Q-Learning for Bandit Problems

M.O. Duff

CMPSCI Technical Report 95-26
March, 1995

Q-Learning for Bandit Problems

(CMPSCI Technical Report 95-26)

Michael O. Duff
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
duff@cs.umass.edu

March 24, 1995

Abstract

Multi-armed bandits may be viewed as decompositionally-structured Markov decision processes (MDP's) with potentially very large state sets. A particularly elegant methodology for computing optimal policies was developed over twenty ago by Gittins [Gittins & Jones, 1974]. Gittins' approach reduces the problem of finding optimal policies for the original MDP to a sequence of low-dimensional stopping problems whose solutions determine the optimal policy through the so-called "Gittins indices." Katehakis and Veinott [Katehakis & Veinott, 1987] have shown that the Gittins index for a task in state i may be interpreted as a particular component of the maximum-value function associated with the "restart-in-i" process, a simple MDP to which standard solution methods for computing optimal policies, such as successive approximation, apply. This paper explores the problem of learning the Gittins indices on-line without the aid of a process model; it suggests utilizing task-state-specific Q-learning agents to solve their respective restart-in-state-i subproblems, and includes an example in which the online reinforcement learning approach is applied to a simple problem of stochastic scheduling—one instance drawn from a wide class of problems that may be formulated as bandit problems.

1 Introduction

Reinforcement learning algorithms, such as the method of temporal differences (TD) [Sutton, 1988] and Q-learning [Watkins, 1989], were originally advanced as models of animal learning, motivated and inspired by the behavioral paradigms of classical and instrumental conditioning. These algorithms have subsequently proved useful in solving certain problems of prediction and control encountered by general adaptive real-time systems or agents embedded in stochastic environments. Supporting theory and applications have reached a stage of development that is relatively mature. Connections have been established with stochastic dynamic programming and heuristic search [Barto *et al*, 1990 & 1991], and a mathematical framework, grounded in the classical theory of stochastic approximation, has led to new and improved proofs of convergence [Jaakkola *et al*, 1994], [Tsitsiklis, 1994]. Researchers have customarily focused their attention upon asymptotic learning of maximally-efficient strategies, and not on the "optimal learning" of these strategies. The most successful applications have been to large, complex problems for which the computational effort required by traditional engineering methods would be unduly burdensome and perhaps unjustified, given that in many cases only approximate models of the underlying systems are known [Tesauro, 1992], [Crites, 1995].

This paper examines a class of problems, called "bandit" problems, that is of considerable practical significance. One basic version of the problem concerns a collection of N statistically independent reward processes (a "family of alternative bandit processes") and a decision-maker who, at each time $t = 1, 2, \ldots$, selects one process to "activate." The activated process yields an immediate reward and then changes state; the other processes remain "frozen" in their current states and yield no reward. The decision-maker's goal is to splice together individual reward processes into one sequence of rewards having maximum expected discounted value.

The size of the state sets associated with bandit problems may typically be of such magnitude as to overwhelm straightforward methods of solution. These large state sets, however, do possess a particular Cartesian-product structure and independence under various control actions, and one may exploit these defining characteristics. In fact, proof that bandit problems can be decomposed into simpler, low-dimensional subproblems— in effect, rendering problems for which previous approaches had exponential complexity

into problems with solutions of linear complexity— has been rigorously established by Gittins [Gittins & Jones, 1974]. In this paper I will show how Q-learning can be integrated with the Gittins approach to solve bandit problems online in a model-free way.

After reviewing the bandit problem formulation, this paper notes the complexity of computing optimal policies for a family of alternative bandit processes by modeling the family, straightforwardly, as one large Markov decision process. This is followed by a discussion of Gittins' approach, which is a comparatively efficient and elegant method with a number of interesting interpretations, one of which allows Q-learning to be applied. (A brief and informal summary of TD(0) and Q-learning is provided in the Appendix.)

The main contribution of this paper appears in Section 6, where the central conceptual argument is summarized and the implementational details of a reinforcement learning algorithm are presented. This is followed by several examples, as well as a discussion of important generalizations of the basic bandit formulation to cases of practical interest.

Finally, this paper concludes by observing that the archetypal multi-armed bandit problem, in which policies map histories to arm-selections, captures the essence of the problem of optimal learning— the algorithm presented in Section 6 may be interpreted as a method for learning how to learn optimally.

2 Bandit Problems

Suppose there exist N stochastic processes $\{x_i(k)\}$, $i = 1, 2, ..., N$, whose values are members of a countable set. At each stage, k, a decision maker chooses an action, $a_k \in \mathcal{A} = \{1, 2, ..., N\}$.

Supposing that $a_k = j$, then the state $\underline{x} = (x_1(k), ..., x_N(k))$ evolves according to

$$\begin{aligned} x_i(k+1) &= x_i(k) \quad i \neq j \\ x_j(k+1) &= f_j(x_j(k), w_j(k)), \end{aligned}$$

where $w_j(k)$ is a random disturbance depending on $x_j(k)$ but not on prior disturbances. For example, for Markov transitions, the state evolution is governed via $Pr\{x_j(k+1) = y\} = P_{x_j(k),y}$, where P is a pre-specified transition matrix. This is the case considered henceforth (Figure 1).

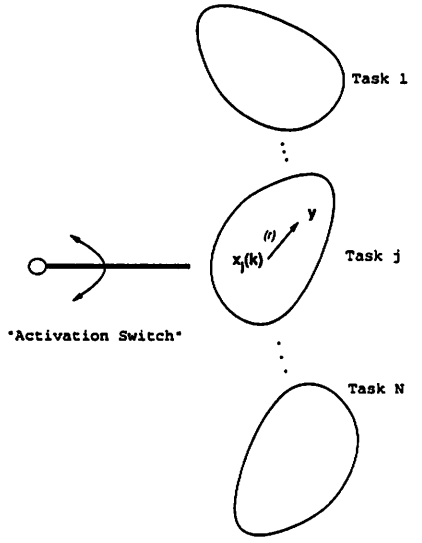


Figure 1: FABP schematic.

The goal is to choose a sequence of actions $\{a_k\}$ to minimize the expected value of the infinite-horizon discounted return:

$$\sum_{k=0}^{\infty} \gamma^k R_{a(k)}(x_{a(k)}(k)),$$

where $R_a(\cdot)$ is a bounded reward function and $\gamma \in (0,1)$ is the discount factor. Hence, the decision maker is essentially switching between the component processes' reward streams; activating a given process causes it to change state, while other component-process states remain "frozen." [1]

This is one version of the "multi-armed bandit" problem. In the literature, each component process is referred to as a *bandit process*, while the entire collection of candidate processes is termed a *family of alternative bandit processes* (FABP).

