	1	0	1	0
$\neg\bar{Q}^*_L \vee \bar{Q}^*_T$	1	0	1	1
$\neg\bar{Q}^*_T$	1	1	0	0
$\bar{Q}^*_L \vee \neg\bar{Q}^*_T$	1	1	0	1
$\neg\bar{Q}^*_L \vee \neg\bar{Q}^*_T$	1	1	1	0
$\bar{Q}^*_{SUP}$	1	1	1	1

$\bar{Q}^*_{SUP}$
1111
$\neg\bar{Q}^*_L \vee \neg\bar{Q}^*_T$
1110
$\bar{Q}^*_L \vee \neg\bar{Q}^*_T$
1101
$\neg\bar{Q}^*_L \vee \bar{Q}^*_T$
1011
$\bar{Q}^*_L \vee \bar{Q}^*_T$
0111
$\neg\bar{Q}^*_T$
1100
$\neg\bar{Q}^*_L$
1010
$\bar{Q}^*_L \underline{\vee} \bar{Q}^*_T$
0110
$\bar{Q}^*_L \underline{\vee} \neg\bar{Q}^*_T$
1001
$\bar{Q}^*_L$
0101
$\bar{Q}^*_T$
0011
$\bar{Q}^*_L \overline{\vee} \bar{Q}^*_T$
1000
$\bar{Q}^*_L \wedge \neg\bar{Q}^*_T$
0100
$\neg\bar{Q}^*_L \wedge \bar{Q}^*_T$
0010
$\bar{Q}^*_L \wedge \bar{Q}^*_T$
0001
$\bar{Q}^*_{INF}$
0000

(a) Boolean table of base and composed WVFs.

(b) Hasse diagram of the Boolean WVF algebra.

Figure 30: Boolean table and Hasse diagram for the WVFs of tasks in the the four-rooms domain. (a) A well order on $\mathcal{G}$ that labels the goals, the induced base WVFs from labeling the goals, the logical expressions for all 16 compositions of the base WVFs, their Boolean values and value functions per goal. (b) The Boolean WVF algebra showing the WVFs for all 16 tasks in $\mathcal{M}$, together with the Boolean values and logical expressions that generates them from the base WVFs.

can attain a goal not explicitly represented by the base tasks, such as the bottom-right goal for the negation. Figure 30 shows the Boolean table and Hasse diagram for all 16 WVFs generated by them, which spans their respective value spaces.

5.5 Zero-Shot Composition With Function Approximation

In this section, we demonstrate that our compositional approach can also be used to tackle domains where function approximation is required. In this setting, it is likely that learned value functions will be suboptimal owing to generalisation error. However, we show in Theorem 7 that the simplicity of the lattice algebra operators—supremum for disjunction and infimum for conjunction—enables disjunctions and conjunctions of ϵ-optimal WVFs without any decrease in optimality. Furthermore, we show that the negation of an ϵ-optimal WVF leads to only a constant decrease in optimality.

Theorem 7 *Let $\mathcal{M}$ be a set of tasks and $\bar{\mathcal{Q}}^*$ the set of optimal WVFs for tasks in $\mathcal{M}$. Denote ${}^{\epsilon}\bar{Q}^*_M$ as the ϵ-optimal WVF for a task $M \in \mathcal{M}$ such that*

$$|\bar{Q}^*_M(s,g,a) - {}^{\epsilon}\bar{Q}^*_M(s,g,a)| \leq \epsilon \quad \textit{for all} \quad (s,g,a) \in \mathcal{S} \times \mathcal{G} \times \mathcal{A}.$$

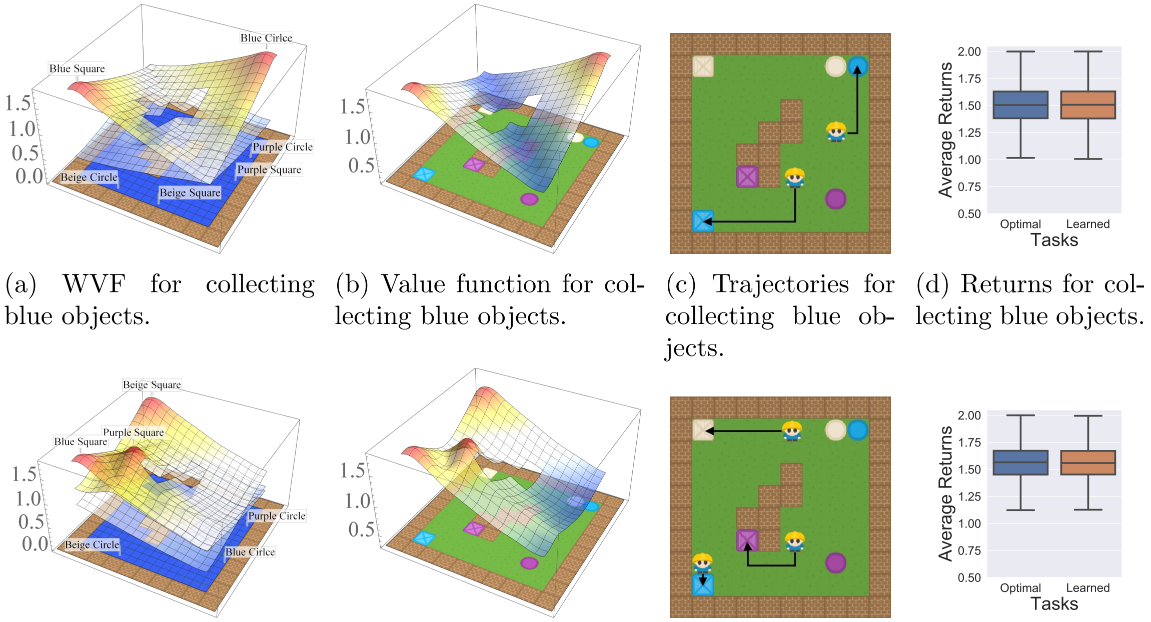
Then for all M_1, M_2 in $\mathcal{M}$ and (s,g,a) in $\mathcal{S} \times \mathcal{G} \times \mathcal{A}$,

$$\textit{(i)} \quad \left| [\bar{Q}^*_{M_1} \vee \bar{Q}^*_{M_2}](s,g,a) - \sup_{M \in \{M_1, M_2\}} {}^{\epsilon}\bar{Q}^*_M(s,g,a) \right| \leq \epsilon$$

$$\textit{(ii)} \quad \left| [\bar{Q}^*_{M_1} \wedge \bar{Q}^*_{M_2}](s,g,a) - \inf_{M \in \{M_1, M_2\}} {}^{\epsilon}\bar{Q}^*_M(s,g,a) \right| \leq \epsilon$$

$$\textit{(iii)} \quad \left| \neg \bar{Q}^*_{M_1}(s,g,a) - \left[\left({}^{\epsilon}\bar{Q}^*_{SUP}(s,g,a) + {}^{\epsilon}\bar{Q}^*_{INF}(s,g,a) \right) - {}^{\epsilon}\bar{Q}^*_{M_1}(s,g,a) \right] \right| \leq 3\epsilon$$

■



(a) WVF for collecting blue objects.

(b) Value function for collecting blue objects.

(c) Trajectories for collecting blue objects.

(d) Returns for collecting blue objects.

(e) WVF for collecting squares.

(f) Value function for collecting squares.

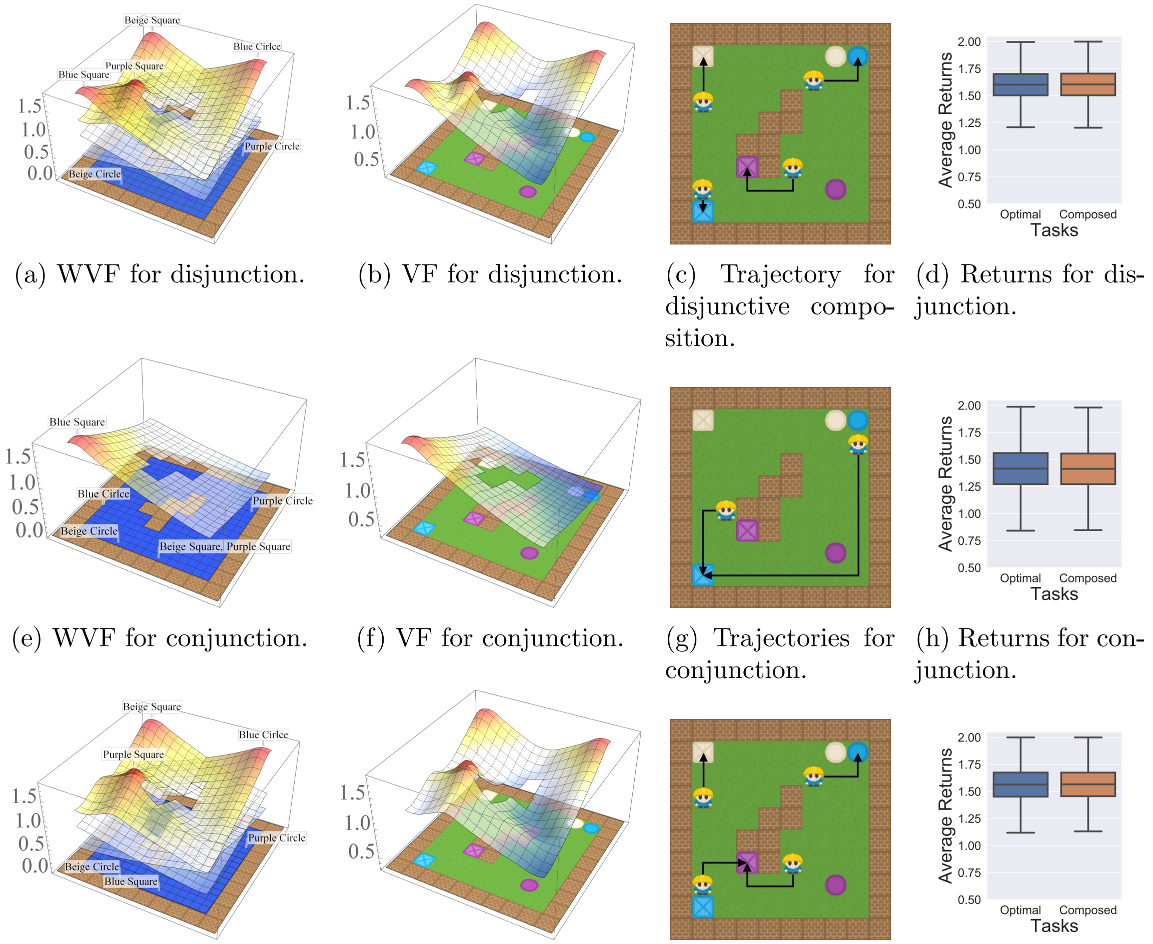
(g) Trajectories for collecting squares.

