Editor: Stratis Gallopoulos, stratis@hpclab.ceid.upatras.gr
Send announcements to cchweh@computer.org

NEURO-DYNAMIC PROGRAMMING

Reviewed by George Cybenko

Neuro-Dynamic Programming, by Dimitri P. Bertsekas and John Tsitsiklis, Athena Scientific, Belmont, Mass., 1996, 512 pp., ISBN 1-886529-10-8, $84.00 (hardcover).

Over the past 10 years, machine learning has experienced a remarkable renaissance. Drawing researchers and practitioners from computer science, electrical engineering, operations research, statistics, psychology, physics, and mathematics, machine learning uses ideas and results from many areas of specialization. At the same time, nonspecialists have had difficulty penetrating the veils of jargon and hyperbole that make machine learning difficult to understand and evaluate. Enter *Neuro-Dynamic Programming*. This remarkable monograph gets my highest recommendation in all categories—readability, completeness, exposition, examples, and references.

Dimitri Bertsekas and John Tsitsiklis are no-nonsense electrical engineering systems professors at MIT with impeccable mathematical and computational credentials. In their own words, when they began to study the machine-learning field, their first impression was that "the new methods were ambitious, overly optimistic and lacked firm foundation." Based on their detailed field study, this monograph documents the mathematical foundations and substance of reinforcement learning. Bertsekas and Tsitsiklis write in the preface,

> Three years later, after a lot of study, analysis, and experimentation, we believe that our initial impressions were largely correct. This is indeed an ambitious, often ad hoc, methodology, but for reasons that we now understand much better, it does have the potential of success with important and challenging problems.... Furthermore, the methodology has a logical structure and a mathematical foundation, which we systematically develop in this book.

Machine learning has become a large field, and it is convenient to think of it as consisting of two main subareas—learning of cognitive tasks and learning of behaviors. Cognitive learning is mathematically rooted in statistical pattern recognition. The basic problems revolve around grouping and classifying patterns—the fundamental problems of cognition. Learning behaviors is more complex. Behaviors, in the popular sense, consist of perceiving some world state and performing an action that transforms the current state into a new state. An objective or cost criterion dictates the choice of action. The technical difficulty arises because the cost criterion depends on current and future actions, so you cannot easily and explicitly evaluate it based solely on the current state and action. Accordingly, an action's immediate cost acts merely as a reinforcement and generally does not constitute a global objective.

Consider, for example, the popular computer game Tetris. The "world state" of Tetris consists of the current board configuration with the current (and possibly next) piece to place on the board. A Tetris player's "action" involves rotating and dropping the piece into the game board, with the goal of clearing the board and moving onto the next level. However, this goal is difficult to quantify to guide the player in making the best rotation and placement of the current piece. A further complication arises because of the randomly selected pieces, which disallows the deterministic prediction of future system states. *Neuro-Dynamic Programming* carefully develops this example and many others, such as machine maintenance and repair (operations research), dynamic channel allocation (communications), and parking (everyone is interested in this application!).

Many researchers in behavioral machine learning have recognized that such problems are most naturally modeled as Markov Decision Processes—standard models in stochastic-control and operations research. Roughly speaking, an MDP is a Markov chain in which actions determine transition probabilities. That is, given a state, an action in that state dictates transition probabilities to the next state. Each such state-action pair results in a cost that can be a random variable. The goal is to operate such a process, minimizing the average, discounted, or other additive accumulated cost over the process's lifetime.