The multi-armed bandit problem appears to have originated with the paper by Thompson [Thompson, 1933]. In a highly-influential paper, Robbins

[1] In some versions of this problem, at each time k, one also has the option of retiring permanently and receiving a one-time-only reward $\gamma^k M$. M provides a useful parametrization for certain derivations or interpretations of the Gittins index, which will be reviewed in Section 4 (note that for M sufficiently small, the retirement option can be excluded).

[Robbins, 1952] initiated systematic study of bandit problems that emphasized strategies for which asymptotic "loss" tends to zero. Bellman [Bellman, 1956] adopted a Bayesian formulation for the infinite-horizon discounted case, and Gittins and Jones [Gittins & Jones, 1976] generalized Bellman's process model to the one given above. Dubins and Savage [Dubins & Savage, 1976] credit F. Mosteller with coining the term "two-armed bandits."

This term and its multi-armed generalization refer to the Bayesian adaptive control problem of selecting a sequence of plays on a slot machine that has several arms corresponding to different but unknown probability distributions of payoff. One may identify the conditional probabilty distributions of success probabilites of the respective arms given the observed past history up to stage k with the process state $\{x_i(k)\}$ given above. The action of pulling an arm elicits an immediate reward or payoff as well as a Bayes-rule update (which is Markov) of the arm's success probability.

The slot machine example highlights a key feature of multi-armed bandit problems, namely, it may be prudent to sacrifice short-term reward for information gain that will allow more informed future decisions. Thus, Whittle [Whittle, 1982] has claimed that a bandit problem "embodies in essential form a conflict evident in all human action." This "exploration versus exploitation" trade-off, a recurring theme in sequential experimentation and adaptive control, makes the general bandit problem a challenging one.

3 Families of Alternative Bandit Proceses as Markov Decision Processes

3.1 Markov Decision Processes

Consider a system whose dynamics are described by a finite state Markov chain with transition matrix P, and suppose that at each time step, in addition to making a transition from state $x_t = i$ to $x_{t+1} = j$ with probability P_{ij}, the system produces a randomly determined reward, r_{t+1}, whose expected value is $R(i)$. The *evaluation function*, V, maps states to their expected, infinite-horizon discounted returns,

$$V(i) = E\left\{\sum_{t=0}^{\infty} \gamma^t r_{t+1} | x_0 = i\right\},$$

and $V(\cdot)$ may also be shown to uniquely satisfy a system of linear equations describing local consistency: for $i = 1$ to N,

$$V(i) = R(i) + \gamma \sum_j P_{ij} V(j). \tag{1}$$

In addition, for each state i, suppose that there is a set of feasible actions, $\mathcal{A}_i$, from which to choose, and that choosing an action, a, determines the transition probabilities and reward distribution associated with that state. The resulting Markov chain is called a Markov decision process (MDP). A (stationary) *policy* is a mapping of states to actions, and one may think of P above as the transition matrix associated with some particular policy. An *optimal* policy is one that optimizes the value function over all states; the associated *optimal* value function is the unique solution to Bellman's optimality equation: for $i = 1$ to N,

$$V(i) = \max_{a \in \mathcal{A}_i} \left[R(i,a) + \gamma \sum_{j=1}^{N} P_{ij}(a) V(j) \right]. \tag{2}$$

An optimal policy is determined by the optimal value function through Equation 2; that is, if V is the optimal value function, then for each state i, an optimal action for that state is the action that achieves equality in Equation 2.

Standard methods for solving Bellman's equation include value iteration, policy iteration, and linear programming—see, for example, [Ross, 1983] or [Bertsekas, 1987].

3.2 FABP's as MDP's

This section adopts a perspective from which a family of alternative bandit processes, such as that described in Section 2, can be viewed as a standard Markov decision process. Computing optimal policies for bandit problems by applying standard methods to their associated MDP's, however, is in general ill-advised.

Consider the bandit problem of Section 2 with N tasks, each with state space S. The overall state of the FABP is then an element of S^N, and the action of activating a given task generates an immediate reward and causes the FABP-state to change in a Markovian fashion. Hence, ignoring the special

structural constraints satisfied by FABP's, they are simply standard MDP's with potentially rather large state-sets. Transition matrices are $|S|^N$-by-$|S|^N$, and standard methods for computing optimal policies have complexity of order $O(|S|^{cN})$, where c is a small constant.

But non-activated tasks do not change state, and rewards received depend only upon the state of the active task. These features may naturally lead one to conjecture the existence of a decompositionally-defined optimal policy whose determination requires work on the order of $O(N|S|^c)$. This is what researchers mean when they say, for example, that "the multi-armed bandit problem was solved, after puzzling researchers for thirty-years, by Gittins and Jones," [Walrand, 1988]. The next section provides a summary of the Gittins-Jones solution.

(*Aside*: For MDP's in general, the problem of computing optimal value functions may alternatively be viewed as a problem of constructing transition-matrix/expected-reward-vector pairs, $(P_{\mu^\bullet}, R_{\mu^\bullet})$, yielding optimal values for V in Equation 1; that is, the optimization procedure seeks $(P_{\mu^\bullet}, R_{\mu^\bullet})$ such that $V = (I - \gamma P_{\mu^\bullet})^{-1} R_{\mu^\bullet}$ is maximized, where the choice of $(P_{\mu^\bullet}, R_{\mu^\bullet})$ is constrained in the following way:

Let P_{a_i} be the transition matrix associated with taking action i in *every* state, and let R_{a_i} be the corresponding entry of the expected reward vector under action i. (If action i is inadmissible for a given state, the corresponding entries in P_{a_i} and R_{a_i} are null.) Then each row of $P_{\mu^\bullet}$ must be set to the corresponding row of one of the P_{a_i}'s, and the corresponding entry of $R_{\mu^\bullet}$ must be consistent with this choice; *i.e.*, if the j^{th} row of $P_{\mu^\bullet}$ is chosen to be the j^{th} row of P_{a_i}, then the j^{th} entry of $R_{\mu^\bullet}$ must be chosen to be the j^{th} entry of R_{a_i}.

For the case of n states and m possible actions in each state, this implies m^n possible candidates for $(P_{\mu^\bullet}, R_{\mu^\bullet})$, and the goal is to select that candidate-pair that yields, as a solution to Equation 1, an optimal value for V—this interpretation is implicit in Bellman's Equation 2.

Consider a very simple example of a bandit problem in which there are three tasks, each with only two states. Then FABP-states may be represented by binary strings of length three; for example, the string "011" signifies a FABP-state in which the first bandit process in is state 0 while the other bandit processes are both in state 1. The transition matrix for a policy defined by taking the same action in all states is 8-by-8, but is quite sparse. For example, the transition matrix associated with the action "activate the

second task" has, for each row, possible nonzero entries only for columns associated with bit-strings that match the row bit-string in all positions except possibly the second, a consequence of the fact that non-activated tasks do not change state. This, along with the fact that rewards received depend only upon the state of the active task, may lead one to suspect the existence of decompositional algorithms of order $O(N|S|^c)$ for computing optimal policies.)