(h) Returns for collecting squares.

Figure 31: Learned base tasks for the video game environment. We show WVFs learned for the blue, and square tasks. The returns are averaged over 10k episodes.

Experiment 5.5 *We use the same video game environment as in Section 4.3, where the observations are images of the 2D game world and the agent must navigate to collect objects of different shapes and colours. For each task, the agent receives a reward of* 2 *when it collects desired objects and* 0 *everywhere else. The discount factor used is* $\gamma = 0.95$*. We first learn to solve two base tasks: collecting blue objects and collecting squares (Figure 31).*

We demonstrate composition characterised by (i) disjunction, (ii) conjunction, and (iii) exclusive-or. This corresponds to tasks where the target items are: (i) blue or square, (ii) blue squares, and (iii) blue or squares, but not blue squares. Figure 32 illustrates WVFs, as well as value functions obtained by maximising over goals and sample trajectories for the respective tasks. We observe that even with function approximation, composition of WVFs still produces WVFs. That is, the composed WVFs not only lead to optimal value



(a) WVF for disjunction.

(b) VF for disjunction.

(c) Trajectory for disjunctive composition.

(d) Returns for disjunction.

(e) WVF for conjunction.

(f) VF for conjunction.

(g) Trajectories for conjunction.

(h) Returns for conjunction.

(i) WVF for exclusive-or composition.

(j) VF for exclusive-or composition.

(k) Trajectories for exclusive-or composition.

(l) Returns for exclusive-or composition.

Figure 32: By composing the WVFs for collecting blue objects and squares, we can act optimally in new tasks with no further learning. The returns are averaged over 10^4 episodes.

functions and trajectories (see their returns in the last column of Figure 32), but still retain the properties of WVFs, such as mastery.

5.6 Investigating Practical Considerations

The theoretical results presented in this work for zero-shot composition are limited to the family of goal-reaching tasks defined in Definition 7, which restrict the tasks' dynamics (same deterministic transitions across tasks) and non-terminal rewards (same across tasks) in potentially undesirable ways for current RL algorithms and environments. For example, while goal-reaching tasks as defined cover a large number of tasks of interest in RL, and sparse rewards are ideal to avoid reward engineering and misspecification, most current RL algorithms still require dense rewards to learn in complex environments. Additionally, many of the current RL environments and benchmarks are designed such that termination only happens at desirable task goals (instead of any environment goal, for example by having a *done* action).

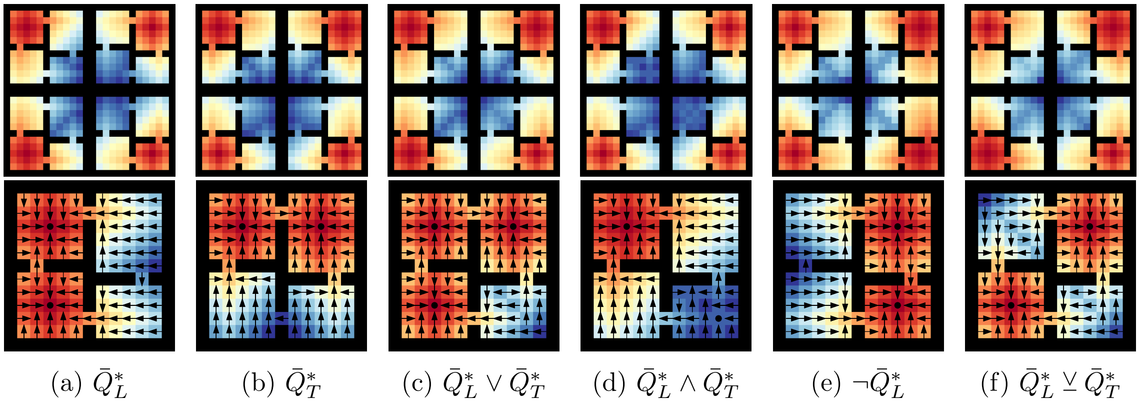
In this section, we investigate whether these theoretical restrictions can be practically ignored. In particular, we investigate the requirement that tasks share the same deterministic dynamics by having noisy transitions with different terminal transitions, and the impact of using dense rewards on non-terminal transitions.

5.6.1 Four Rooms Experiments

We use the same setup as the four-rooms experiment outlined in Experiment 5.4, but modify it in two ways. We first investigate the difference between using sparse and dense rewards. Our sparse reward function is defined as

$$r_{\text{sparse}}(s,a,s') = \begin{cases} 2 & \text{if } s' \text{ is absorbing and } s \text{ is a target task goal,} \\ -0.1 & \text{otherwise,} \end{cases}$$

and we use a dense reward function similar to Peng et al. [2019]:



(a) $\bar{Q}^*_L$ (b) $\bar{Q}^*_T$ (c) $\bar{Q}^*_L \vee \bar{Q}^*_T$ (d) $\bar{Q}^*_L \wedge \bar{Q}^*_T$ (e) $\neg\bar{Q}^*_L$ (f) $\bar{Q}^*_L \veebar \bar{Q}^*_T$

Figure 33: An example of Boolean algebraic composition using the learned WVFs with dense rewards. Top row shows WVFs while bottom one shows resulting value functions and policies obtained by maximising over goals.

$$r_{\text{dense}}(s, a, s') = \frac{0.1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \exp\left(-\frac{|s-g|^2}{4}\right) + r_{\text{sparse}}(s, a, s'). \tag{3}$$

Using this dense reward function, we again learn to solve the two base tasks M_T (centre of top rooms) and M_L (centre of left rooms). We then compose them to solve a variety of tasks, with the resulting value functions illustrated by Figure 33.

We also modify the domain so that tasks have potentially noisy transitions and need not share the same terminating states (that is, if the agent enters a terminating state for a *different* task, the episode does not terminate and the agent can continue as if it were a normal state). The transitions are made noisy by adding a slip probability sp when the agent takes actions in any of the cardinal directions. That is when the agent picks a cardinal action, with probability $1-sp$ the agent goes in the direction it chooses and with probability sp it goes in one of the other 3 chosen uniformly at random. This results in four versions of the experiment for varying noise levels $sp \in \{0, 0.1, 0.3\}$:

(i) `sparse reward and same terminal set for` M_L `and` M_T

(ii) `sparse reward and different terminal set for` M_L `and` M_T

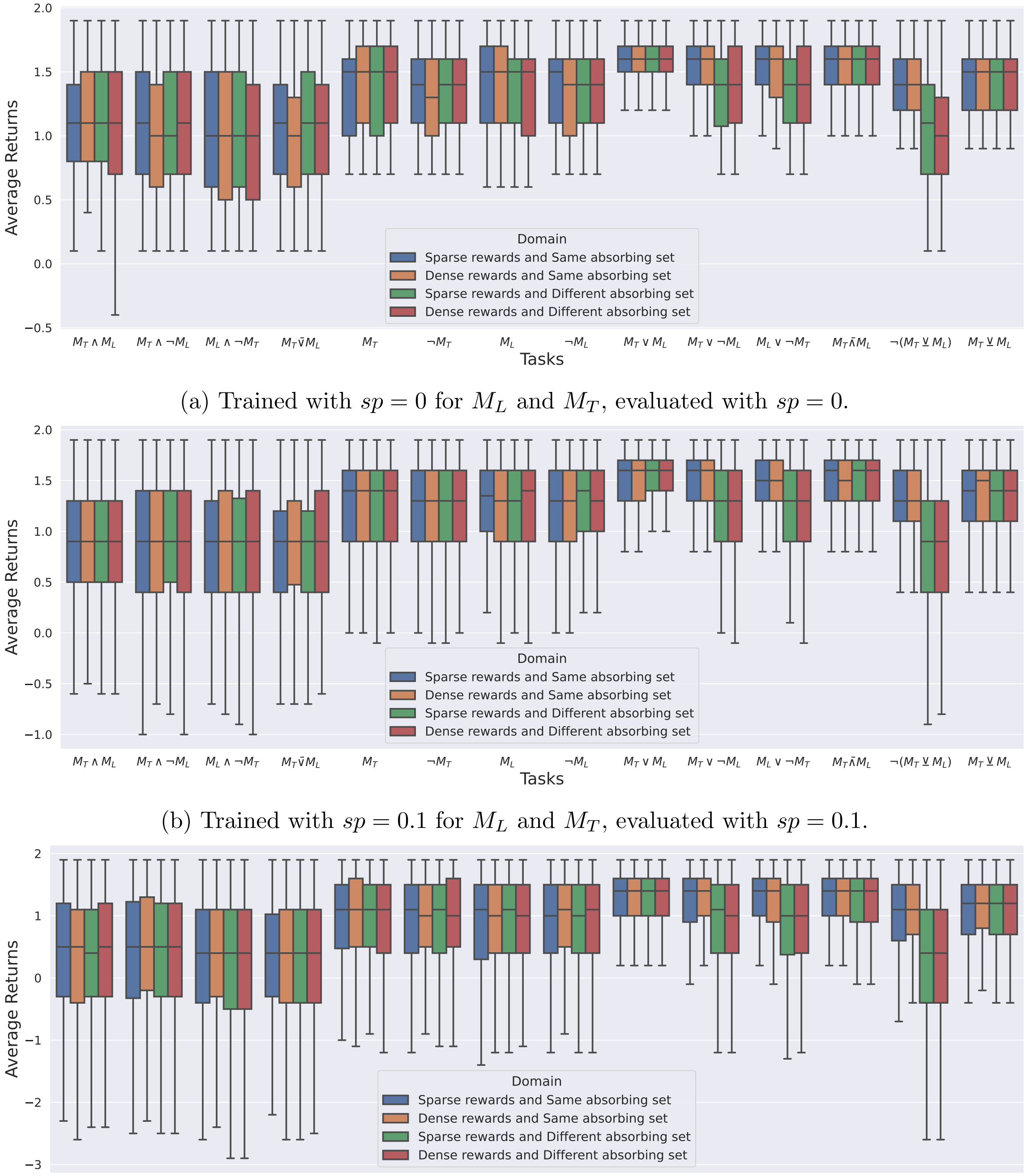
(iii) `dense reward and same terminal set for` M_L `and` M_T

(iv) `dense reward and different terminal set for` M_L `and` M_T

We learn WVFs for each of the above setups, and then compose them to solve each of the 2^4 tasks representable in the Boolean algebra. We measure each composed value function by evaluating its policy in the sparse reward setting, averaging results over 1000 episodes. The results are shown in Figure 34. Our results indicate that WVFs learned in the theoretically optimal manner (`sparse reward, same terminal set`) are indeed optimal. However, for the majority of the tasks with potentially noisy dynamics, relaxing the restrictions on terminal transitions and reward functions results in policies that are either identical or very close to optimal.

