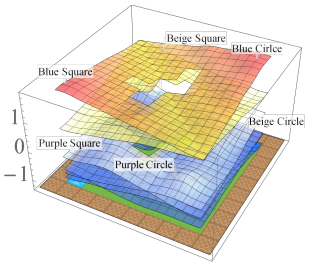
We thank the reviewers for their constructive feedback and hope to clarify and address their concerns in this response.

R*: Responses that are relevant to all reviewers



(i) **UVFAs may help with more complex settings. What happens if one applies the Boolean operators on them?** Note that our extended value functions (EVFs) are a subset of general value functions (GVFs) [4] and UVFAs ($Q(s, g, a, \theta)$) are their function approximators. So you can think of the function approximators of EVFs as a subset of UVFAs that have a structure useful for zero-shot composition. This structure is the decoupling of values per goal, and is illustrated here with the EVF of the "collect blue objects" task. We will add this explanation in the paper.

(ii) **The extensive set of assumptions limit the applicability (same terminal states for all tasks, same state space, binary rewards in terminal states).** Note that this work is mainly theoretical and follows previous theoretical work [1]. We show that under the same assumptions as [1] (Assump 1), we improve their result (optimal union and *approximate* intersection) by obtaining optimal union, intersection and negation (Thm 3). Note that Assump 1 does not require binary rewards in terminal states (also see discussion after Assump 1). Also please note that when previous works assume goal reaching tasks share the same transition dynamics, they mean formally the tasks also share the same absorbing states. If we want to adhere strictly to the theory, then in practice, one can have an action that the agent chooses to achieve goals. For example, in the four-rooms experiments, we have a 5th action for "stay", such that a goal position only becomes terminal if the agent chooses to stay in it. This represents the intuition that if an agent is at the goal location of a different task, and chooses to stay in it, then it has clearly chosen the wrong behaviour for the current task.

(iii) **General comment.** The usefulness of this line of work is that it shows how to compose value functions to guarantee zero-shot recovery of useful skills. In [1] these value functions are normal value functions ($Q(s, a)$) and in this work they are general value functions ($Q(s, g, a)$). Since these value functions have large bodies of work on learning them, this line of work focuses on their composition after learning to obtain combinatorial explosion of skills.

R1: (i) **Given the natural construction of a Boolean algebra from a set of goal states, the extension to task reward functions is quite straightforward.** Note that the Boolean algebra just formalises what previous works have been saying when they say union and intersection over tasks. The more important contribution here is the zero-shot composition (Thm 3) and the homomorphism between the Boolean algebra of tasks and Q-values (Cor 2).

(ii) **Could you evaluate in the sparse reward setting?** The function approximation experiment was in this setting.

(iii) **How does one guarantee that an optimal policy terminates in an absorbing state?** This is a standard assumption in infinite horizon/SSP problems, and is a requirement for policy/value iteration [2].

(iv) **Can these results be applied to the setting of [1].** Our zero-shot results (Thm 3) is in this setting (Assump 1).

R2: (i) **It is hard to estimate if the method will scale to much more complex problems, such as robotic manipulation.** Please see response R* (i,iii) above. EVFs are a subset of GVFs which can be viewed as learning N value functions. If you can learn one (e.g. via PPO) then you can learn them all [4]. Methods like hindsight experience replay [3] demonstrate this can be done efficiently using UVFAs with any suitable RL algorithm (e.g DQN, PPO, DDPG).

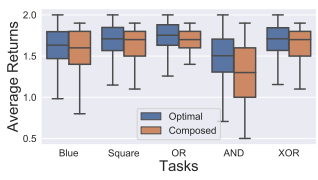
(ii) **Under broader impact, I think one should at least give a brief outline.** We will add the following: Our work is mainly theoretical, but is a step towards creating agents that can solve tasks through human-understandable Boolean expressions, which could one day be deployed in practical RL systems.

R3: (i) **In order to derive the policy the burden is on the engineer to describe the new task using Boolean operations between previously learned ones (not a learning approach).** Note that this is not a weakness, but rather a desirable feature of getting to the point where we can program RL agents to solve combinatorially more tasks than they learn, and do it in a human understandable way! This is the main motivation for this line of work.

(ii) **There's no relevant benchmark for lifelong learning used. The paper does not address the problems specific to lifelong learning.** Note that this line of works (see Sec 6) are steps towards lifelong learning by enabling agents to solve combinatorially more tasks than they learn. We use the same theoretical setting and experiment domain as [1].

(iii) **Authors do not quantitatively compare their results with (Van Niekerk, 2019)** We did. Please see Figure 3(c). Here we demonstrated the combinatorial benefit of having zero-shot negation and conjunction in addition to disjunction.

R4: (i) **Experiments with different absorbing states in the image-based domain.** Just as in [1], the experiments we did in the image-based domain was indeed with different absorbing states (See trajectory to blue circle in Figure 4(a,c) and more in source code). Also, here is the plot showing average returns (over 10k episodes) for the composed tasks across the random placements of agent and objects. We will add it to paper. Also thanks for the additional references which we'll incorporate.



[1] Van Niekerk et al., Composing Value Functions in RL, 2019; [2] Bertsekas, RL and Optimal Control, 2019; [3] Andrychowicz et al., Hindsight Experience Replay, 2017; [4] Sutton et al., Horde, 2011.