4 The Gittins Index

> "...The term 'Gittins index' now has firm currency in the literature, denoting the concept which first proved so crucial in the solution of the long-standing multi-armed bandit problem and since then has provided a guide for the deeper understanding of all such problems....The multi-armed bandit is a prototype of this class of problems, propounded during the Second World War, and soon recognised as so difficult that it quickly became a classic, and a by-word for intransigence. In fact, John Gittins had solved the problem by the late sixties, although the fact that he had done so was not generally recognised until the early eighties. I can illustrate the mode of progagation of this news, when it began to progagate, by telling of an American friend of mine, a colleague of high repute, who asked an equally well-known colleague 'What would you say if you were told that the multi-armed bandit problem had been solved?' The reply was somewhat in the Johnsonian form: 'Sir, the multi-armed bandit problem is not of such a nature that it *can* be solved.' "
>
> —Peter Whittle[2]

Consider the version of the multi-armed bandit problem described previously in Section 2 in which the decision maker has the added option at each stage k of permanently retiring and receiving retirement reward $\gamma^k M$.

[2]From the forward to [Gittins, 1989]. Elsewhere (in the discussion following [Gittins, 1979]), Whittle recalls that the efforts to solve the multi-armed bandit problem "... so sapped the energies and minds of Allied analysts that the suggestion was made that the problem be dropped over Germany, as the ultimate instrument of intellectual sabotage."

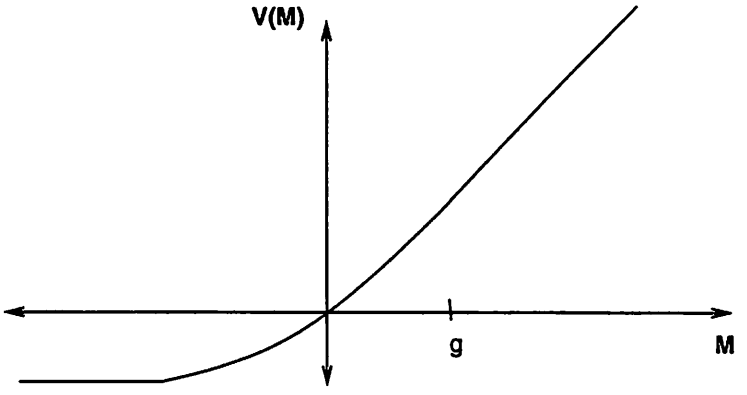


Figure 2: The Gittins index as an indifference threshold (after [Bertsekas, 1987], p. 262).

The rich structure of the bandit problem turns out to imply that there exist functions, $g_i(x_i(k))$, for each task that map bandit process states to numbers, or "indices," such that optimal policies for the FABP have the form:

$$\text{Retire if } M > \max_i\{g_i(x_i(k))\}$$
$$\text{Activate task } j \text{ if } g_j(x_j(k)) = \max_i\{g_i(x_i(k))\} \geq M.$$

Thus one interpretation of $g_i(x_i(k))$ is as an index of profitability for activating task i; it is known as the "Gittins index."

In order to gain further insight into the meaning of the Gittins index and, perhaps, a method for calculating it, consider the bandit problem for a single task i. This is a standard stopping problem.

Let $V_i^*(x_i, M)$ be the optimal value function viewed as a *function of* M for fixed x_i. For large values of M, $V_i^*(x_i, M) = M$, while for sufficiently small M, $V_i^*(x_i, M)$ is some constant value independent of M (*i.e.*, the optimal policy excludes retirement). Between these two extremes, it may be shown that $V_i^*(x_i, M)$ is convex and monotonically non-decreasing, and that there is a minimal value of M such that $V_i^*(x_i, M) = M$ (Figure 2).

In fact, the Gittins index is this minimal value; that is, for all x_i,

$$g_i(x_i) = \min\{M | V_i^*(x_i, M) = M\}.$$

Thus, another interpretation for the index is that it provides an indifference

threshold for each state between retiring and activating the task when in state x_i.

Proof of the fact that policies determined by indices defined in this way are optimal is beyond the scope of this paper (see [Gittins, 1989], [Varaiya *et al*, 1985], or [Bertsekas, 1987] for rigorous proofs). Whittle's proof [Whittle, 1982] that the index rule yields an optimal policy reveals along the way an interesting relationship that exists between the optimal value function of the overall multi-task FABP, $V^*(\underline{x}, M)$, and the optimal value functions of the component bandit processes, $V_i^*(x_i, M)$:

$$\frac{\partial V^*(\underline{x}, M)}{\partial M} = \prod_{i=1}^{N} \frac{\partial V_i^*(x_i, M)}{\partial M}.$$

Another interpretation of the index may be derived [Ross, 1983] by again considering the single-task problem in initial state x_i and retirement reward $M = g_i(x_i)$; *i.e.*, the optimal policy is indifferent between continuing and retiring. It follows that, for any positive random retirement time, τ, (a "stopping time" in the sense of stochastic process theory [3])

$$g_i(x_i) \geq E[\text{discounted return prior to } \tau] + g_i(x_i)E[\gamma^\tau], \tag{3}$$

with equality holding under the optimal continuation policy. Therefore,

$$g_i(x_i) = \max_{\tau > 0} \frac{E[\text{discounted return prior to } \tau]}{1 - E[\gamma^\tau]},$$

or

$$(1-\gamma)g_i(x_i) = \max_{\tau > 0} \frac{E[\text{discounted return prior to } \tau]}{E[\text{discounted time prior to } \tau]}.$$

Thus, to calculate an index, it suffices to find the stopping time, τ, such that the maximum reward per unit time (both discounted) prior to τ is maximal.

Weber provides an intuitive proof [Weber, 1992] for the optimality of the Gittins index rule that is based on the notion of a fair game and an interpretation of the index that is equivalent to the previously-mentioned

[3] An integer-valued positive random variable τ is said to be a *stopping time* for the sequence $\{X(k)\}$ if the event $\{\tau = t\}$ is independent of $X(t+1), X(t+2), \ldots$ for all $t = 1, 2, \ldots$

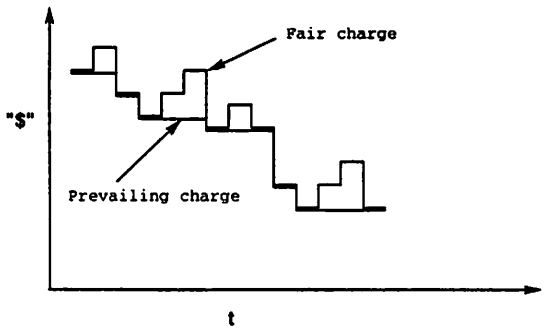


Figure 3: Fair charge and prevailing charge associated with an example bandit process trajectory.

indifference-threshold interpretation. A similar view is presented in [Ishikida & Varaiya, 1994], where the index is interpreted as the winning bid in an auction for the right to use a "pipe," and in [Gittins, 1989], where candidate bandit processes are callibrated against a "standard bandit process." The following discussion follows [Weber, 1992].