5.6.2 Function Approximation Experiments

In this section, we investigate whether we can again loosen some of the restrictive assumptions when tackling high-dimensional, continuous-control environments. We consider a continuous 3D Four Rooms environment where the ant robot of Duan et al. [2016] must navigate to the center of specific rooms. The environment is simulated in MuJoCo [Todorov et al., 2012] with a 29-dimensional continuous state space (representing the position and velocity of the ant's joints) and an 8-dimensional continuous action space. Figure 35 shows a rendered view of the environment.



(a) Trained with $sp = 0$ for M_L and M_T, evaluated with $sp = 0$.

(b) Trained with $sp = 0.1$ for M_L and M_T, evaluated with $sp = 0.1$.

(c) Trained with $sp = 0.3$ for M_L and M_T, evaluated with $sp = 0.3$.

Figure 34: Average returns for each of the compositional tasks, and for each variations of the domain. Results are averaged over 1000 episodes with random start positions.

Since this environment is particularly challenging due to its large state space and the ant's nonlinear, highly unstable dynamics, we use dense rewards for non-terminal transi-

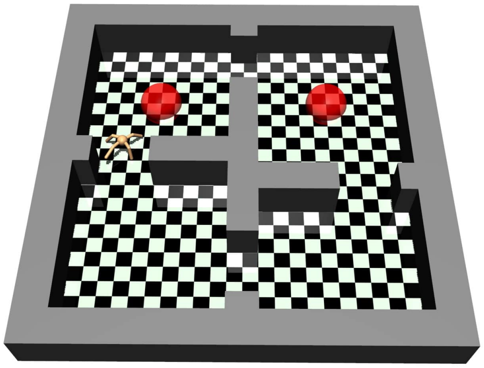
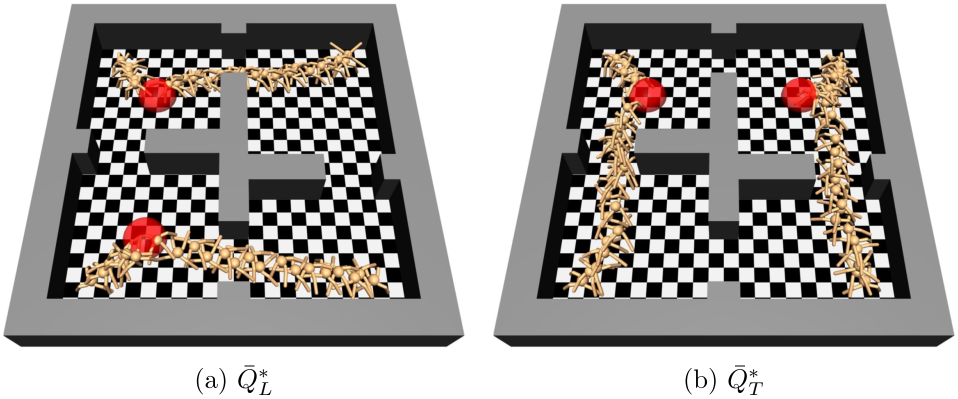


Figure 35: The layout of the MuJoCo Four Rooms domain with a quadruped ant robot. The spheres indicate goals the agent must reach (center of the two top rooms in this case).



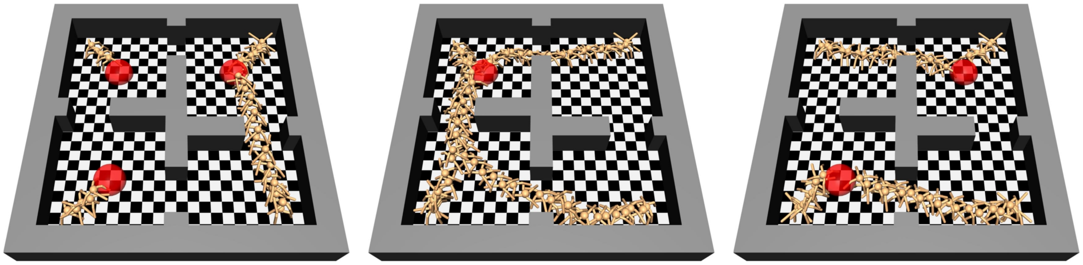
(a) $\bar{Q}^*_L$

(b) $\bar{Q}^*_T$

Figure 36: Learned base tasks for the MuJoCo environment. We show the trajectories obtained from the learned WVFs for each base task from different starting states.

tions (Equation 3) to facilitate learning. Additionally, we restrict terminal states to only the desired ones—specifically, the environment terminates only when the ant is ϵ-close to the center of a desired room. For training, we employ soft actor-critic with automated entropy adjustments (SAC-AEA) [Haarnoja et al., 2018b] as the off-policy RL algorithm to learn world values for each goal (see Appendix B for neural network hyperparameters). We first train on two base tasks: navigating to the top rooms and navigating to the left rooms. The resulting trajectories from the learned WVFs are illustrated in Figure 36. Next, we demonstrate that zero-shot compositions remain valid by showcasing compositions characterized by disjunction, conjunction, and exclusive-or. The resulting trajectories are shown in Figure 37.

In summary, we have shown that our compositional approach offers strong empirical performance, even when the theoretical assumptions are violated. We have also demonstrated that our method can also scale to the function approximation setting, and handle both discrete and continuous state and action spaces. Finally, we expect that, in practice, the errors due to assumption violations will be far outweighed by the errors due to non-linear function approximation.



(a) $\bar{Q}^*_L \vee \bar{Q}^*_T$ (b) $\bar{Q}^*_L \wedge \bar{Q}^*_T$ (c) $\bar{Q}^*_L \veebar \bar{Q}^*_T$

Figure 37: An example of Boolean algebraic composition using the learned WVFs with dense rewards. We show the trajectories obtained from different starting states.

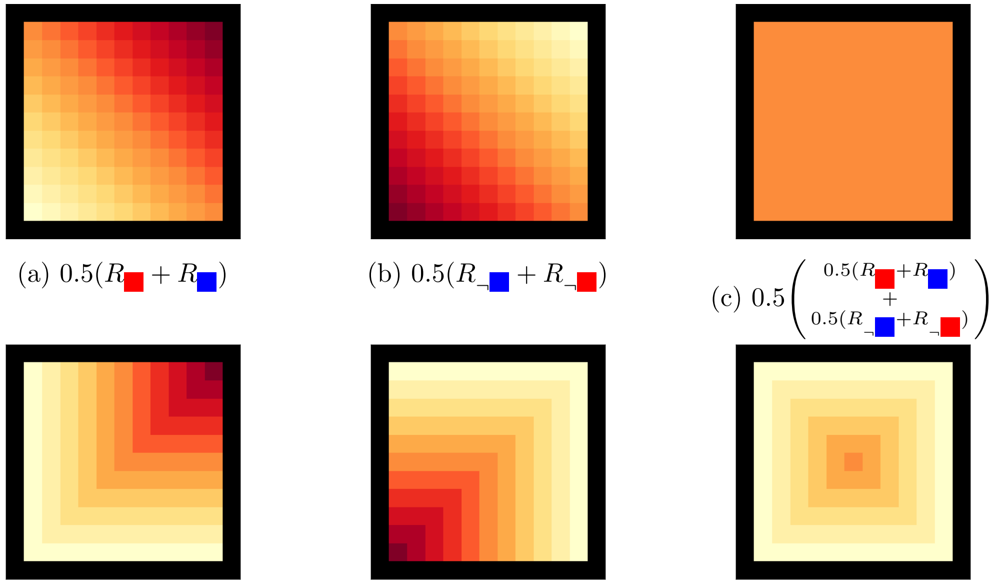
6. Related Works

In this section, we contextualise our work within three broad areas of related work: logical composition, goal-conditioned RL, and temporal-logic composition.

6.1 Logical Composition

The compositionality of tasks and value functions was initially established within the framework of linearly-solvable MDPs (LMDPs) [Todorov, 2007], where linear combinations of value functions were shown to solve disjunctive-like tasks [Todorov, 2009]. Subsequent works introduced hierarchical extensions of LMDPs to enable multi-level task compositions [Saxe et al., 2017; Ringstrom et al., 2020; Infante et al., 2022, 2024], although applications to high-dimensional domains requiring function approximation remained limited.

Building on the ideas from LMDPs, recent works [Haarnoja et al., 2018a; van Niekerk et al., 2019] have demonstrated that entropy-regularised RL [Fox et al., 2016; Haarnoja et al., 2017] could similarly support provable forms of composition, with empirical validations in high-dimensional discrete and continuous domains. In particular, Haarnoja et al. [2018a] proposed using energy-based models in the entropy-regularised setting to approximate task conjunctions by averaging reward functions, showing that averaging the corresponding optimal value functions yields near-optimal behaviour. Extending these insights, van Niekerk et al. [2019] further proved that disjunctive task compositions can be solved zero-shot both in entropy-regularised RL and standard RL under deterministic dynamics. Beyond zero-shot composition, some works also introduce few-shot approaches to compose policies similarly to the conjunctive case [Hunt et al., 2019; Peng et al., 2019; Urain et al., 2023], although generally lacking theoretical guarantees. Adamczyk et al. [2023] later provides general bounds for the sub-optimality of applying different classes of composition operators—beyond linear and logical—on both the reward and value functions, which leads to a reward-shaping approach to learn value functions few-shot. More recently, approaches based on successor features and generalised policy improvement [Barreto et al., 2017] provide a framework for few-shot transfer when the reward functions can be well-approximated as linear combinations of features [Barreto et al., 2018; Borsa et al., 2018; Ma et al., 2020; Alver & Precup, 2022; Carvalho et al., 2023].



(a) $0.5(R_{🟥} + R_{🟦})$

(b) $0.5(R_{\neg 🟦} + R_{\neg 🟥})$

(c) $0.5\begin{pmatrix} 0.5(R_{🟥}+R_{🟦}) \\ + \\ 0.5(R_{\neg 🟦}+R_{\neg 🟥}) \end{pmatrix}$

(d) $\min\{R_{🟥}, R_{🟦}\}$

(e) $\min\{R_{\neg 🟦}, R_{\neg 🟥}\}$

(f) $\min\begin{Bmatrix} \min\{R_{🟥},R_{🟦}\}, \\ \min\{R_{\neg 🟦},R_{\neg 🟥}\} \end{Bmatrix}$

Figure 38: Consider four tasks, 🟥, ¬🟥, 🟦, and ¬🟦 in the bin packing domain, in which the robot must respectively pack all the red objects into the bin, remove all the red objects from the bin, pack all the blue objects into the bin, and remove all the blue objects from the bin. The top row shows their conjunctions approximated using the average of rewards while the bottom one shows their true conjunctions using the *min* of rewards. By averaging rewards, goal states quickly become equally rewarded, losing the meaning of the true conjunction.

