Reinforcement learning faces a scaling problem [CITE: XYZ]. As the state and action spaces grow, standard techniques, such as those canvassed in the previous section, falter. Yet real-world environments feature essentially infinite state and action spaces. If reinforcement learning is to provide satisfying explanations of rational behavior beyond the confines of the lab, it must tame this complexity [CITE: Sutton Barto, etc.].[1]

## 0.1 Cross-state generalization

Large state and action spaces pose difficulties for reinforcement learners because most states (or state-action pairs) are encountered at most once. That is, for most states, if the learner encounters that state, it will do so for the first time, and hence will not have had the opportunity to learn about it. In turn, the energy it spends learning from its visit is essentially wasted, since it will in all likelihood never see that state again. (Model-based learners, who can learn the value of an action without trying out that action, might seem to be in a better position. But models carry a cost of their own: storing them consumes memory, and making decisions using them takes time and is error-prone. These costs scale along with the size of the model. And on the assumption that the agent must learn its model by interacting with its environment, precisely the same problems arise as in the model-free case: to learn the dynamics of a state, that state has to be visited.)

This predicament suggests that the issue is one of generalization: the learner needs to generalize from the states it has encountered to the ones it has not, so that what it has learned about the one can be put to use in the other [CITE: Sutton and Barto]. The most straightforward way to do this is to have the agent learn *representations* of the state space: ways of grouping states together, such that two states that are grouped together count as the same from the learner's perspective.[2] Anything learned about one state will then automatically transfer to all states sharing the same representation. And any aspect of a state not relevant to how it is represented is treated as irrelevant for the purposes of decision-making.

As an example, consider the problem of deciding when to brake when approaching an intersection. We may assume that there is no cost to making it through the intersection unharmed, a small cost to stopping too early, and a large cost to stopping too late. Since this is a decision you are likely to have to make repeatedly, it is aptly modeled through the reinforcement learning framework.[3] If the light is green, no braking is needed. If the light is red, you should brake immediately. If the light is yellow, whether you should brake depends on your current speed and distance from the intersection: if you have the space to

[1]In addition, real-world environments are likely not to be Markov, so some extension of the framework is required here. See the discussion in [CITE: Sutton and Barto, ch. 17] for some options.

[2]Note that this concept of representation is broader than most conceptions of representations in philosophy. Nonetheless, the concepts are not entirely disconnected.

[3]Since your decision does not affect which choice situation you next face, the task is a so-called *contextual bandit* task (see [CITE: Sutton and Barto: 41]).

safely stop, you should brake, but if you don't, you should not. Besides the color of the light and your distance to the intersection, several other factors bear on your choice, though they might be less important: road and weather conditions, the presence and general behavior of other drivers, the number of passengers in your car, and so on. Many features of the situation, however, are not relevant to your decision at all: whether the birds in the nearby trees are singing, the name of the restaurant on the corner, etc. If these latter features are considered part of the state of the environment, it is exceedingly unlikely that you will find yourself in the very same state twice. But intuitively, states that differ only in whether nearby birds are singing should count as the same choice situation when the goal is to safely negociate the intersection. Even if we restrict our attention to a very minimal number of state variables, such as the color of the light and the distance to the intersection, we face a difficult problem: the distance variable can take infinitely many values, resulting in an infinite state space.[4] If the state $s_1 = (\texttt{light == green}, \texttt{distance == 22.162m})$ is treated as distinct from the state $s_2 = (\texttt{light == green}, \texttt{distance == 22.864m}$, then what is learned about $s_1$ will not transfer over to $s_2$, and vice-versa.

One way around this problem is to discretize the state space. In our example, this would involve dividing up the maximal possible distance (say, 100m) into $k$ chunks and treating distances that fall within the same chunk as identical for the purposes of decision-making.

This choice of discretization works because states that are close together along the `distance` dimension (and otherwise alike) are likely to call for the same response.

...

## 0.2 Hierarchical Reinforcement Learning

Another way to deal with large state or action spaces is through Hierarchical Reinforcement Learning (HRL). HRL replace actions with *options*: short-term policies that control behavior for part of the agent's interaction with the environment. At each time step, the agent can choose to execute an option. The option then governs the agent's actions until one of its termination conditions is met. Options are very flexible: they strictly generalize actions, since each action can be thought of as a one-step option, but they can also capture entire policies, if their termination conditions only include fulfilment of the goal. Obviously, one-step options afford no advantages over actions, while whole-environment options require knowledge of how to solve the task and are drastically insensitive to the fine-grained feedback gathered by the agent. Options are most useful when they are pitched at an intermediate level of grain: when they "chunk" together a handful of one-step options (i.e. actions). Whereas the generalization

[4]There is a further issue of how to integrate the essentially discrete temporal nature of MDPs with continuously varying variables. For present purposes, we may assume that the state variables are sampled at regular intervals, inducing a discrete temporal structure. A fuller treatment would require introducing the theory of continuous control [CITE: continuous control].

strategies mentioned in the previous section usually feature spatial abstraction (though the space in question may be somewhat abstract, such as color space), HRL features *temporal* abstraction.

[Say something about hierarchically composing options.]

To illustrate the HRL framework, consider an example from [CITE: Sutton: HRL].

INSERT FOUR ROOMS HERE

This environment consists of four rooms making up the four quadrants of a square. Each room is connected to the two adjacent rooms. The agent starts in the bottom left room and must make its way to the goal in the top right room.

In a standard reinforcement learning setting, the agent's actions are to move to adjacent cells, receiving a small negative reward for each step that does not lead to the goal and a large positive reward for reaching the goal. Significant learning occurs only once the goal has been reached for the first time. When the goal is reached, standard model-free algorithms propagate the prediction error backwards a few steps. Due to the wide range of possible states and actions, it may take a while before the agent first reaches the goal, and a while longer before action values are propagated far enough to influence behavior in the starting state.

By contrast, an options framework can enrich the problem with `get-to-the-door` options taking the agent to one of the adjacent doorways. It would take just two such options to reach the goal room, from which point making one's way to the goal using one-step options would be relatively easy. Thus, the right options can dramatically speed up learning.

[Say something about the drawbacks of options.]

The main challenge facing HRL, of course, is to learn options. In the four rooms example above, we assumed that the doorway options were given to the agent. We essentially gave the agent some hints about how to navigate its environment, using our own knowledge of which courses of action were likely to lead to success. Such a benevolent maker is of course not available in naturalistic settings, where the agent must discover for itself how to group one-step options into larger action sequences.