Suppose there is only one bandit process and that the decision-maker (gambler) may choose to activate (play) the process or not, but must pay a fixed *prevailing charge* for each play. For bandit i in state x_i, one may define the *fair charge*, $g_i(x_i)$, as the value of prevailing charge for which optimal play of the bandit is a fair game; that is,

$$g_i(x_i) = \sup\left\{g : \sup_{\pi}\left[\sum_{t=0}^{\tau-1}\gamma^t(R_i(x_i(t)) - g)|x_i(0) = x_i\right] \geq 0\right\},$$

where the stopping time τ is defined by the policy π.

As the bandit process state evolves, so too does the fair charge. In the event that the fair charge of the current state dips below the prevailing charge, in which case the gambler would normally stop playing, imagine that the prevailing charge is reset to the fair charge (Figure 3). Then the sequence of prevailing charges for each bandit process is non-increasing with the number of plays, and the gambler experiences continued play of a fair game. For the case of multiple bandit processes, by following a policy of playing the bandit of greatest prevailing charge (or equivalently fair charge), the gambler

interleaves the prevailing charges from component bandit streams into one non-increasing sequence. By the nature of discounting, such a policy maximizes the expected total-discounted charge paid by the gambler. Since the gambler is engaged in playing a fair game, this policy maximizes expected total-discounted reward.

5 Restart-in-state-i problems and the Gittins Index

Restrict attention, for the moment, to the transition- and reward-structure associated with a single task and consider the following "restart-in-i" problem. In each state, j, one has the option of either continuing from state j and accumulating discounted rewards, or else instantaneously "teleporting" to state i and accumulating discounted rewards from there. The problem is to find a policy that optimizes the expected discounted value of each state.

The dynamic programming equation for the optimal value function for this problem may be written: for $j = 1$ to N,

$$V_j^i = \max\{\underbrace{r_j + \gamma \sum_k P_{jk} V_k^i}_{\text{"Continue"}}, \quad \underbrace{r_i + \gamma \sum_k P_{ik} V_k^i}_{\text{"Restart"}}\},$$

where V_j^i signifies the j^{th} component of the optimal value function for the restart-in-state-i problem.

In particular, the i^{th} component satisfies

$$V_i^i = r_i + \gamma \sum_k P_{ik} V_k^i,$$

and V_i^i may also be interpreted as the maximum value in state i for the corresponding embedded single-state semi-Markov decision chain; *i.e.*, V_i^i satisfies

$$V_i^i = \max_{\tau > 0} E\left\{[\text{discounted reward prior to } \tau] + \gamma^\tau V_i^i\right\},$$

where τ is a stopping time for the process, namely, the first period in which one chooses to restart in state i in the restart-in-i problem. Comparing this last equation with Equation 3 in the preceeding section under the optimal

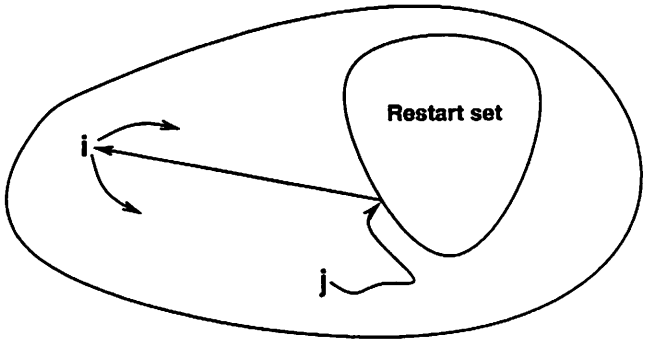


Figure 4: The restart-in-i problem.

continuation policy, one concludes that V_i^i may be identified with the Gittins index, $g(i)$, for state i.

For a given state i, there is a set of states, the "optimal restarting set," for which, once entered, it is optimal to restart in i (see Figure 4). The number of transitions taken to reach this restarting set, starting from state i, is the optimal stopping time associated with the Gittins index.

Katehakis and Veinott [Katehakis & Veinott, 1987] suggest calculating the Gittins indices by solving the corresponding restart-in-i problems via successive approximation:

$$\begin{aligned}
V_1 &\leftarrow \max\{V_i, r_1 + \sum_k P_{1k} V_k\} \\
V_2 &\leftarrow \max\{V_i, r_2 + \sum_k P_{2k} V_k\} \\
&\vdots \\
V_i &\leftarrow r_i + \sum_k P_{ik} V_k \\
&\vdots \\
V_N &\leftarrow \max\{V_i, r_N + \sum_k P_{Nk} V_k\}
\end{aligned}$$

For each state i, there corresponds a restart-in-i subproblem that can be solved in this way, yielding as its i^{th} component the Gittins index for state i.

6 On-line estimation of Gittins indices via Q-learning

The multi-armed bandit problem was stated in Section 2, and Section 4 presented the Gittins index approach for constructing optimal policies, an approach that reduces the bandit problem to a sequence of low-dimensional stopping problems; Section 5 asserted that the Gittins index for a given state may be characterized as a component of the optimal value function associated with a stopping problem, a simple MDP.

Reinforcement learning methods, such as Q-learning, are adaptive, model-free algorithms that can be applied online for computing optimal policies for MDP's. Q-learning [Watkins, 1989] was originally advanced as a sample-based, Monte-Carlo extension of successive approximation for solving Bellman's equation; alternative motivation and justification for the algorithm, as well as rigorous proofs of convergence, appeal to results from the theory of stochastic approximation, see [Jaakkola *et al*, 1994] and [Tsitsiklis, 1994]—the Appendix to this paper provides an informal summary and rationale.

It follows that, in principle, Q-learning can be applied to calculate Gittins indices and hence provides a model-free means for learning to solve bandit problems online.

To be a bit more specific, for a given task and given restart-in-i subproblem there are two actions available for each state: "continue" and "restart." Associated with each of these actions is a transition matrix and expected reward vector corresponding to a policy in which only the continue action or only the restart action is applied (all rows of the restart transition matrix and entries of restart expected reward are identical to the i^{th} row of the continue transition matrix and i^{th} entry of the reward vector, respectively). As was discussed in Section 3.2, the problem of finding an optimal policy for the restart-in-i MDP is equivalent to the problem of constructing a transition-matrix/expected-reward-vector pair, whose rows and entries are drawn from rows and entries of continue- and restart- matrices and reward vectors, that has maximal value.

In principle, the theory of reinforcement learning implies that Q-learning will converge to the correct optimal values associated with the various restart-in-i MDP's. However, in practical terms, it is reasonable to question the meaning of the "restart" action in, for example, the context of stochastic

scheduling. One cannot, simply, reset a given task to a desired state by an omnipotent act of will. What one desires is that Q-learning "backups" (componentwise computational contractions — see the Appendix) be performed for states that arise naturally along sample paths of the FABP process.