All these works either consider disjunctions and conjunctions of tasks separately, or use the linear combination of rewards (such as the average for conjunctions) to unify them. In contrast, we unify these using lattice structures to provably obtain tasks specified as arbitrary disjunctions, conjunctions, and negations of basis tasks. We also note that while previous work has used the average reward function to approximate the conjunction operator [Haarnoja et al., 2018a; Hunt et al., 2019; van Niekerk et al., 2019], tasks specified by averaging the rewards quickly diverge from the task specified by the *min* of rewards (the conjunction). In particular, Figure 38 shows how the average reward task becomes very different to the *min* rewards task after only 3 operations. This highlights the importance of working towards zero-shot composition using the true logical operators, as it enables multiple arbitrary logical compositions while retaining the meaning of the specified tasks.

6.2 Goal-Conditioned RL

While there is a vast literature on goal-conditioned RL [Colas et al., 2022], the idea of learning how to achieve all goals in a multi-goal environment from a single stream of experience was first introduced by Kaelbling [1993] with dynamic goal functions (DGFs). DGFs encode the distance between states and goals learned by giving an agent minimum rewards at non-terminal states and maximum rewards at terminal states. While lacking in theory, Kaelbling

[1993] demonstrated in the tabular case that DGFs learned how to achieve goals significantly faster than standard value functions. Veeriah et al. [2018] later used UVFAs to extend these results to the function approximation case, and showed that such goal-conditioned value functions were also useful as pre-training and auxiliary knowledge for improving sample efficiency. Another similar work is hindsight experience replay [Andrychowicz et al., 2017], where an agent also learns to achieve multiple goals to accelerate the learning of a main task. It is similar to Veeriah et al. [2018] in that it also uses UVFAs for function approximation, but differs in that it only considers tasks with a single desired goal per episode—given upfront to the agent by the environment. The agent then learns to achieve the desired goals faster by also learning how to achieve undesirable ones.

In contrast, WVFs are a principled generalisation of DGFs to the case of arbitrary task reward functions—by using the extended reward function instead of simply giving an agent minimum rewards at internal states and maximum rewards at terminal states—which we show is sufficient for downstream zero-shot composition. WVFs can then be approximated using suitable function approximators like UVFAs, similarly to how Veeriah et al. [2018] uses UVFAs to approximate DGFs.

6.3 Temporal-Logic Composition

Finally, while we focus on tasks specified using only logic operators (like $\vee, \wedge, \neg$), there is a significant number of works that also consider temporal operators (like `NEXT`, `UNTIL`, `EVENTUALLY`) for learning and planning [Littman et al., 2017; Yu et al., 2023]. For example, tasks like 🟥 ∧ ¬🟦 `NEXT` ¬🟥 ∧ 🟦, where the agent needs to first pack only red objects into the bin and then pack only blue objects into the bin. These works often focus on state augmentations and reward shaping for improving sample efficiency [Li et al., 2017; Jothimurugan et al., 2019; Camacho et al., 2019; Voloshin et al., 2023], or task embeddings for improving generalisation [Vaezipoor et al., 2021; León et al., 2022; Yalcinkaya et al., 2024]. However, they assume a given or predefined reward function—commonly binary rewards of 1 for successful transitions and zero or -1 for failure transitions. Some recent works also consider skill composition approaches for improving both sample-efficiency and generalisation [Jothimurugan et al., 2021; Araki et al., 2021; Icarte et al., 2022; Furelos-Blanco et al., 2023; Qiu et al., 2023], however they only consider the temporal composition of sub-tasks and sub-skills (and not their logical composition). This is commonly done by converting a temporal logic task specification into a state machine that represents the temporal order of reach-avoid sub-tasks—specified by Boolean expressions like 🟥 ∧ ¬🟦. These works also assume that the rewards for said sub-tasks are given or predefined, so that temporally composable sub-skills can be learned for each sub-task—often through the options framework [Sutton et al., 1999]. Finally, Tasse et al. [2024] recently proposed Skill Machines to achieve both logical and temporal zero-shot composition, but their reward functions are restricted to binary rewards.

In contrast, we provide a framework for obtaining the logical composition of tasks with arbitrary rewards (and skills for goal-reaching tasks without further learning), with guarantees on the semantics of the logic operators. This can then be used in conjunction with applicable prior works to define sub-task rewards and skills.

7. Conclusion

We have formalised the logical composition of tasks and value functions using lattice algebras. This enables tasks to be treated algebraically in a similar way to sets in set theory, and propositions in propositional logic. We also introduced world value functions—a new type of value function for goal-oriented tasks that encodes more information about solved tasks than standard value functions. We showed how an agent can learn a diversity of solutions for each task using these WVFs. This rich knowledge was sufficient to immediately solve composite tasks. This was achieved by first learning the WVFs, and then composing them similarly to the composed task. This zero-shot ability combined with the notion of basis tasks obtained from the task algebra guaranteed a super-exponential explosion of skills.

However, there is much room for improvement. An exciting avenue for future work is to extend the composition framework beyond the classes of MDPs considered here, such as different environments and partially observable MDPs—since they most accurately model the case of real-world agents. Another natural avenue for future work is extending the logical compositions presented here to temporal compositions, potentially leveraging this framework to derive the sub-task rewards and corresponding sub-skills required by temporal-logic task specification approaches. Extending zero-shot composition to these problems would therefore make the compositions presented in this work far more applicable in practice.

Hence, our proposed approach is a step towards both interpretable RL—since both the tasks and optimal value functions can be specified using logic operators—and the ultimate goal of lifelong learning agents, which are able to solve combinatorially many tasks in a sample-efficient manner.

Acknowledgments

The authors wish to thank Jesse Bristow for his help with the Yumi ABB robot demonstrations in the Bin packing domain. Computations were performed using the High Performance Computing Infrastructure provided by the Mathematical Sciences Support unit at the University of the Witwatersrand. G.N.T. was supported by an IBM PhD Fellowship. B.R. is the director of the MIND institute in Johannesburg and also a CIFAR Azrieli Global Scholar in the Learning in Machines & Brains program. G.N.T. and S.J. are Fellows of the MIND institute.

Appendix A. Pseudo-Codes

Algorithm 2: Goal-oriented learning

```
Input: Off-policy RL algorithm $\mathscr{A}$, number of episodes $N$, lower-bound extended
       reward $\bar{R}_{\text{MIN}}$, discount factor $\gamma$
Initialise $\bar{Q}_\theta : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ according to $\mathscr{A}$
Initialise goal and replay buffers $\mathcal{G} \leftarrow \{\omega\}$ and $\mathcal{D} \leftarrow \varnothing$ respectively
for $e \leftarrow 0$ to $N$ do
    Initialise state $s$
    Select a random goal $g$
    while $s$ is not terminal do
        Select action $a$ using the behaviour policy from $\mathscr{A}$: $a \leftarrow \bar{\pi}_\theta(s, g)$
        Take action $a$, observe reward $r$ and next state $s'$
        if $s$ is terminal then $\mathcal{G} \leftarrow \mathcal{G} \cup \{s\}$ end
        $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s, a, r, s')\}$
        Sample mini-batch of goals $\mathcal{G}' \subseteq \mathcal{G}$ and transitions $\mathcal{D}' \subseteq \mathcal{D}$
        Initialize learning minibatch $\mathcal{B} \leftarrow \varnothing$
        foreach $g', (s, a, r, s') \in \mathcal{G}' \times \mathcal{D}'$ do
            $\bar{R} \leftarrow \bar{R}_{\text{MIN}}$ if $g' \neq s \in \mathcal{G}$ else $r$                /* ERF, Equation 1 */
            $\mathcal{B} \leftarrow \mathcal{B} \cup \{(s, g', a, \bar{R}, s')\}$
        Update $\bar{Q}_\theta$ with $\mathscr{A}$ using the discount factor $\gamma$ and minibatch $\mathcal{B}$
        $s \leftarrow s'$
return $\bar{Q}_\theta$
```

Appendix B. Additional Training Details

B.1 Function Approximation for the 2D Video Game Environment

To train the world value functions in the 2D video game environment, we use a neural network with the following architecture:

1. Three convolutional layers: (a) Layer 1 has 6 input channels, 32 output channels, a kernel size of 8 and a stride of 4. (b) Layer 2 has 32 input channels, 64 output channels, a kernel size of 4 and a stride of 2. (c) Layer 3 has 64 input channels, 64 output channels, a kernel size of 3 and a stride of 1.

2. Two fully-connected linear layers: (a) Layer 4 has input size 3136 and output size 512 and uses a ReLU activation function. (b) Layer 5 has input size 512 and output size 5 with no activation function.

We use the ADAM optimiser with mini-batch size 32 and a learning rate of 10^{-4}. We train every 4 timesteps and update the target Q-network every 1000 steps. Finally, we use ϵ-greedy exploration, annealing ϵ from 1 to 0.01 over 100000 timesteps.

B.2 Function Approximation for the MuJoCo Four Rooms Environment

To train a world value function in the MuJoCo Four Rooms Environment, we use $2 \times |\mathcal{G}|$ neural networks to represent the policy and value function for each goal. For each goal, we use the following architectures for the policy and action-value networks:

1. Three fully-connected linear layers for the policy networks: (a) Layer 1 has input size 29 and output size 256 and uses a ReLU activation function. (b) Layer 2 has input size 256 and output size 256 and uses a ReLU activation function. (c) Layer 3 has input size 256 and output size 16, representing the mean and standard deviation of the Gaussian distribution over actions.

2. Three fully-connected linear layers for the action-value networks: (a) Layer 1 has input size 37 and output size 256 and uses a ReLU activation function. (b) Layer 2 has input size 256 and output size 256 and uses a ReLU activation function. (c) Layer 3 has input size 256 and output size 1 with no activation function.

We use the ADAM optimiser with a learning rate of 10^{-4}.

Appendix C. Proofs of Theoretical Results

C.1 Composing Tasks

Proposition 1 *Let $M_1, M_2 \in \mathcal{M}_{\mathbb{R}}$ be tasks with reward functions R_{M_1} and R_{M_2} respectively. Then $(\mathcal{M}_{\mathbb{R}}, \leq)$ is a partially ordered set with the relation $\leq$ given by*

$$M_1 \leq M_2 \textit{ if } R_{M_1}(s,a,s') \leq R_{M_2}(s,a,s') \textit{ for all } (s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}.$$

Proof Follows from the usual $\leq$ relation on $\mathbb{R}$. ■

Proposition 2 $(\mathcal{M}_{\mathbb{R}}, \vee, \wedge)$ *is a distributive lattice.*

Proof Follows from the properties of inf and sup and their distributivity. ■

Proposition 3 $(\mathcal{M}_{[\mathcal{M}_{INF}, \mathcal{M}_{SUP}]}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ *is a De Morgan algebra.*

Proof Let $M_1, M_2 \in \mathcal{M}_{[\mathcal{M}_{INF}, \mathcal{M}_{SUP}]}$. We show that $\neg, \vee, \wedge$ satisfy the De Morgan algebra axioms.

