Ontogenetic and Phylogenetic Reinforcement Learning

Julian Togelius, Tom Schaul, Daan Wierstra, Christian Igel, Faustino Gomez, Jürgen Schmidhuber

Reinforcement learning (RL) problems come in many flavours, as do algorithms for solving them. It is currently not clear which of the commonly used RL benchmarks best measure an algorithm's capacity for solving real-world problems. Similarly, it is not clear which types of RL algorithms are best suited to solve which kinds of RL problems. Here we present some dimensions along the axes of which RL problems and algorithms can be varied to help distinguish them from each other. Based on results and arguments in the literature, we present some conjectures as to what algorithms should work best for particular types of problems, and argue that tunable RL benchmarks are needed in order to further understand the capabilities of RL algorithms.

1 Introduction

As defined by Sutton and Barto, any algorithm that can solve a reinforcement learning (RL) problem, which is defined by a (partially observable) Markov decision process, (PO)MDP, is an RL algorithm [19]. In the last few decades, a wide variety of algorithms have been used to successfully tackle RL problems in different guises. Surprisingly, these algorithms are based on very different principles. This is possibly due to the fact that they have been proposed and are actively studied within separate academic communities (e.g. machine learning, computational intelligence, computational neuroscience and control theory). There has been limited communication between these research fields, leading to insufficient analysis of the differences and similarities between these algorithms. It would therefore be a great boon to RL research to find a unified view, allowing us to understand the relative benefits of algorithms based on different principles.

At the same time, a large variety of RL problems has been defined and studied, varying from real-world continuous control problems to abstract discrete toy benchmarks, where real-world problems can be defined as problems that were not created with the purpose of testing RL algorithms. It is well known that some RL algorithms work well for some problems where other algorithms fail, but in many cases it is not clear what algorithms perform best under what conditions. The principal dimensions along which RL problems can vary are listed below.

The purpose of this short paper is to discuss some distinctions between types of RL problems and RL methods, especially between *ontogenetic* and *phylogenetic* methods, which in our experience is one of the distinctions that most clearly divides the disparate research communities concerned with RL in one form or another. We also make some conjectures about what types of methods would work best on what types of problems, and argue for the need of RL benchmarks that are tunable in important problem dimensions.

2 RL problem dimensions

Reinforcement learning problems may vary along multiple dimensions, for example:

- **Discrete vs. Continuous.** The environment's state, action and observation spaces can each be discrete, continuous or mixed.
- **Size and Dimensionality.** Apart from being continuous or discrete, the state, action and observation spaces may vary in their dimensionality. For instance the state can be represented by a single integer or by a visual scene. The size of discrete dimensions can vary from small (binary) to large (e.g. dictionaries).
- **State Space Structure.** There can be varying degrees of *structure* in the state space. Many benchmarks assume a certain locality (i.e. state transitions reach only a neighborhood of states) or have hierarchical properties. That structure is not necessarily ergodic, which thus allows for 'catastrophic' actions after which the agent cannot return to parts of the state space (e.g. a tabletop robot might fall off the table). There also exist exotic state representations based on, e.g., relations between logical predicates.
- **Stochasticity.** The problem can have varying degrees of stochasticity at different levels:
  - The state *transitions* may be stochastic. For example a robot's wheels might slip, so it does not know how far it will move when trying to move forward.
  - The *reward* function needs not be deterministic, for instance in a randomized game.
  - The *observations* may be stochastic, for example in the case of noisy sensor input.
  - There could be different *start states* drawn according to some random distribution.
- **Observability.** The environment can be *fully observable*, where the underlying state is directly accessible to the agent, or *partially observable*, where the agent can only make indirect, potentially stochastic *observations* of the state. This can impose a memory requirement on the agent as the only way to reliably infer the current state. In addition, observations can have varying degrees of redundancy, which in turn can make learning harder [21].
- **Generalization.** Observation representations may vary in the amount of meaningful structure encapsulating aspects of the transition model and enabling generalization to similar states.

Auszug aus: Künstliche Intelligenz, Heft 3/2009, ISSN 0933-1875, BöttcherIT Verlag, Bremen, **www.kuenstliche-intelligenz.de/order**