Consider, then, one step of the FABP process in which a given task is activated and changes state from state i to state j generating reward r. This simple transition yields data relevent to the value of taking the continue action when in state i. Note that *the data are relevent to* all *restart problems for the given task.* Observe also that the transition supplies information about taking the restart action for the restart-in-i subproblem *for all states* in the given task.

In summary, observing a state-transition from i to j and reward r for a given active task with n states allows $2n$ Q-learning backups to be performed; that is, for $k = 1$ to n, backup:

Q(state=i, action=Continue, restart problem = k) —"Continue data"
Q(state=k, action=Restart, restart problem = i) —"Restart data".

It remains to define task activations in a way that achieves the requisite sampling of states and actions as discussed in the Appendix. There are many reasonable ways of doing this; a generalized Boltzman-distribution-based action-selection method is proposed here.

Suppose that the multi-task bandit process is in some given state $\underline{x} = (x_1, x_2, ..., x_N)$. The current estimate of the Gittins index for task i in state x_i is given by

Q(state=x_i, action=Continue, restart problem=x_i, task=i).

Define action-selection via the following Boltzman distribution: for $i = 1$ to N,

$$Pr\{\text{activate task } i\} = \frac{e^{Q(x_i,C,x_i,i)/T}}{\sum_{i=1}^{N} e^{Q(x_i,C,x_i,i)/T}}$$

—where T is the Boltzman temperature described in the Appendix.

In summary, at each stage:

- Select a task to activate via the Boltzman distribution.

- Observe the state-transition $i \to j$ and immediate reward r elicited by activating the task.
- Perform $2n$ backups, where n is the number of states for the activated task: for $k = 1$ to n,

$$\begin{aligned}
&\text{Q(state=}i\text{, action=Continue, restart problem=}k\text{, task)} = \\
&\qquad (1-\alpha)\text{Q(state=}i\text{, action=Continue, restart problem=}k\text{, task)} \\
&\qquad +\alpha\left[r+\gamma \max_{a\in\{C,R\}} \text{Q(state=}j\text{, action=a, restart problem=}k\text{, task)}\right] \\
&\text{Q(state=}k\text{, action=Restart, restart problem=}i\text{, task)} = \\
&\qquad (1-\alpha)\text{Q(state=}k\text{, action=Restart, restart problem=}i\text{, task)} \\
&\qquad +\alpha\left[r+\gamma \max_{a\in\{C,R\}} \text{Q(state=}j\text{, action=a, restart problem=}i\text{, task)}\right],
\end{aligned}$$

where "C" and "R" respectively denote the admissible actions, continue and restart.

If each of the N alternative processes or tasks has n possible states, then $2Nn^2$ Q-values must be calculated and stored. Note that this is a substantial reduction from the Nn^N values required by an approach based upon the straightforward MDP formulation.

Moreover, each state-transition gives rise to $2n$ backups, and this effective parallelism may be viewed as further reducing the computational complexity. That is, to calculate all the Gittins indices, the algorithm solves Nn MDP's (number of tasks $\times$ number of restart-problems per task), each of size n. But for each task the associated n restart-problems are solved in parallel, and are rather simple MDP's in that there are only two admissible actions per state.

7 Examples

To confirm that this algorithm works, first consider the simple bandit problem shown in Figure 5. This problem has two tasks, each with two states. Transition probabilities/rewards label arcs, and the discount factor is chosen to be $\gamma = .7$.

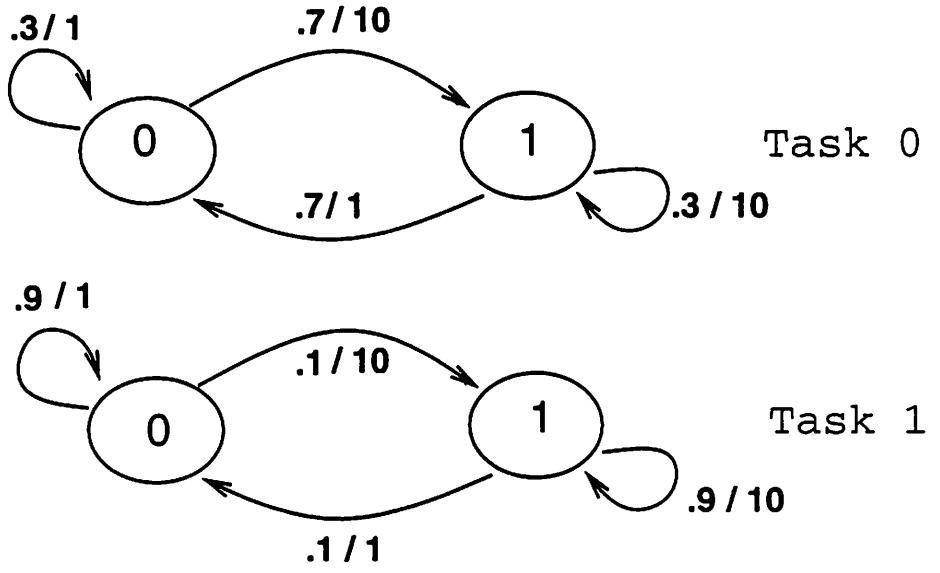


Figure 5: A simple bandit problem.

The optimal policy may be calculated by solving a four-state MDP, as discussed in Section 3.2, or by applying the model-based, successive approximation scheme of Katehakis and Veinott offline. The optimal policy is to activate task 1 if it is in state 1, but otherwise to activate task 0.

Figure 6 plots the convergence of the Gittins indices to their true values using the reinforcement learning algorithm proposed in Section 6.

The optimal policy is to activate task 0 once task 1 leaves state 1. Consequently, as the Boltzman temperature is lowered, an increasing number of transitions are made under the greedy policy with respect to the index estimates; that is, an increasing proportion of transition samples are drawn from task 0 activations.

Miscellaneous parameters that govern the rate of Boltzman temperature and step-size reduction have not been optimized; the purpose of this example has been simply to demonstrate that the on-line algorithm works.

A more meaningful example bandit problem is that of (static) stochastic scheduling. Consider the scenario in which, at the begining of each "trial," one is presented with a fixed number of tasks to be completed, where each

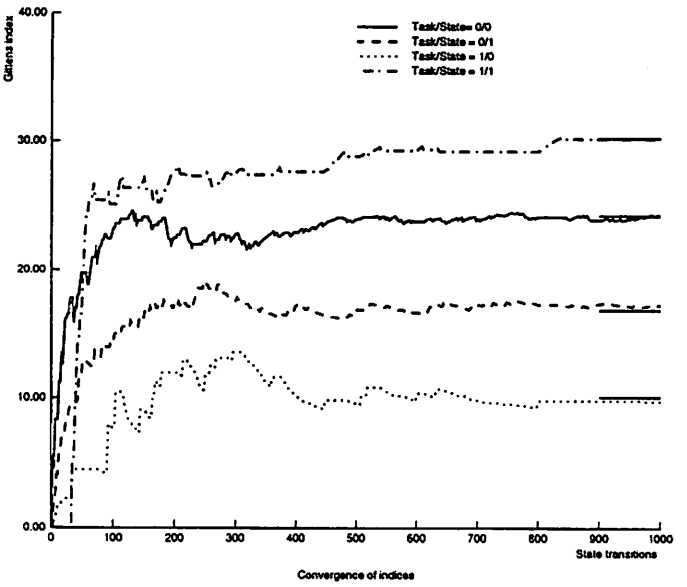


Figure 6: Convergence of Gittens Indices for a simple example.

task, i, has a service time determined by a respective distribution function, F_i.