(i)–(v): These follow from the properties of inf and sup.

(vi): This follows from the bounds $\mathcal{M}_{SUP}, \mathcal{M}_{INF} \in \mathcal{M}_{[\mathcal{M}_{INF}, \mathcal{M}_{SUP}]}$ which are guaranteed to exist by definition.

(vii): The first condition easily follows from the definition of $\neg$. For the second condition, let $R_{\neg(M_1 \vee M_2)}$ be the reward function for $\neg(M_1 \vee M_2)$. Then for all (s,a,s') in $\mathcal{S} \times \mathcal{A} \times \mathcal{S}$,

$$\begin{aligned}
R_{\neg(M_1 \vee M_2)}(s,a,s') &= (R_{\mathcal{M}_{SUP}}(s,a,s') + R_{\mathcal{M}_{INF}}(s,a,s')) - \sup_{M \in \{M_1,M_2\}} R_M(s,a,s') \\
&= (R_{\mathcal{M}_{SUP}}(s,a,s') + R_{\mathcal{M}_{INF}}(s,a,s')) + \inf_{M \in \{M_1,M_2\}} -R_M(s,a,s') \\
&= \inf_{M \in \{M_1,M_2\}} (R_{\mathcal{M}_{SUP}}(s,a,s') + R_{\mathcal{M}_{INF}}(s,a,s')) - R_M(s,a,s') \\
&= R_{\neg M_1 \wedge \neg M_2}(s,a,s').
\end{aligned}$$

Thus $\neg(M_1 \vee M_2) = \neg M_1 \wedge \neg M_2$.

■

Proposition 4 $(\mathcal{M}_{\{\mathcal{M}_{INF},\mathcal{M}_{SUP}\}}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ *is a Boolean algebra.*

Proof Let $M_1, M_2 \in \mathcal{M}_{\{\mathcal{M}_{INF},\mathcal{M}_{SUP}\}}$. We show that $\neg, \vee, \wedge$ satisfy the Boolean algebra axioms.

(i)–(vi): These follow from the De Morgan task algebra since $\mathcal{M}_{\{\mathcal{M}_{INF},\mathcal{M}_{SUP}\}}$ satisfies its assumptions.

(vii): Let $R_{M_1 \wedge \neg M_1}$ be the reward function for $M_1 \wedge \neg M_1$. Then for all (s,a,s') in $\mathcal{S} \times \mathcal{A} \times \mathcal{S}$,

$$\begin{aligned}
R_{M_1 \wedge \neg M_1}(s,a,s') &= \inf\{R_{M_1}(s,a,s'), \\
&\qquad (R_{\mathcal{M}_{SUP}}(s,a,s') + R_{\mathcal{M}_{INF}}(s,a,s')) - R_{M_1}(s,a,s')\} \\
&= \begin{cases} R_{\mathcal{M}_{INF}}(s,a,s'), & \text{if } R_{M_1}(s,a,s') = R_{\mathcal{M}_{SUP}}(s,a,s') \\ R_{\mathcal{M}_{INF}}(s,a,s'), & \text{if } R_{M_1}(s,a,s') = R_{\mathcal{M}_{INF}}(s,a,s') \end{cases} \\
&= R_{\mathcal{M}_{INF}}(s,a,s').
\end{aligned}$$

Thus $M_1 \wedge \neg M_1 = \mathcal{M}_{INF}$, and similarly $M_1 \vee \neg M_1 = \mathcal{M}_{SUP}$.

■

Proposition 5 *The Boolean task algebra on* $\mathcal{M}_{\{\mathcal{M}_{INF},\mathcal{M}_{SUP}\}}$ *is isomorphic to the power set Boolean algebra on* $2^{\mathcal{S} \times \mathcal{A} \times \mathcal{S}}$.

Proof This follows from the bijection F and the fact that it is clearly homomorphic. ■

C.2 World Value Functions

Theorem 1 *Let R_M, $\bar{R}_M$, Q^*_M and $\bar{Q}^*_M$ be the standard reward function, extended reward function, optimal value function and optimal world value function respectively for a goal-reaching task M. Then for all (s,a) in $\mathcal{S} \times \mathcal{A}$, we have*

$$R_M(s,a,s') = \max_{g \in \mathcal{G}} \bar{R}_M(s,g,a,s') \text{ and } Q^*_M(s,a) = \max_{g \in \mathcal{G}} \bar{Q}^*_M(s,g,a).$$

Proof

We will omit the task subscript M for better readability. First note that

$$\max_{g \in \mathcal{G}} \bar{R}(s,g,a,s') = \begin{cases} \max\{\bar{R}_{\text{MIN}}, R(s,a,s')\}, & \text{if } s' \text{ is absorbing} \\ R(s,a,s'), & \text{otherwise.} \end{cases} = R(s,a,s'). \tag{4}$$

Now define

$$Q^*_{max}(s,a) \coloneqq \max_{g \in \mathcal{G}} \bar{Q}^*(s,g,a).$$

Let $\mathcal{T}$ be the Bellman optimality operator. Then, if $s' \sim p(\cdot|s,a)$ is not absorbing, it follows that

$$\begin{aligned}
&[\mathcal{T}Q^*_{max}](s,a) = \sum_{s' \in \mathcal{S}} p(s'|s,a) \left[R(s,a,s') + \gamma \max_{a' \in \mathcal{A}} Q^*_{max}(s',a') \right] \\
&= \sum_{s' \in \mathcal{S}} p(s'|s,a) \left[R(s,a,s') + \gamma \max_{a' \in \mathcal{A}} \max_{g \in \mathcal{G}} \bar{Q}^*(s',g,a') \right] \\
&= \max_{g \in \mathcal{G}} \sum_{s' \in \mathcal{S}} p(s'|s,a) \left[R(s,a,s') + \gamma \max_{a' \in \mathcal{A}} \bar{Q}^*(s',g,a') \right] \\
&\quad \text{(Since } \gamma \geq 0 \text{ and the dynamics are deterministic)} \\
&= \max_{g \in \mathcal{G}} \sum_{s' \in \mathcal{S}} p(s'|s,a) \left[R(s,g,a,s') + \gamma \max_{a' \in \mathcal{A}} \bar{Q}^*(s',g,a') \right] \quad \text{(Using Equation 4)} \\
&= \max_{g \in \mathcal{G}} \bar{Q}^*(s,g,a) \\
&= Q^*_{max}(s,a).
\end{aligned}$$

Hence Q^*_{max} is a fixed point of the Bellman optimality operator.
If $s' \sim p(\cdot|s,a)$ is absorbing, then

$$Q^*_{max}(s,a) = \max_{g \in \mathcal{G}} \bar{Q}^*(s,g,a) = \max_{g \in \mathcal{G}} \bar{R}(s,g,a,s') = R(s,a,s') = Q^*(s,a).$$

Since $Q^*_{max} = Q^*$ holds in $\mathcal{G}$ and Q^*_{max} is a fixed point of $\mathcal{T}$, then $Q^*_{max} = Q^*$ holds everywhere.

■

Theorem 2 *For all goal-reaching tasks M, $\bar{Q}^*_M$ has mastery.*

Proof

Let each g in $\mathcal{G}$ define an MDP M_g with reward function

$$R_g(s,a,s') := \bar{R}_M(s,g,a,s')$$

for all (s,a) in $\mathcal{S} \times \mathcal{A}$. By Definition 7, there exists an optimal policy that reaches a terminal transition. Let

$$\pi^*_g(s) \in \arg\max_{a \in \mathcal{A}} Q^*_g(s,a) \text{ for all } s \in \mathcal{S}.$$

be such a policy. If g *is* reachable from $s \in \mathcal{S} \setminus \{g\}$, then we show that following π^*_g must reach g. Assume π^*_g reaches a different goal state g', with $g' \neq g$. Let π_g be a policy that produces the shortest trajectory to g. Also let $G^{\pi^*_g}$ and G^{π_g} be the returns for the respective policies. Then,

$$\begin{aligned}
&G^{\pi^*_g} \geq G^{\pi_g} \\
&\implies G^{\pi^*_g}_{T-1} + R_g(g', \pi^*_g(g'), s') \geq G^{\pi_g}, \\
&\text{where } G^{\pi^*_g}_{T-1} = \sum_{t=0}^{T-1} \gamma^t R_{M_g}(s_t, \pi^*_g(s_t), s_{t+1}) \\
&\text{and } T \text{ is the time at which } g' \text{ is reached.} \\
&\implies G^{\pi^*_g}_{T-1} + \bar{R}_{\text{MIN}} \geq G^{\pi_g}, \text{ since } g \neq g' \in \mathcal{G} \\
&\implies \bar{R}_{\text{MIN}} \geq G^{\pi_g} - G^{\pi^*_g}_{T-1} \\
&\implies (R_{\text{MIN}} - R_{\text{MAX}})D \geq G^{\pi_g} - G^{\pi^*_g}_{T-1}, \\
&\quad \text{by definition of } \bar{R}_{\text{MIN}} \\
&\implies G^{\pi^*_g}_{T-1} - R_{\text{MAX}}D \geq G^{\pi_g} - R_{\text{MIN}}D, \\
&\quad \text{since } G^{\pi_g} \geq R_{\text{MIN}}D \\
&\implies G^{\pi^*_g}_{T-1} - R_{\text{MAX}}D \geq 0 \\
&\implies G^{\pi^*_g}_{T-1} \geq R_{\text{MAX}}D.
\end{aligned}$$

But this is a contradiction, since the result obtained by following an optimal trajectory up to a terminal state without the reward for entering the terminal state must be strictly less than receiving R_{MAX} for every step of the longest possible optimal trajectory. Hence we must have $g' = g$.

