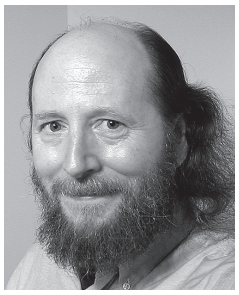
Interview with Richard S. Sutton



Richard S. Sutton was born in Ohio, and grew up in Oak Brook, Illinois, a suburb of Chicago. He received the B.A. degree in psychology from Stanford University in 1978, and the M.S. and Ph.D. degrees in Computer Science from the University of Massachusetts in 1980 and 1984. He worked for nine years at GTE Laboratories in Waltham as principal investigator of their connectionist machine learning project, and for three years at the University of Massachusetts in Amherst as a research scientist in the computer science department. In 1998-2002 Rich worked at AT&T Labs in Florham Park, New Jersey, and since August of 2003 he has been professor and iCORE chair of computing science at the University of Alberta. He is a fellow of the Association for the Advancement of Artificial Intelligence. Rich's research interests center on the learning problems facing a decision-maker interacting with its environment, which he sees as central to artificial intelligence. He is the author of the original paper on temporal-difference learning and, with Andrew Barto, of the textbook Reinforcement Learning: An Introduction. He is also interested in animal learning psychology, in connectionist networks, and generally in systems that continually improve their representations and models of the world.

KI: Dear Rich, 10 years have passed since the publication of your frequently used textbook „Reinforcement Learning: An Introduction" written together with Andrew Barto. Did you expect such a success at the time of writing?

Sutton: I think we did expect our book to become the standard textbook in the field. The surprise is more that the field has grown as big as it has – the book has been successful largely because the field has been successful.

We need methods for discovering knowledge

KI: What are in your opinion the most important advancements in the field since your book was published?

Sutton: Well, of course there have been many. In no particular order, the ones that come to mind most quickly are: policy-gradient methods, temporal abstraction methods such as options, and the new predictive ways of thinking about state. Also important are the new Monte Carlo search methods, such as are used in Computer Go. I am also impressed by some of the new applications, such as on autonomous helicopter flight and other robotics work.

KI: What are the biggest problems in RL that are most important to overcome?

Sutton: The biggest challenges may be primarily in abstraction methods, in creating ways of representing and learning about the world that capture high-level concepts, and yet relate to low-level sensation and action. Another way of looking at it is that we need systems that can represent large quantities of knowledge about the world, and that the most important knowledge is high-level and abstract. Reinforcement learning, so far, has been mainly about finding specific policies and value functions. But for the future we need to do much more than that. To achieve AI, we must have agents that know all sorts of things other than value functions and policies. We must have agents that know all the little facts about life, about physics and trees and people and houses, and everything else, and are able to apply it appropriately to solve a variety of problems as they come up. That is, to figure out how to get reward efficiently in a large variety of different situations.

We need agents that know all the little facts about life

Let me say a little more. What would it mean to address the problem of abstraction? I think the key thing is what is sometimes called the „signals to symbols" problem. Life gives us low-level signals – sensations and actions – and somehow we structure and represent them to achieve our goals. There is an awesome gap between the low-level signals and the high-level ways that we think about them. That is the main thing that we need to understand better in order to solve AI.

We need methods for discovering knowledge, for discovering structure, for finding the right features and options, and for maintaining all this knowledge in such a way that it will be ready to use when needed. Our AI agents have to take much more responsibility not just for acquiring knowledge but for maintaining it and structuring it, for continually checking whether it is still valid and useful. I guess we are a long way from this, but much of that is because we don't have the right framework in which to explore the issues. If we can find the right framework, then perhaps progress can be made rapidly.

KI: What are the most exciting research directions at the moment? And what are you currently working on?

Sutton: What I have been working on is a little bit of a technical problem in temporal-difference learning – the problem of off-policy temporal-difference learning with linear function approximation. This is something that I've been working on for more than a dozen years. In some sense it is the second half of the idea of temporal-difference learning. An essential part of the idea of temporal-difference learning is that you can learn from a sequence of experience that is incomplete. You can start observing a sequence, gain some information from the events that occur, and learn without actually seeing the final outcome that you are trying to

predict. This ability could be extremely useful for learning abstract knowledge about the world. But it turns out that this cannot reliably be done with the current temporal-difference algorithms, at least not when linear function approximation is used. And of course function approximation is essential to any large application of reinforcement learning, so if temporal difference learning doesn't work with function approximation, then it is much less useful. And so, I have been working on off-policy learning for a long time. There have been many partial solutions, but none that has been entirely satisfactory.