For example, consider the problem of task scheduling where each task i has a geometric service time:

$$Pr\{\tau_i = s\} = \rho_i(1-\rho_i)^{s-1}.$$

See Figure 7a.

Each task i thus has a constant hazard rate, ρ_i, and it is known that, in order to minimize either mean flow time [4] or mean waiting time, an optimal policy is to activate the tasks in decreasing order of this parameter.

It may be unreasonable to presume that the service-time distributions are known *a priori*. In this case, the reinforcement-learning algorithm of Section 6 can be applied, online, to calculate the respective Gittins indices directly, without building an explicit task model.

In a simple experiment, ten ρ_i's where drawn uniformly from the unit interval, and the discount rate was set to $\gamma = .9$. The reinforcement-learning

[4] *Mean (weighted) flowtime* is defined as the (weighted) sum of task finishing times, divided by the number of tasks.

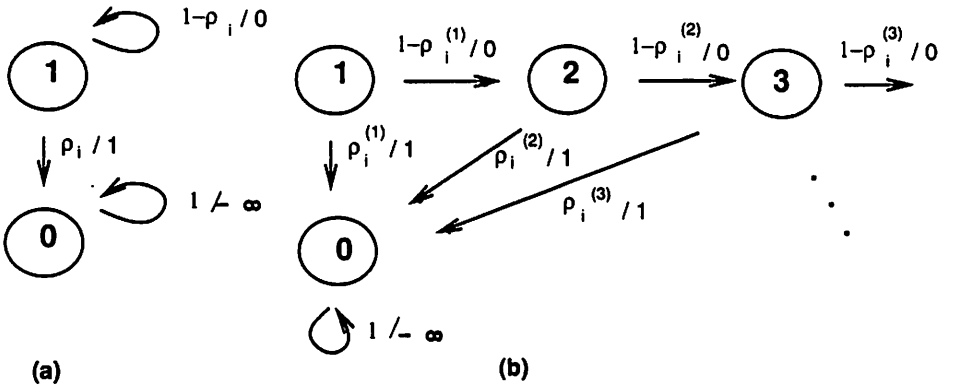


Figure 7: Task models with constant (a) and non-constant (b) hazard rates.

algorithm was applied, online, in a trial-based fashion. (A trial consists of activating tasks in some manner until all tasks are complete.) Index estimates converged to values whose magnitudes are consistent with the optimal highest-hazard-rate policy.

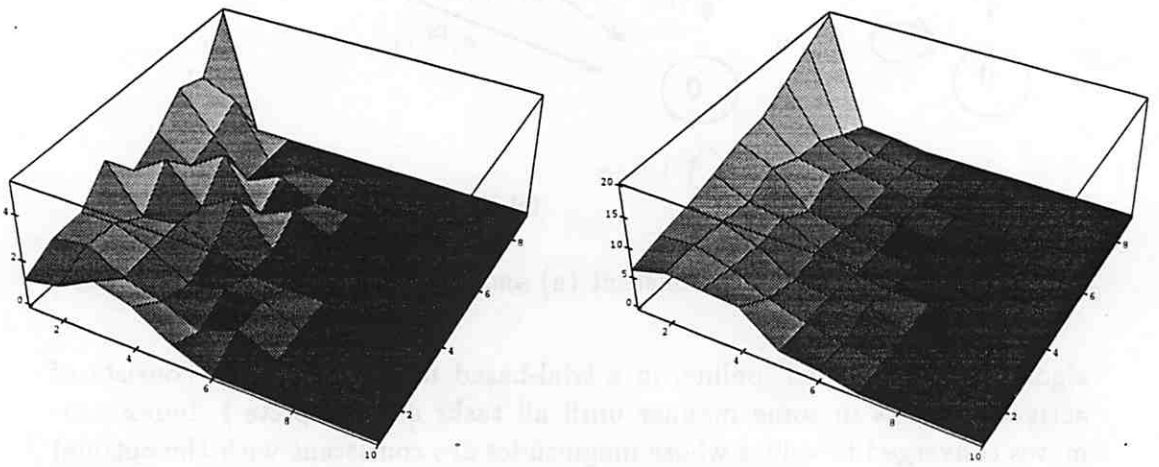
The constant hazard rate case was considered in the previous example because the optimal policy is known and provides a simple check. Non-constant hazard rate cases can be handled by defining the task models as suitable Markov reward chains, see Figure 7b.

For example, consider a task model for a case in which each task has increasing hazard rate:

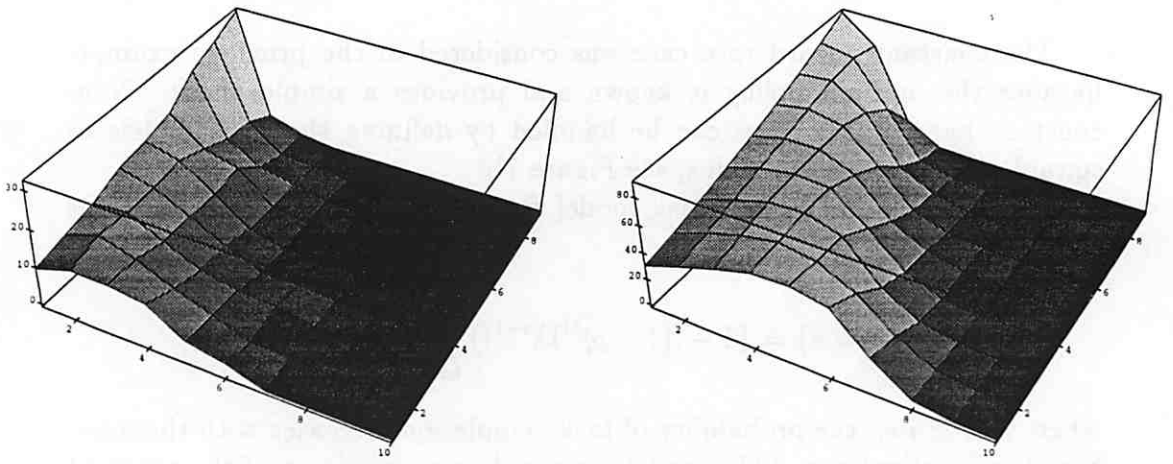
$$Pr\{\tau_i = s\} = \{1 - [(1 - \rho_i^{(1)})\lambda^{s-1}]\} \prod_{k=1}^{s-1} (1 - \rho_i^{(1)})\lambda^{k-1},$$

where $\lambda < 1$; *i.e.*, the probability of task completion increases with the number of task activations. (The model is a simple generalization of the constant hazard-rate case, where now the probability of non-completion decreases in a simple exponential fashion with respect to the number of task activations.)