■

C.3 Composing World Value Functions

Proposition 6 *Let $\bar{\mathcal{Q}}^*$ be the set of optimal $\bar{Q}$-value functions for tasks in a set of tasks $\mathcal{M}$. Then $(\bar{\mathcal{Q}}^*, \leq)$ is a partially ordered set with the relation $\leq$ given by*

$$\bar{Q}^*_1 \leq \bar{Q}^*_2 \textit{ if } \bar{Q}^*_1(s,g,a) \leq \bar{Q}^*_2(s,g,a) \textit{ for all } (s,g,a) \in \mathcal{S} \times \mathcal{G} \times \mathcal{A}.$$

Proof Follows from the usual $\leq$ relation on $\mathbb{R}$. ∎

Lemma 1 *Let $(\mathcal{M}, \vee, \wedge)$ be a lattice of goal-reaching tasks. Also let $M_1, M_2 \in \mathcal{M}$ and $g \in \mathcal{G}$ with*

$$\bar{R}_{M_1}(g, g, \bar{\pi}^*_{M_1}(g,g), s') \leq \bar{R}_{M_2}(g, g, \bar{\pi}^*_{M_2}(g,g), s').$$

Then for all $s \in \mathcal{S}$, we have

$$\bar{V}^*_{M_1}(s,g) \leq \bar{V}^*_{M_2}(s,g).$$

Proof

$\bar{R}_{M_1}(g, g, \bar{\pi}^*_{M_1}(g,g), s') \leq \bar{R}_{M_2}(g, g, \bar{\pi}^*_{M_2}(g,g), s')$

$\implies \bar{R}_{M_1}(s, g, \bar{\pi}^*_{M_1}(s,g), s') \leq \bar{R}_{M_2}(s, g, \bar{\pi}^*_{M_2}(s,g), s')$ for all $s \in \mathcal{S}$,

since the rewards are the same at non-terminal states and $\bar{R}_{\text{MIN}}$ at terminal states $s \neq g$.

$$\implies \mathbb{E}_{\bar{\pi}^*_{M_1}}\left[\sum_{t=0}^{\infty} \gamma^t \bar{R}(s_t, g, a_t, s_{t+1})\right] \leq \mathbb{E}_{\bar{\pi}^*_{M_2}}\left[\sum_{t=0}^{\infty} \gamma^t \bar{R}(s_t, g, a_t, s_{t+1})\right]$$

$\implies \bar{V}^*_{M_1}(s,g) \leq \bar{V}^*_{M_2}(s,g).$

∎

Proposition 7 *Let $(\mathcal{M}, \vee, \wedge)$ be a lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then for all $M_1, M_2 \in \mathcal{M}$,*

(i): $\sup\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\} = \bar{Q}^*_{M_1 \vee M_2} \in \bar{\mathcal{Q}}^*$, **(ii):** $\inf\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\} = \bar{Q}^*_{M_1 \wedge M_2} \in \bar{\mathcal{Q}}^*$.

Proof Let $M_1, M_2 \in \mathcal{M}$. Then for all (s,g,a) in $\mathcal{S} \times \mathcal{G} \times \mathcal{A}$,

(i):

$$\begin{aligned}
\bar{Q}^*_{sup}(s,g,a) &\coloneqq \sup_{M \in \{M_1, M_2\}} \bar{Q}^*_M(s,g,a) \\
&= \sup_{M \in \{M_1, M_2\}} \sum_{s' \in \mathcal{S}} p(s,a,s') \left[\bar{R}_M(s,g,a,s') + \gamma \bar{V}^*_M(s',g)\right] \\
&= \sum_{s' \in \mathcal{S}} p(s,a,s') \left[\sup_{M \in \{M_1, M_2\}} \bar{R}_M(s,g,a,s') + \gamma \sup_{M \in \{M_1, M_2\}} \bar{V}^*_M(s',g)\right] \quad \text{(Using Lemma 1)} \\
&= \sum_{s' \in \mathcal{S}} p(s,a,s') \left[\bar{R}_{M_1 \vee M_2}(s,g,a,s') + \gamma \bar{V}^*_{M_1 \vee M_2}(s',g)\right],
\end{aligned}$$

Using Lemma 1 since $\sup\limits_{M \in \{M_1, M_2\}} \bar{R}_M(g, g, \bar{\pi}^*_M(g,g), s')$ defines $\bar{V}^*_{M_1 \vee M_2}(s',g)$.

$$= \bar{Q}^*_{M_1 \vee M_2}(s,g,a).$$

$\implies \bar{Q}^*_{sup} = \bar{Q}^*_{M_1 \vee M_2} \in \bar{\mathcal{Q}}^*$.

Since $\bar{Q}^*_{sup}$ is in $\bar{\mathcal{Q}}^*$, it follows from the pointwise $\leq$ on $\mathbb{R}$ that it is the lowest upper bound of $\bar{Q}^*_1$ and $\bar{Q}^*_2$.

(ii): Follows similarly to (i).

■

Proposition 8 *Let $(\mathcal{M}, \vee, \wedge)$ be a lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then $(\bar{\mathcal{Q}}^*, \vee, \wedge)$ is a distributive lattice.*

Proof Follows from the distributivity of inf and sup. ■

Theorem 3 *Let $(\mathcal{M}, \vee, \wedge)$ be a lattice of goal-reaching tasks, and let $(\bar{\mathcal{Q}}^*, \vee, \wedge)$ be the corresponding lattice of WVFs. Let $H\ : \mathcal{M} \to \bar{\mathcal{Q}}^*$ be any map from $\mathcal{M}$ to $\bar{\mathcal{Q}}^*$ such that $H(M) = \bar{Q}^*_M$ for all M in $\mathcal{M}$. Then H is a homomorphism.*

Proof Follows from the proof of Proposition 7, which gives

$$\bar{Q}^*_{M_1} \vee \bar{Q}^*_{M_2} = \sup\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\} = \bar{Q}^*_{M_1 \vee M_2}$$

and

$$\bar{Q}^*_{M_1} \wedge \bar{Q}^*_{M_2} = \inf\{\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2}\} = \bar{Q}^*_{M_1 \wedge M_2}$$

$\forall M_1, M_2 \in \mathcal{M}$. ■

Lemma 2 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a De Morgan lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then for all s in $\mathcal{S} \setminus \mathcal{G}$ and g in $\mathcal{G}$, there exists an optimal policy $\bar{\pi}^*$ such that*

$$\bar{\pi}^*(s,g) \in \arg\max_{a \in \mathcal{A}} \bar{Q}^*_{M_1}(s,g,a) \text{ and } \bar{\pi}^*(s,g) \in \arg\max_{a \in \mathcal{A}} \bar{Q}^*_{M_2}(s,g,a)\ \forall M_1, M_2 \in \mathcal{M}.$$

Proof Let $g \in \mathcal{G}, s \in \mathcal{S}^-$.

If g *is* reachable from s, then we are done since $\bar{Q}^*_{M_1}$ and $\bar{Q}^*_{M_2}$ have mastery.

If g is unreachable from s, then for all (a, s') in $\mathcal{A} \times \mathcal{S}$ we have

$$\begin{aligned} s \neq g \implies \bar{R}_{M_1}(s,g,a,s') &= \begin{cases} \bar{R}_{\text{MIN}}, & \text{if } s' \text{ is absorbing} \\ 0, & \text{otherwise} \end{cases} = \bar{R}_{M_2}(s,g,a,s') \\ \implies \bar{Q}^*_{M_1}(s,g,a) &= \bar{Q}^*_{M_2}(s,g,a). \end{aligned}$$

■

Proposition 9 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a De Morgan lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then $\neg\bar{Q}^*_M = \bar{Q}^*_{\neg M} \in \bar{\mathcal{Q}}^*$ for all $M \in \mathcal{M}$.*

Proof Let $\mathcal{T}$ be the Bellman optimality operator. Then, for all (s, g, a) in $\mathcal{S} \times \mathcal{G} \times \mathcal{A}$,

$$
\begin{aligned}
&\left[\mathcal{T}\neg\bar{Q}^*_M\right](s,g,a) = \sum_{s' \in \mathcal{S}} p(s,a,s') \left[\bar{R}_{\neg M}(s,g,a,s') + \gamma \max_{a' \in \mathcal{A}} \neg\bar{Q}^*_M(s',g,a')\right] \\
&= \sum_{s' \in \mathcal{S}} p(s,a,s') \Bigg[\Big(\bar{R}_{\mathcal{M}_{SUP}}(s,g,a,s') + \bar{R}_{\mathcal{M}_{INF}}(s,g,a,s') - \bar{R}_M(s,g,a,s')\Big) + \\
&\qquad\qquad \gamma \max_{a' \in \mathcal{A}} \Big(\bar{Q}^*_{SUP}(s',g,a') + \bar{Q}^*_{INF}(s',g,a') - \bar{Q}^*_M(s',g,a')\Big)\Bigg] \\
&= \sum_{s' \in \mathcal{S}} p(s,a,s') \Bigg[\Big(\bar{R}_{\mathcal{M}_{SUP}}(s,g,a,s') + \bar{R}_{\mathcal{M}_{INF}}(s,g,a,s') - \bar{R}_M(s,g,a,s')\Big) + \\
&\qquad\qquad \gamma\Big(\max_{a' \in \mathcal{A}} \bar{Q}^*_{SUP}(s',g,a') + \max_{a' \in \mathcal{A}} \bar{Q}^*_{INF}(s',g,a') - \max_{a' \in \mathcal{A}} \bar{Q}^*_M(s',g,a')\Big)\Bigg] \\
&\quad \text{(Using Lemma 2)} \\
&= \sum_{s' \in \mathcal{S}} p(s,a,s') \left[\bar{R}_{\mathcal{M}_{SUP}}(s,g,a,s') + \gamma \max_{a' \in \mathcal{A}} \bar{Q}^*_{SUP}(s',g,a')\right] + \\
&\quad \sum_{s' \in \mathcal{S}} p(s,a,s') \left[\bar{R}_{\mathcal{M}_{INF}}(s,g,a,s') + \gamma \max_{a' \in \mathcal{A}} \bar{Q}^*_{INF}(s',g,a')\right] - \\
&\quad \sum_{s' \in \mathcal{S}} p(s,a,s') \left[\bar{R}_M(s,g,a,s') + \gamma \max_{a' \in \mathcal{A}} \bar{Q}^*_M(s',g,a')\right] \\
&= \big(\bar{Q}^*_{SUP}(s,g,a) + \bar{Q}^*_{INF}(s,g,a)\big) - \bar{Q}^*_M(s,g,a) \\
&= \neg\bar{Q}^*_M(s,g,a).
\end{aligned}
$$

Hence $\neg\bar{Q}^*_M$ is a fixed point of the Bellman optimality operator.
If $s \in \mathcal{G}$, then

$$
\begin{aligned}
\neg\bar{Q}^*_M(s,g,a) &= \big(\bar{Q}^*_{SUP}(s,g,a) + \bar{Q}^*_{INF}(s,g,a)\big) - \bar{Q}^*_M(s,g,a) \\
&= \big(\bar{R}_{\mathcal{M}_{SUP}}(s,g,a,s') + \bar{R}_{\mathcal{M}_{INF}}(s,g,a,s')\big) - \bar{R}_M(s,g,a,s') = \bar{Q}^*_{\neg M}(s,g,a).
\end{aligned}
$$

Since $\neg\bar{Q}^*_M = \bar{Q}^*_{\neg M}$ holds in $\mathcal{G}$ and $\neg\bar{Q}^*_M$ is a fixed point of $\mathcal{T}$, then $\neg\bar{Q}^*_M = \bar{Q}^*_{\neg M}$ holds everywhere.