By the mid-1960s, Richard Bellman and R. Howard had already laid down the mathematical formulations and computational procedures for solving these problems, which are routinely studied in the operations-research and stochastic-control communities. Recent machine-learning research brings ideas for dealing with two major challenges:

- *The curse of dimensionality.* This arises from the astronomical number of states that will exist in many inter-

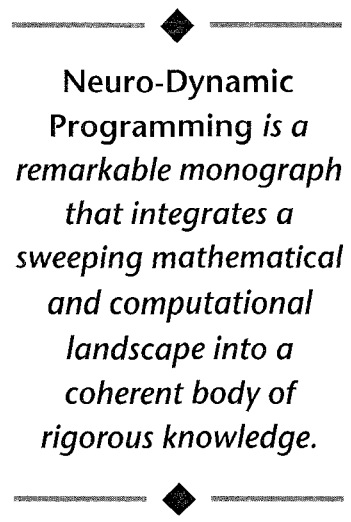
esting problems. In the Tetris example, the state space consists of all legal board positions and new pieces. In another example, a queuing system with m queues that have buffers of size n, would have n^m possible states. For such systems, exactly representing the state space and evaluating possible actions for the states is impossible.

- *The curse of modeling.* This arises from the difficulty of estimating, in many systems, all model parameters, such as transition probabilities and immediate cost. It might be desirable to use an online system or a simulator to generate appropriate actions from "experience," largely because size constraints disallow explicit system representation.

For instance, humans learn complex multistep tasks such as driving, playing a sport, and basic muscular controls by exploring the state and action spaces, observing results, and iteratively improving performance—not by consciously building a comprehensive, probabilistic, and dynamic model. *Neuro-Dynamic Programming* strives to make models and algorithms for such adaptive, learning systems mathematically sound. Bertsekas and Tsitsiklis accomplish this by developing the scientific foundations of the many recent developments in behavioral machine learning that take major steps toward dealing with the above two challenges.

Neuro-Dynamic Programming is almost self-contained, owing to four foundational chapters. The first, a short orientation, paints the big picture. The second, about dynamic programming, covers the basic ideas behind MDPs and classical computational approaches to solving them. The third chapter reviews results about neural networks from the point of view of approximation, developing both concepts and computational methods. The fourth deals with stochastic approximation in the spirit of Robbins-Monro methods.

These foundational chapters are small treasures themselves. They cover the basics in a rigorous and scholarly manner. At the same time, distilling modern discrete stochastic control into one 45-page chapter leaves little room for developing intuition or examples. The same can be said for the chapters on neural networks and stochastic approximation. I think a reader would have trouble starting from scratch and reading these chapters without some intuition or experience gained from previous efforts. However, a mathematically sophisticated scientist or engineer might try this "fire hose" treatment and succeed. Each chapter has a "Notes and Sources" section that can be used for historical, background, and supplementary material. The chapter on stochastic approximation ideas, titled "Stochastic Iterative Algorithms," has particular merit in that it covers the topic rigorously but without the excessive generality and formalism found in most modern treatments.



Bertsekas and Tsitsiklis begin their synthesis in Chapter 5 and continue to expand on the foundations for the remainder of the monograph. The results are beautiful and rigorous. In a nutshell, they cover

- neural-network approximation ideas used to build low-complexity approximations of the "cost to go" functions needed to evaluate optimal actions or behaviors,
- stochastic approximation ideas that underlie the iterative and the model-free methods used to estimate those approximations, and
- a large number of previously isolated special cases and results.

Methods such as Q-Learning, temporal-difference learning, and many variants of so-called heuristic dynamic programming also fit in the general framework the authors develop. Extensions to dynamic games and the various comprehensive examples previously mentioned fill out the rest of the book.

Neuro-Dynamic Programming is a remarkable monograph that integrates a sweeping mathematical and computational landscape into a coherent body of rigorous knowledge. The topics are current, the writing is clear and to the point, the examples are comprehensive, and the historical notes and comments are scholarly. The book is appropriate for a graduate-level seminar course or research reference, but given the pace of development and the lack of a problems section and truly introductory material, it would be difficult to use as a beginning graduate-level, let alone undergraduate, textbook. Bertsekas and Tsitsiklis have performed a Herculean task that generations to come will study and appreciate. I strongly recommend it to scientists and engineers eager to seriously understand the mathematics and computations behind modern behavioral machine learning. ♦

George Cybenko *is the Dorothy and Walter Gramm Professor of Engineering at Dartmouth College and Editor-in-Chief of* IEEE CS&E; *gvc@dartmouth.edu.*