An experiment was performed in which nine tasks were modeled via $\rho_i^{(1)} = .1i$, $i = 1, 2, ..., 9$, $\lambda = .8$, and the discount factor, γ, was set to 0.99.



10 trials 50 trials



100 trials 1000 trials

Figure 8: Gittins index surface plotted as function of state (x-axis) and task (y-axis).

Again, the reinforcement learning algorithm was applied online in a trial-based way, and the results are presented in Figure 8, which plots the Gittins-index "surface" estimate (vertical axis) versus task (axis into the page, ranging from task 1 to task 9) and task-state (axis left-to-right, ranging from state 1 to state 10) at various stages of the learning process.

It may be seen that the index values gradually "unroll" from the task axis (Q-values were initialized to zero). Low-numbered states are sampled more frequently, and consequently their index values converge more rapidly than do those of rarely-encountered task-states such as, for example, state 3 of task 9.

Again, it is known through analytical means that for this problem the optimal schedule is to sequence the tasks non-preemptively, highest hazard-rate first.

The plots of Figure 8 appear to be converging to indices that would give rise to such a policy—index estimates for rarely-encountered task-states are slowly rising to their true values. For commonly-encountered bandit states, the Gittins surface estimate yields an optimal scheduling policy relatively early in the learning process.

Note that if one were to pursue the straightforward MDP approach of Section 3.2, it would entail a state-set size on the order of fifty million ($\approx 9 \times 9 \times 8 \times 8 \times 7 \times 7 \times 6 \times 6 \times 5$) and transition matrices of corresponding dimension—this is assuming that one knows beforehand the effective range of states for each task.

It is perhaps important to stress that, in the scheduling literature, it is always assumed that the service-time distributions are known; one contribution of this paper is that the reinforcement learning algorithm makes no such assumption.

The problem of stochastic scheduling for the cases of constant or monotone hazard-rates is analytically tractable, and the resulting policies are usually somewhat intuitive and can be stated simply. For arbitrary, non-monotone hazard-rates, things are less well-understood, but there is nothing in the reinforcement learning approach that would preclude its application to these cases.

For further applications of the bandit formulation to job scheduling, resource allocation, sequential random sampling, random search, *etc.*, refer to the papers by Glazebrook, the book by Gittins [Gittins, 1989], and references listed therein.

8 Conclusion

This paper has traced the following chain of reasoning:

- A family of alternative bandit processes is a Markov decision process (MDP) possessing a special decompositional (Cartesian-product) structure.
- Optimal policies for these processes can be constructed efficiently by calculating Gittins indices.
- The Gittins index for a given task state i is also the i^{th} component of the "restart-in-i" problem.
- The restart-in-i process is a standard MDP.
- Optimal policies for MDP's can be computed online in a model-free way using Q-learning.
- Therefore, Q-learning can be applied online, without using a process model, to compute solutions to bandit problems. (The implementational details of a practical algorithm were presented in Section 6.)

For each alternative n-state process, the resulting algorithm computes, in parallel, the desired Gittins indices by solving n two-action MDP's, each of size n.

A proof of convergence follows from existing convergence proofs of Q-learning for conventional MDP's [Jaakkola *et al*, 1994], [Tsitsiklis, 1994].

One advantage of reinforcement learning methods that has not been mentioned thus far is that, as Monte-Carlo methods, they may inherit some computational advantage over conventional (model-based) methods, particularly for very large problems. This aspect is discussed in [Barto & Duff, 1994]. If one has a model of the process, or processes, real-time dynamic programming [Barto *et al*, 1991] can be applied, in which full model-based backups are performed for states encountered along sample-paths. Indirect methods, such as adaptive real-time dynamic programming, adaptively construct a model

for the controlled process and base control policies and value-function update computations on the latest model (see [Gullapalli & Barto, 1994] for a convergence proof).

There are a number of generalizations of the basic bandit formulation that are of extreme practical interest for scheduling. For example, Glazebrook and Gittins [Glazebrook & Gittins, 1981] have examined the issue of the existence of index theorems for bandit problems with precedence contraints (their focus is on such constraints that have a tree structure). Whittle [Whittle, 1981] has studied bandit problems in which new tasks arrive (index results are preserved when the arrival process is Poisson/Bernoulli). The case of context-switching costs has been addressed in [Glazebrook, 1980]. When there is more than one server or processor available—thus enabling more than one process to be active at a time—in general, quite strong additional conditions are required for an index theorem to hold. (The special case of the general scheduling problem with only two processors, deterministic service times, and no precedence constraints is an alternative standard specification of the knapsack problem.)

It is interesting to consider the possible application of the bandit problem formulation to certain problems in robotics. For example in [Grupen *et al*, forthcoming] a "control basis" approach is proposed for distributed control of manipulation tasks in which control actions are derived from a sequence of concurrent activations of a subset of a feedback control basis (a set of declarative feedback laws) bound to specific system degrees of freedom. A generalization of standard bandit processes, called "superprocesses" ([Whittle, 1980],[Varaiya *et al*, 1985]), may be relevant to the problem of learning compositional policies online.

From a cognitive science perspective, it is tempting to identify the notion of task activation with the concept of "attention." In performing a motor-control task, for example, humans adaptively allocate sensory resources. [Gelfand *et al*, 1994], in the context of robotic sensor integration for motor control, notes that the act of invoking a vision system incurs a high computional cost. It follows that a vision system may be employed initially to help learn calibrations for unmodeled interactions, but as learning takes place, the vision system's role in a given task should be transfered to other internal (proprioceptive) sensors.

It has been said that "Robotics is the intelligent connection of perception to action" [Brady, 1985], but for many natural or synthetic intelligent systems it may, in fact, be impossible to distinguish or separate these two activities—a sensing task may be viewed as an action that changes the "state" of a world model. Agents are active perceivers.

Thus, in the context of active, sensor-based information gathering, a sensor-planning process must decide what and how to observe, and how much effort to allocate to sensing a given task.

The bandit problem formulation, with its rigorous theoretical foundation and attendant analytical machinary, presents a compelling model for this information-gathering process. Task-directed sensor plans could be derived from the optimal strategies computed for suitable bandit models. In a "sensor-fusion" setting, this paper's on-line, model-free reinforcement learning algorithm constitutes one approach to the problem of learning how to perceive.