■

Proposition 10 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a De Morgan lattice of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then $(\bar{\mathcal{Q}}^*, \vee, \wedge, \neg, \bar{Q}^*_{SUP}, \bar{Q}^*_{INF})$ is a De Morgan algebra.*

Proof Let $\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2} \in \bar{\mathcal{Q}}^*$ be the optimal $\bar{Q}$-value functions for tasks $M_1, M_2 \in \mathcal{M}$ with reward functions r_{M_1} and r_{M_2}. We show that $\neg, \vee, \wedge$ satisfy the De Morgan algebra axioms (i) – (vii).[7]

(i)–(v): These follow from the properties of inf and sup.

(vi): This follows from the bounds $\bar{Q}^*_{SUP}, \bar{Q}^*_{INF} \in \bar{\mathcal{Q}}^*$.

(vii): The first condition easily follows from the definition of $\neg$. For the second condition, we have that for all (s, g, a) in $\mathcal{S} \times \mathcal{G} \times \mathcal{A}$,

$$\begin{aligned}
\neg(\bar{Q}^*_{M_1} \vee \bar{Q}^*_{M_2})(s,g,a) &= (\bar{Q}^*_{SUP}(s,g,a) + \bar{Q}^*_{INF}(s,g,a)) - \sup_{M \in \{M_1, M_2\}} \bar{Q}^*_M(s,g,a) \\
&= (\bar{Q}^*_{SUP}(s,g,a) + \bar{Q}^*_{INF}(s,g,a)) + \inf_{M \in \{M_1, M_2\}} -\bar{Q}^*_M(s,g,a) \\
&= \inf_{M \in \{M_1, M_2\}} (\bar{Q}^*_{SUP}(s,g,a) + \bar{Q}^*_{INF}(s,g,a)) - \bar{Q}^*_M(s,g,a) \\
&= (\neg\bar{Q}^*_{M_1} \wedge \neg\bar{Q}^*_{M_2})(s,g,a).
\end{aligned}$$

■

Theorem 4 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a De Morgan lattice of goal-reaching tasks, and let $(\bar{\mathcal{Q}}^*, \vee, \wedge, \neg, \bar{Q}^*_{SUP}, \bar{Q}^*_{INF})$ be the corresponding De Morgan lattice over WVFs. Let $H : \mathcal{M} \to \bar{\mathcal{Q}}^*$ be any map from $\mathcal{M}$ to $\bar{\mathcal{Q}}^*$ such that $H(M) = \bar{Q}^*_M$ for all M in $\mathcal{M}$. Then H is a homomorphism.*

Proof This follows from the proof of Proposition 9 and the homomorphism between the task and WVF lattices. ■

Proposition 11 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a Boolean algebra of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$.*
Then $(\bar{\mathcal{Q}}^*, \vee, \wedge, \neg, \bar{Q}^*_{SUP}, \bar{Q}^*_{INF})$ is a Boolean WVF algebra.

Proof

Let $\bar{Q}^*_{M_1}, \bar{Q}^*_{M_2} \in \bar{\mathcal{Q}}^*$ be the optimal $\bar{Q}$-value functions for tasks $M_1, M_2 \in \mathcal{M}$ with reward functions r_{M_1} and r_{M_2}. We show that $\neg, \vee, \wedge$ satisfy the Boolean algebra axioms (i) – (vii) stated in Definition 4.

(i)–(vi): These follow from the De Morgan WVF algebra since $\bar{\mathcal{Q}}^*$ satisfies its assumptions.

(vii):

$$\begin{aligned}
\bar{Q}^*_{M_1} \wedge \neg\bar{Q}^*_{M_1} &= \bar{Q}^*_{M_1} \wedge \bar{Q}^*_{\neg M_1} \quad \text{(Proposition 9)} \\
&= \bar{Q}^*_{M_1 \wedge \neg M_1} \quad \text{(Theorem 3)} \\
&= \bar{Q}^*_{INF} \quad \text{(Proposition 4).}
\end{aligned}$$

[7] The De Morgan algebra axioms are stated in Definition 3.

Similarly, $\bar{Q}^*_{M_1} \vee \neg\bar{Q}^*_{M_1} = \bar{Q}^*_{SUP}$. ■

Theorem 5 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a Boolean algebra of goal-reaching tasks, and let $(\bar{\mathcal{Q}}^*, \vee, \wedge, \neg, \bar{Q}^*_{SUP}, \bar{Q}^*_{INF})$ be the corresponding Boolean algebra of WVFs. Let $H : \mathcal{M} \to \bar{\mathcal{Q}}^*$ be any map from $\mathcal{M}$ to $\bar{\mathcal{Q}}^*$ such that $H(M) = \bar{Q}^*_M$ for all M in $\mathcal{M}$. Then H is a homomorphism.*

Proof Follows from the homomorphism between the De Morgan task and WVF algebras. ■

Theorem 6 *Let $(\mathcal{M}, \vee, \wedge, \neg, \mathcal{M}_{SUP}, \mathcal{M}_{INF})$ be a Boolean algebra of goal-reaching tasks, and let $\bar{\mathcal{Q}}^*$ be the set of optimal WVFs of tasks in $\mathcal{M}$. Then the Boolean algebra on $\bar{\mathcal{Q}}^*$ is isomorphic to the Boolean algebra on $\mathcal{M}$.*

Proof This follows from the isomorphism between $\bar{\mathcal{Q}}^*$ and $P(\mathcal{G})$, and that between $P(\mathcal{G})$ and $\mathcal{M}$. ■

Theorem 7 *Let $\mathcal{M}$ be a set of tasks and $\bar{\mathcal{Q}}^*$ the set of optimal $\bar{Q}$-value functions for tasks in $\mathcal{M}$. Denote ${}^{\epsilon}\bar{Q}^*_M$ as the ϵ-optimal $\bar{Q}$-value function for a task $M \in \mathcal{M}$ such that*

$$|\bar{Q}^*_M(s,g,a) - {}^{\epsilon}\bar{Q}^*_M(s,g,a)| \leq \epsilon \text{ for all } (s,g,a) \in \mathcal{S} \times \mathcal{G} \times \mathcal{A}.$$

Then for all M_1, M_2 in $\mathcal{M}$ and (s,g,a) in $\mathcal{S} \times \mathcal{G} \times \mathcal{A}$,

(i) $\left|[\bar{Q}^*_{M_1} \vee \bar{Q}^*_{M_2}](s,g,a) - \sup_{M \in \{M_1,M_2\}} {}^{\epsilon}\bar{Q}^*_M(s,g,a)\right| \leq \epsilon$

(ii) $\left|[\bar{Q}^*_{M_1} \wedge \bar{Q}^*_{M_2}](s,g,a) - \inf_{M \in \{M_1,M_2\}} {}^{\epsilon}\bar{Q}^*_M(s,g,a)\right| \leq \epsilon$

(iii) if $\mathcal{M}$ is bounded then,

$$\left|\neg\bar{Q}^*_{M_1}(s,g,a) - \left[\left({}^{\epsilon}\bar{Q}^*_{SUP}(s,g,a) + {}^{\epsilon}\bar{Q}^*_{INF}(s,g,a)\right) - {}^{\epsilon}\bar{Q}^*_{M_1}(s,g,a)\right]\right| \leq 3\epsilon$$

Proof

(i):

$$\begin{aligned}
&\left|[\bar{Q}^*_{M_1} \vee \bar{Q}^*_{M_2}](s,g,a) - \sup_{M \in \{M_1,M_2\}} {}^{\epsilon}\bar{Q}^*_M(s,g,a)\right| \\
&= \left|\sup_{M \in \{M_1,M_2\}} \bar{Q}^*_M(s,g,a) - \sup_{M \in \{M_1,M_2\}} {}^{\epsilon}\bar{Q}^*_M(s,g,a)\right| \\
&\leq \sup_{M \in \{M_1,M_2\}} \left|\bar{Q}^*_M(s,g,a) - {}^{\epsilon}\bar{Q}^*_M(s,g,a)\right| \\
&\leq \epsilon
\end{aligned}$$

(ii):

$$\begin{aligned}
&\left|[\bar{Q}^*_{M_1} \wedge \bar{Q}^*_{M_2}](s,g,a) - \inf_{M\in\{M_1,M_2\}} {}^{\epsilon}\bar{Q}^*_{M}(s,g,a)\right| \\
&= \left|\inf_{M\in\{M_1,M_2\}} \bar{Q}^*_{M}(s,g,a) - \inf_{M\in\{M_1,M_2\}} {}^{\epsilon}\bar{Q}^*_{M}(s,g,a)\right| \\
&\leq \inf_{M\in\{M_1,M_2\}} \left|\bar{Q}^*_{M}(s,g,a) - {}^{\epsilon}\bar{Q}^*_{M}(s,g,a)\right| \\
&\leq \epsilon
\end{aligned}$$

(iii): Let $\mathcal{M}$ be bounded. Then,

$$\begin{aligned}
&\left|\neg\bar{Q}^*_{M_1}(s,g,a) - \quad \left({}^{\epsilon}\bar{Q}^*_{SUP}(s,g,a) + {}^{\epsilon}\bar{Q}^*_{INF}(s,g,a)\right) - {}^{\epsilon}\bar{Q}^*_{M_1}(s,g,a)\right| \\
&= |\left(\bar{Q}^*_{SUP}(s,g,a) + \bar{Q}^*_{INF}(s,g,a)\right) - \bar{Q}^*_{M_1}(s,g,a) - \\
&\quad \left({}^{\epsilon}\bar{Q}^*_{SUP}(s,g,a) + {}^{\epsilon}\bar{Q}^*_{INF}(s,g,a)\right) - {}^{\epsilon}\bar{Q}^*_{M_1}(s,g,a)| \\
&= |\left(\bar{Q}^*_{SUP}(s,g,a) - {}^{\epsilon}\bar{Q}^*_{SUP}(s,g,a)\right) + \left(\bar{Q}^*_{INF}(s,g,a) - {}^{\epsilon}\bar{Q}^*_{INF}(s,g,a)\right) + \\
&\quad \left({}^{\epsilon}\bar{Q}^*_{M_1}(s,g,a) - \bar{Q}^*_{M_1}(s,g,a)\right)| \\
&\leq \left|\bar{Q}^*_{SUP}(s,g,a) - {}^{\epsilon}\bar{Q}^*_{SUP}(s,g,a)\right| + \left|\bar{Q}^*_{INF}(s,g,a) - {}^{\epsilon}\bar{Q}^*_{INF}(s,g,a)\right| + \\
&\quad \left|{}^{\epsilon}\bar{Q}^*_{M_1}(s,g,a) - \bar{Q}^*_{M_1}(s,g,a)\right| \\
&\leq 3\epsilon
\end{aligned}$$

■

References

Adamczyk, J., Tiomkin, S., and Kulkarni, R. Leveraging prior knowledge in reinforcement learning via double-sided bounds on the value function. *arXiv preprint arXiv:2302.09676*, 2023.