For me this has been all the more galling because my whole story about abstraction – about the use of options and of models of options as knowledge – relies on some form of off-policy learning. The natural way to learn about options is to use intra-option learning methods based on temporal-difference learning, but that would be off-policy learning, and so our current methods don't really work for them.

Our current techniques are sufficient to handle fairly general problems

All that having been said, I think we may have finally found a solution. This is new joint work with a bunch of other people, and the papers are still in preparation, so I won't say too much. But the general idea is to take a gradient-descent approach to temporal-difference learning, one that can be applied even in off-policy training. If a gradient-descent approach can be made to work, then we can get strong convergence assurances, even with nonlinear function approximation. The linear TD(lambda) learning algorithm is very simple and is guaranteed converge, but only when trained on-policy. There have been many extensions that can learn off-policy, but they are all more complicated. I'm thinking here of the least-squares methods that are of quadratic complexity as opposed to the linear complexity of TD(lambda). Anyway, it looks like we may have finally found a way to do temporal-difference learning with off-policy training while retaining the same order of simplicity as linear TD(lambda).

KI: Reinforcement Learning has its roots partly in the neuro- and behavioral sciences. Now there is a growing interest in using abstract RL learning rules in computational neuroscience. Do you expect significant progress in this domain in the near future? What is missing to develop a theory of reward-driven decision making in humans and animals?

Sutton: The work developing reinforcement learning as a computational theory of brain reward systems is very exciting. It seems like enormous progress is being made very rapidly and, yes, I expect this to continue. 10 years ago it was hypothesized that the dopamine system – the main brain reward system – learns according to temporal-difference learning algorithms and distributes temporal-difference error to the rest of the brain. This has proved to be remarkably accurate, and is now the standard model in neuroscience. It has been replicated many times in many different laboratories. It just seems to work – the brain reward systems really seem to follow the principles of temporal-difference learning. This is an amazing thing! Ever since there has been neuroscience, people have looked for analogs of theoretical, engineering ideas in the brain. Much of the time this was very speculative, and did not prove all that useful. Because of this, we should always be guarded and conservative, but still, we should watch for and be alert to the possibility of confluence between engineering and biology. To me, it looks like we have such a confluence in the temporal-difference models of brain reward systems. It is not universally accepted. There are alternative theories. However, the reinforcement learning theories have held up very well, and have, at least, contributed to theoretical sophistication in neuroscience.

Confluence in the temporal-difference models of brain reward systems

You ask what is missing to develop a theory of reward-driven decision making in humans and animals. That's a good question. Of course, there is a lot more to do, and in that sense there is a lot missing. But let me come at this question the other way. In an important sense we are already there. The theory of reinforcement learning already constitutes a theory of reward-driven decision-making in humans and animals. It is not a complete theory, but I think, yes, it deserves to be considered a psychological as well as an engineering theory.

KI: There is a zoo of different approaches to technical RL: Temporal difference learning, policy gradient methods, direct policy search algorithms etc. Do you have rules of thumb when to use which approach?

Sutton: My rules of thumb are extremely simplistic – I guess because I tend to focus on the general problem of AI rather than particular applications. Thus, my rule of thumb is to use Sarsa if possible, and actor-critic methods when pressed. By actor-critic methods, I would now mean the newer policy-gradient methods, using in conjunction with an estimated state-value function. I tend to use linear function approximation with constant step sizes and epsilon-greedy action selection. In other words, I use the simplest possible methods that I can understand very well. For my purposes, that is usually sufficient.

KI: There have been recent studies showing good results of black-box optimization methods applied for direct policy search. I know that you are very skeptical about such methods. Why do you think that adopting black box optimization techniques for RL is not a promising direction?

