Proposed Research Program

Richard S. Sutton

March 9, 2020

My proposed research program is based on the idea of growing a complex agent from a small seed of basic principles and a single stream of experience interacting with an environment, an idea I call *constructive artificial intelligence*. The growth and construction is ongoing and continual—never ending and never completing—because the agent's computational resources, though large, are nevertheless tiny compared to the complexity of the environment, which after all contains many other agents. The agent faces a never-ending problem of how best to use its limited resources to predict and control its complex environment.

Constructive intelligence is all about predicting and controlling the experiential data stream. Everything the agent knows is a fact about that data; it searches for statistical patterns in it. The overarching issue that I am driving toward might be called *the problem of the small and the big*. The data consists entirely of temporally small stuff—primitive actions like muscle twitches and primitive observations like video frames—and yet an understanding of the data should consist primarily of temporally larger stuff—concepts like objects, events, people, places, and relationships among them. Crossing this abstraction gap between the small and the big is the main challenge that I propose to address. How can an agent construct the right abstractions to predict and control its environment?

I envision the agent's growth as as a search for structural elements of four types—*features, subproblems, options, and option models*—which together contribute to its effective operation. Candidate elements are continually generated and tested for utility, which is ultimately grounded in reward. The best candidate elements are retained, and the worst are discarded in favor of new candidates. The contribution of each element is carefully monitored, and there is a budget for each type of element. Newly discovered elements are made available to the construction of new candidate elements. The intended result is a continual increase in the quality and level of abstraction of the stable of elements, leading to improved performance (a greater rate of obtained reward). Although the size of the whole system is strictly limited—there is a fixed budget for each kind of element—hierarchical complexity should increase until computational resources are fully utilized. This is the vision of constructive artificial intelligence.

Key to constructive AI is that the search for good elements builds upon previously created elements, in a *virtuous cycle of discovery*. One such cycle is in the state features. The state features can be formed by a recurrent structure that is essentially an artificial neural network, though constructed by search processes involving generate and test in addition to stochastic gradient descent. New candidate features are generated (partly at random) and are tested by their contribution to prediction, control, or to the formation of other features that contribute to prediction or control.

The second major virtuous cycle that I envision is for the discovery of subproblems of feature attainment. As features are discovered by the first cycle, the most useful of them become the basis for subproblems of a specific form. Like the original problem, the subproblems seek to maximize reward, but they also balance this against reaching a state (time step) at which the feature is high. The subproblem is to behave so as to maximize the sum of the reward along the way to termination plus a bonus proportional to the feature value at the time of termination. Given this subproblem, its solution is well defined. That solution is a way of behaving together with a way of terminating—a pair known as an *option*. There are well established learning methods that can learn options and corresponding value functions given such a well-specified subproblem. Given the option, there are also established ways of learning environmental models of the consequences of executing the options. All these learning processes use the features and thus provide a basis for evaluating and discovering new features.

Let us describe the first two cycles—one discovering features and one discovering subproblems and options—and their proposed interaction one more time. The agent is tasked to continually search for new features. The features found to be most useful are protected from change and preferentially used as building blocks in constructing new candidate features, completing a cycle. The most useful features are then turned into subproblems of attainment. That is, the agent learns how to reach and terminate in states where the feature is high without having lost much reward. In solving each subproblem the agent learns a policy and a value function (an option). These learned functions are new users of features; their existence leads to a different and broader sense of which features are useful, completing a second cycle. For some of the learned options we also learn option models. These are predictions of what will happen to the reward and to all the state features if the option is executed. Each option model comprises many predictions (one for each state feature plus one more for the reward), each of which is a new user of features, leading again to a different and broader sense of which features are useful, and a *third* cycle. Finally, the option models are used for planning (by dynamic-programming-style methods like Dyna). Their utility in planning is also assessed empirically and used to evaluate the options and option models, completing a fourth cycle.

These are the ways in which I hope to find the abstractions in state (feature) and time (options) that enable superior prediction and control. Most important is that this strategy is domain general and open ended. It is not based on prior assumptions about the domain, and the abstractions would build one upon the other limited only by the agent's computational resources.

This research vision is ambitious, but much of the way has been prepared by work over the last 30 years. Dyna-style planning was introduced in 1990 [44,116] (numbers refer to the list of publication in my CV), and its extension to linear function approximation was introduced in 2008 [81]. Learning and planning with temporal abstraction was introduced in 1995 [106] and then fully developed (for the tabular case) in 1999 [26]. Generate and test ideas for feature construction are natural developments of my work on "random representations" [107], Kanerva coding [2], and step-size adaptation [109,108,150]. My student, Rupam Mahmood, took these ideas much further in his 2010 PhD thesis (see Chapters 11 & 12). And of course all the learning processes are based on temporal-difference learning [29,10] and, in particular, on its extension to off-policy learning [12,77], to general value functions [74], and to massive parallelism [14]. The idea that knowledge of the environment can be represented in option models has been explored and tested [89,140]. All these works are careful to cite and acknowledge prior work on similar topics. The vision of constructive artificial intelligence is ambitious, but also obvious. It is time, with better algorithms and greater computational resources, to give it another considered try.

Mahmood, A. R. (2010). *Incremental Off-policy Reinforcement Learning Algorithms*. University of Alberta PhD thesis.