Alver, S. and Precup, D. Constructing a good behavior basis for transfer using generalized policy updates. In *Proceedings of the International Conference on Learning Representations*, 2022.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pp. 5048–5058, 2017.

Araki, B., Li, X., Vodrahalli, K., DeCastro, J., Fry, M., and Rus, D. The logical options framework. In *Proceedings of the International Conference on Machine Learning*, pp. 307–317. PMLR, 2021.

Barreto, A., Dabney, W., Munos, R., Hunt, J., Schaul, T., van Hasselt, H., and Silver, D. Successor features for transfer in reinforcement learning. In *Advances in neural information processing systems*, pp. 4055–4065, 2017.

Barreto, A., Borsa, D., Quan, J., Schaul, T., Silver, D., Hessel, M., Mankowitz, D., Zidek, A., and Munos, R. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *Proceedings of the International Conference on Machine Learning*, pp. 501–510. PMLR, 2018.

Birkhoff, G. *Lattice theory*, volume 25. American Mathematical Soc., 1940.

Borsa, D., Barreto, A., Quan, J., Mankowitz, D., Munos, R., Van Hasselt, H., Silver, D., and Schaul, T. Universal successor features approximators. *arXiv preprint arXiv:1812.07626*, 2018.

Camacho, A., Icarte, R. T., Klassen, T. Q., Valenzano, R. A., and McIlraith, S. A. Ltl and beyond: Formal languages for reward function specification in reinforcement learning. In *IJCAI*, volume 19, pp. 6065–6073, 2019.

Carvalho, W., Filos, A., Lewis, R. L., Singh, S., et al. Composing task knowledge with modular successor feature approximators. *arXiv preprint arXiv:2301.12305*, 2023.

Colas, C., Fournier, P., Chetouani, M., Sigaud, O., and Oudeyer, P. Curious: intrinsically motivated modular multi-goal reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, pp. 1331–1340, 2019.

Colas, C., Karch, T., Sigaud, O., and Oudeyer, P.-Y. Autotelic agents with intrinsically motivated goal-conditioned reinforcement learning: a short survey. *Journal of Artificial Intelligence Research*, 74:1159–1199, 2022.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the International Conference on Machine Learning*, pp. 1329–1338, 2016.

Foster, D. and Dayan, P. Structure in the space of value functions. *Machine Learning*, 49 (2-3):325–346, 2002.

Fox, R., Pakman, A., and Tishby, N. Taming the noise in reinforcement learning via soft updates. In *32nd Conference on Uncertainty in Artificial Intelligence*, 2016.

Furelos-Blanco, D., Law, M., Jonsson, A., Broda, K., and Russo, A. Hierarchies of reward machines. In *Proceedings of the International Conference on Machine Learning*, pp. 10494–10541. PMLR, 2023.

Glanois, C., Weng, P., Zimmer, M., Li, D., Yang, T., Hao, J., and Liu, W. A survey on interpretable reinforcement learning. *Machine Learning*, pp. 1–44, 2024.

Grätzer, G. *General lattice theory*. Springer Science & Business Media, 2002.

Grätzer, G. *Lattice theory: foundation*. Springer Science & Business Media, 2011.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. In *Proceedings of the International Conference on Machine Learning*, pp. 1352–1361, 2017.

Haarnoja, T., Pong, V., Zhou, A., Dalal, M., Abbeel, P., and Levine, S. Composable deep reinforcement learning for robotic manipulation. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 6244–6251. IEEE, 2018a.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

Heuillet, A., Couthouis, F., and Díaz-Rodríguez, N. Explainability in deep reinforcement learning. *Knowledge-Based Systems*, 214:106685, 2021.

Hunt, J., Barreto, A., Lillicrap, T., and Heess, N. Composing entropic policies using divergence correction. In *Proceedings of the International Conference on Machine Learning*, volume 97, pp. 2911–2920. PMLR, 2019.

Icarte, R. T., Klassen, T. Q., Valenzano, R., and McIlraith, S. A. Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73:173–208, 2022.

Infante, G., Jonsson, A., and Gómez, V. Globally optimal hierarchical reinforcement learning for linearly-solvable markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6970–6977, 2022.

Infante, G., Jonsson, A., and Gómez, V. Hierarchical average-reward linearly-solvable markov decision processes. In *ECAI 2024*, pp. 1325–1332. IOS Press, 2024.

Jothimurugan, K., Alur, R., and Bastani, O. A composable specification language for reinforcement learning tasks. *Advances in Neural Information Processing Systems*, 32, 2019.

Jothimurugan, K., Bansal, S., Bastani, O., and Alur, R. Compositional reinforcement learning from logical specifications. *Advances in Neural Information Processing Systems*, 34:10026–10039, 2021.

Kaelbling, L. P. Learning to achieve goals. In *International Joint Conferences on Artificial Intelligence*, pp. 1094–1099, 1993.

León, B. G., Shanahan, M., and Belardinelli, F. In a nutshell, the human asked for this: Latent goals for following temporal specifications. In *Proceedings of the International Conference on Learning Representations*, 2022.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

Li, X., Vasile, C.-I., and Belta, C. Reinforcement learning with temporal logic rewards. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3834–3839. IEEE, 2017.

Lillicrap, T., Hunt, J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. In *Proceedings of the International Conference on Learning Representations*, 2016.

Littman, M., Topcu, U., Fu, J., Isbell, C., Wen, M., and MacGlashan, J. Environment-independent task specifications via GLTL. *arXiv preprint arXiv:1704.04341*, 2017.

Ma, C., Ashley, D. R., Wen, J., and Bengio, Y. Universal successor features for transfer reinforcement learning. *arXiv preprint arXiv:2001.04025*, 2020.

Mendez, J. A. and Eaton, E. How to reuse and compose knowledge for a lifetime of tasks: A survey on continual learning and functional composition. *arXiv preprint arXiv:2207.07730*, 2022.

Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., et al. Learning to navigate in complex environments. In *Proceedings of the International Conference on Learning Representations*, 2017.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A., Veness, J., Bellemare, M., Graves, A., Riedmiller, M., Fidjeland, A., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Moore, A., Baird, L., and Kaelbling, L. Multi-value-functions: Efficient automatic action hierarchies for multiple goal MDPs. In *Proceedings of the international joint conference on artificial intelligence*, pp. 1316–1323, 1999.

Nangue Tasse, G., James, S., and Rosman, B. A Boolean task algebra for reinforcement learning. *Advances in Neural Information Processing Systems*, 33:9497–9507, 2020.

Peng, X., Chang, M., Zhang, G., Abbeel, P., and Levine, S. MCP: Learning composable hierarchical control with multiplicative compositional policies. *arXiv preprint arXiv:1905.09808*, 2019.

Qing, Y., Liu, S., Song, J., Wang, H., and Song, M. A survey on explainable reinforcement learning: Concepts, algorithms, challenges. *arXiv preprint arXiv:2211.06665*, 2022.

Qiu, W., Mao, W., and Zhu, H. Instructing goal-conditioned reinforcement learning agents with temporal logic objectives. *Advances in Neural Information Processing Systems*, 36: 39147–39175, 2023.

Ringstrom, T. J., Hasanbeig, M., and Abate, A. Jump operator planning: Goal-conditioned policy ensembles and zero-shot transfer. *arXiv preprint arXiv:2007.02527*, 2020.

Saxe, A., Earle, A., and Rosman, B. Hierarchy through composition with multitask LMDPs. *Proceedings of the International Conference on Machine Learning*, 70:3017–3026, 2017.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *Proceedings of the International Conference on Machine Learning*, volume 37, pp. 1312–1320. PMLR, 2015.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

Sutton, R., Barto, A., et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

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

Sutton, R., Modayil, J., Delp, M., Degris, T., Pilarski, P., White, A., and Precup, D. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 761–768. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

Tasse, G. N., James, S., and Rosman, B. Generalisation in lifelong reinforcement learning through logical composition. In *Proceedings of the International Conference on Learning Representations*, 2022.

Tasse, G. N., Jarvis, D., James, S., and Rosman, B. Skill machines: Temporal logic skill composition in reinforcement learning. In *Proceedings of the International Conference on Learning Representations*, 2024.

Taylor, M. and Stone, P. Transfer learning for reinforcement learning domains: a survey. *Journal of Machine Learning Research*, 10:1633–1685, 2009.

Thrun, S. Is learning the n-th thing any easier than learning the first? In *Advances in neural information processing systems*, pp. 640–646, 1996.

Todorov, E. Linearly-solvable Markov decision problems. In *Advances in Neural Information Processing Systems*, pp. 1369–1376, 2007.

Todorov, E. Compositionality of optimal control laws. In *Advances in Neural Information Processing Systems*, pp. 1856–1864, 2009.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Urain, J., Li, A., Liu, P., D'Eramo, C., and Peters, J. Composable energy policies for reactive motion generation and reinforcement learning. *The International Journal of Robotics Research*, 42(10):827–858, 2023.

Vaezipoor, P., Li, A. C., Icarte, R. A. T., and Mcilraith, S. A. Ltl2action: Generalizing ltl instructions for multi-task rl. In *Proceedings of the International Conference on Machine Learning*, pp. 10497–10508. PMLR, 2021.

van Niekerk, B., James, S., Earle, A., and Rosman, B. Composing value functions in reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, volume 97, pp. 6401–6409. PMLR, 2019.

Veeriah, V., Oh, J., and Singh, S. Many-goals reinforcement learning. *arXiv preprint arXiv:1806.09605*, 2018.

Voloshin, C., Verma, A., and Yue, Y. Eventual discounting temporal logic counterfactual experience replay. In *Proceedings of the International Conference on Machine Learning*, pp. 35137–35150. PMLR, 2023.

Watkins, C. *Learning from delayed rewards.* PhD thesis, King's College, Cambridge, 1989.

Yalcinkaya, B., Lauffer, N., Vazquez-Chanlatte, M., and Seshia, S. Compositional automata embeddings for goal-conditioned reinforcement learning. *Advances in Neural Information Processing Systems*, 37:72933–72963, 2024.

Yu, C., Zheng, X., Zhuo, H. H., Wan, H., and Luo, W. Reinforcement learning with knowledge representation and reasoning: A brief survey. *arXiv preprint arXiv:2304.12090*, 2023.