Sutton: The first problem is that the notion of direct policy search is not clear. Different people have used the term for different purposes. Some of these are probably sensible, and others are not. Unfortunately, I think the term may be hopelessly compromised, and it may not be possible to establish a clear consistent definition for it.

Leaving aside the name, what about the ideas? One version of the black-box idea is that you don't need to know anything about what actions were performed, or what states were encountered. According to this idea you just try the policy for a while and measure overall performance. This is what I think is not promising. It is a very general technique, and it can work, but it is not promising because it is inherently inefficient. One reason for the inefficiency is that you're not taking advantage of the state property, of the Markov property. Of course, this is both a strength and a weakness. If you do not have good state information, then it is problematic to try

to use it. But if you do have state, then one can learn much more efficiently if you take advantage of it. One strong way of taking advantage is to use temporal-difference learning. But even if one does not do that, one can still learn much more efficiently by taking advantage of state. For example, using valued functions one can assign credit to particular time steps and to the actions taken at them. Using eligibility traces one can assign credit to the actions actually taken and not to the ones that just might have been taken. Whereas, if you just look at an overall agent, and not the specific events of its life, then credit can only be assigned only very generally and imprecisely, which makes learning inefficient.

assigning credit to states is important

What it comes down to is that I think experience is very important. To assign credit as accurately as possible, one has to pay attention to what actions were made and what states were encountered. Yet some direct policy search methods ignore all that, they ignore the actual experience of the agent altogether, only noting its policy and overall performance.

KI: Still most RL methods consider RL problems restricted to simple problem types (finite state and action spaces, MDPs). What is in your opinion the most promising approach to solve general RL problems?

Sutton: In contrast with many other researchers, I imagine that our current, best-practice techniques are sufficient to handle fairly general problems. I think that we have methods for handling function approximation (and thus hidden state), for mixing model-free and model-based methods, and even for abstraction in full-time and state. I would acknowledge that it is not always easy or immediate to get good results from these techniques. Parameters must be set and domain knowledge must be incorporated, and it takes some experience before it becomes easy to do these things. But why should we expect it to be easy? Perhaps one of the biggest problems in our field is that we think it should be easy, and then we are quick to complain when it is not. Reinforcement learning is addressing a hard problem, ultimately the problem of making an artificial intelligence and understanding the human mind. Why should we expect it to be easy?

RL is addressing a hard problem

KI: In contrast to the progress in RL theory, the current state-of-the-art in empirically evaluating RL algorithms seems not to be well developed. Many papers consider simple toy problems, the benchmarks tasks you used in your book are still the standard for comparing RL methods - but one may argue that after 10 years all poles are balanced and all cars are on top of the hill. What do you think? Is evaluating RL algorithms particularly difficult? What has to be done to improve benchmarking?

test problems must be programs

Sutton: I'm glad you asked that, because we have been making a major effort over the last few years to improve the standard of empirical research via software and community development. Brian Tanner and Adam White, in particular, have been pursuing this as part of their PhD research at the University of Alberta in the form of the RL-Glue and RL-Library open-source projects. Evaluating algorithms is more difficult in reinforcement learning than in other kinds of machine learning because the test problems must be programs. One cannot simply have files with training sets and test sets. One needs programs that can respond to whatever the learning algorithm does. The RL-Glue project is both a standard protocol for interconnecting agent programs and environment programs, and a collection of software implementing the protocol in such a way that the agent and environment can be written in different languages and can even be running in different places over the internet. The RL-Library is a growing collection of standard implementations of environments, agents, and benchmarks, so that people don't have to re-implement everything, and can more easily make valid comparisons with others' work.

The existence of standards, however, is not enough. The community must still choose to use them. I hope that anybody out there who is doing empirical work in reinforcement learning takes a close look at what we have done to see if it can't make it easier for them to do a good job.

KI: What real-world applications of RL do you find most impressive?

Sutton: Autonomous helicopter acrobatics. Game playing in Go, checkers and, still, backgammon. There are a lot of new real-world applications coming out all the time now, but I am still impressed at these other applications because we can clearly see a high-level of performance that had not been possible to reach in any other way.

KI: Do you have plans for a second, extended edition of your textbook?

Sutton: Yes, but don't hold your breath. It is probably still a couple of years away.

Interview by Verena Heidrich-Meisner