A Reinforcement Learning Framework for Combinatorial Optimization

Justin A. Boyan
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
email: jab@cs.cmu.edu

The combination of reinforcement learning methods with neural networks has found success on a growing number of large-scale applications, including backgammon move selection (Tesauro 1992), elevator control (Crites & Barto 1996), and job-shop scheduling (Zhang & Dietterich 1995). In this work, we modify and generalize the scheduling paradigm used by Zhang and Dietterich to produce a general reinforcement-learning-based framework for combinatorial optimization.

The problem of combinatorial optimization is simply stated: given a finite state space X and an *objective function* $f : X \to \Re$, find an optimal state $x^* = \text{argmax}_{x \in X} f(x)$. Typically, X is huge, and finding an optimal x^* is intractable. However, there are many effective heuristic algorithms that attempt to exploit f's structure to locate good optima. One important class of such algorithms is based on hillclimbing (HC). HC makes use of a neighborhood structure on X. In its simplest form, HC works by starting at a random state $x = x_0$, then repeatedly considering transitions from x to a random neighbor x'. When $f(x') \geq f(x)$, the transition is accepted, and x is set to x'. HC terminates upon reaching a local maximum.

HC corresponds exactly to following a greedy trajectory through a generalized Markov Decision Process (MDP). This MDP may be defined as follows: state space = X; uniform stochastic transitions defined by the neighborhood structure; action space = {Accept, Reject}; and $\text{Reward}(x \to x') = f(x') - f(x)$. In this context, simulated annealing and other popular approaches to improving HC can be seen as poor approximations to the correct, principled approach: solving the MDP by learning its *optimal value function* V^*. V^* maps each state x to the discounted return expected when optimal Accept/Reject decisions are made:

$V^*(x) = \max \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathrm{Reward}(\mathrm{x_t} \to \mathrm{x_{t+1}}) | \mathrm{x_0} = \mathrm{x}\right]$

(We set γ slightly less than 1.) Once learned, V^* can be used in hillclimbing or simulated annealing as a plug-in replacement for the objective function $f(x)$. Unlike $f(x)$, however, V^* "sees past" local optima: greedy actions taken with respect to V^* maximize $f(x)$ *globally.*

The value function V^* gives a principled way to choose actions in an optimization domain. Our belief is that in practical large-scale domains, V^* will have significant underlying structure, allowing a useful approximation to be learned. There are several reasons for optimism:

- All relevant features of state x can be given as input to the function approximator. (By contrast, standard hillclimbing and simulated-annealing techniques cannot incorporate structural information about the state space.)
- During learning, extrapolation by the function approximator can be used to help guide exploration of the space, focusing successive searches on regions identified as promising during previous runs.
- Even a poor approximation to V^* may aid optimization. In this framework, $V^*_{\text{approx}} \equiv 0$ produces regular hillclimbing; any deviation from 0 which correlates with long-term reward should improve optimization performance.

We are using two algorithms to approximate V^*: Tesauro's variant of TD(λ) for control (Tesauro 1992), and an original algorithm called ROUT (Boyan & Moore 1996). We are also exploring several alternative MDP formulations to the one sketched above. Our poster will contrast our frameworks with that of Zhang and Dietterich; discuss the types of problem that this approach should benefit; and present results on several large-scale optimization problems, including VLSI channel routing and an information retrieval task.

Acknowledgments: Thanks to Andrew Moore, Michael Littman, Wei Zhang and the anonymous reviewers.

References

Boyan, J. A., and Moore, A. W. 1996. Learning evaluation functions for large acyclic domains. In Saitta, L., ed., *ICML-13.* Morgan Kaufmann.

Crites, R., and Barto, A. 1996. Improving elevator performance using reinforcement learning. In Touretzky, D.; Mozer, M.; and Hasselno, M., eds., *NIPS-8.*

Tesauro, G. 1992. Practical issues in temporal difference learning. *Machine Learning* 8(3/4).

Zhang, W., and Dietterich, T. G. 1995. A reinforcement learning approach to job-shop scheduling. In *IJCAI-95.*