The bandit formalism may also serve as a useful tool in analyzing/solving certain general classes of planning problems. A key feature of bandit problems is the (statistical) independence of their component processes. Operator decomposability [Korf, 1987], which is sufficient for the existence of nontrivial macro tables, is similar in nature to bandit process independence. For planning problems in which subgoals are independent, serializable, or perhaps block-serializable, one may consider identifying the evolution of a given state-component in problem space under the action of an operator or macro-operator toward a given subgoal with a bandit problem component process. The motivation for adopting the bandit problem formalism is that, as in the specific context of sensor planning mentioned above, plans could be derived from optimal policies computed for the bandit model. Admittedly, in many contexts the basic bandit formulation may be too restrictive; in the analysis of planning problems with serializable subgoals, for example, it may be necessary to generalize the notion of bandit process independence to "serializable" process independence. The bandit problem formalism may also prove useful in the exploration of independent subgoals that are subject to a common resource restriction, or in planning with respect to an abstraction hierarchy and its associated hierarchy of macro networks.

The reinforcement learning algorithm presented in Section 6 has undergone only preliminary empirical testing; its convergence could be accellerated through the utilization of function approximators for representing Q-values or thoughtful selection of learning rate parameters, which raises an interesting issue:

The examples of Section 7 considered problems of stochastic scheduling as specific instances of the general bandit problem formulation. But general bandit problems themselves are archetypes of "optimal learning" problems, in which the goal is to collect information and use it to inform behavior so as to yield the largest expected reward from actions taken throughout the *entire* duration of the learning process. (The reader is urged to recall the slot machine interpretation of the multi-armed bandit problem stated at the end of Section 2.) This paper has presented a reinforcement learning-based algorithm for solving bandit problems and thus, in a sense, it might well have been entitled, "Learning how to Learn Optimally." But the reinforcement learning algorithm is itself surely not optimal; its Boltzman-distribution-based scheme of action selection is practical and provisional, neither inspired nor informed by a bandit-problem mode of analysis.

One could envision, then, the problem of optimally learning how to learn optimally. (But could one learn how to do this, and do so optimally?...) This regress, as stated, is not entirely meaningful, for as Watkins has observed (citing [McNamara & Houston, 1985]): "Learning is optimal only with respect to some prior assumptions concerning the ... probability distributions over environments the animal [decision-maker] may encounter."

Appendix: An informal summary of TD(0) and Q-learning

TD(0)

Recall from Section 3.1 the system of linear equations that determines a set of values for a Markov chain with rewards: for $i = 1$ to N,

$$V(i) = R(i) + \gamma \sum_j P_{ij} V(j). \tag{4}$$

Since the right-hand side, viewed as an operator acting on $V(\cdot)$, is a

contraction, successive approximation is one viable computational scheme for finding the solution: for $i = 1$ to N,

$$V^{(k+1)}(i) = R(i) + \gamma \sum_j P_{ij} V^{(k)}(j).$$

TD(0) [Sutton, 1988] is a stochastic approximation method for finding solutions to Equation 4; it proceeds by taking very small steps in the directions suggested by the "instantaneously-" sampled version of successive approximation; *i.e.*, having observed a transition for state i to j with reward r, one's instantaneous view of the right-hand side of the successive approximation recursion is $r + \gamma V^{(k)}(j)$. TD(0) takes a small step in the direction $r + \gamma V^{(k)}(j) - V(i)$:

$$\begin{aligned} V^{(k+1)}(i) &= V^{(k)}(i) + \alpha_k [r + \gamma V^{(k)}(j) - V^{(k)}(i)] \\ &= (1 - \alpha_k) V^{(k)}(i) + \alpha_k [r + \gamma V^{(k)}(j)]. \end{aligned}$$

The update is severely "underrelaxed" (α_k is very small) so as to, over the long run, average out the sampling noise. [5] The fact that the expected value of the instantaneous sample is equal to successive approximation's right-hand side implies, by the theory of stochastic approximation, that if step-sizes α_k diminish at the appropriate rate and other sensible assumptions hold, then the sequence of value-function estimates generated by the TD(0) proceedure will converge to the true solution with probability one [Jaakkola *et al*, 1994].

Q-learning

Recall from Section 3 the system of *nonlinear* equations that determines a set of *optimal* values for a Markov decision process; *i.e.*, Bellman's equation: for $i = 1$ to N,

$$V(i) = \max_{a \in \mathcal{A}_i} \left[R(i,a) + \gamma \sum_{j=1}^{N} P_{ij}(a) V(j) \right]. \tag{5}$$

One may show that the right-hand side of Equation 5, viewed as an operator, is also a contraction, and so a successive approximation method will

[5] This view of TD(0) seems to have been first suggested by Etienne Barnard in [Barnard, 1992].

converge to the solution. Blindly attempting to apply a stochastic approximation approach using this successive approximation recursion, however, is complicated by the presence of the "max" operation. What is the meaning of "instantaneous sample of $\max_a \left\{R(i,a) + \gamma \sum_j P_{ij}(a) V^{(k)}(j)\right\}$?" The issue can be sidestepped by shifting the focus from the estimation of optimal state values to the estimation of optimal state/action values — the so-called optimal "Q-values" [Watkins, 1989].

The Q-function of a state/action pair, with regard to a value-function estimate, is the expected infinite-horizon return when, starting in the given state, the given action is applied, and actions thereafter are prescribed by a policy having the given value-function; that is, $Q_V(i,a) \equiv R(i,a) + \gamma \sum_j P_{ij}(a) V(j)$, the quantity appearing within the scope of the "max" operator in Bellman's equation. Informally, the Q-function with regard to a policy is the value of the policy if the policy's initial action is perturbed.

In terms of Q-functions, Bellman's equation is just $\max_{a'} Q^*(j,a') = V^*(j)$, where the star signifies *optimal* value functions. Therefore, by the definition of Q,

$$Q^*(i,a) = R(i,a) + \gamma \sum_j P_{ij}(a) \max_{a'} Q^*(j,a'),$$

which suggests the successive approximation recursion:

$$Q^{(k+1)}(i,a) = R(i,a) + \gamma \sum_j P_{ij}(a) \max_{a'} Q^{(k)}(j,a').$$

The meaning of "instantaneous sample of the right-hand side" for this equation is less problematic than that for the original form of Bellman's equation. The introduction of Q allows one, in effect, to interchange the order of expectation and maximization. Observing a transition from state i to j and reward r upon executing action a admits an instananeous view of the right-hand side as $r + \gamma \max_{a'} Q^{(k)}(j,a')$, which suggests a stochastic approximation recursion (referred to as a "backup") of the form

$$\begin{aligned} Q^{(k+1)}(i,a) &= Q^{(k)}(i,a) + \alpha_k \left[r + \gamma \max_{a'} Q^{(k)}(j,a') - Q^{(k)}(i,a) \right] \\ &= (1-\alpha_k) Q^{(k)}(i,a) + \alpha_k \left[r + \gamma \max_{a'} Q^{(k)}(j,a') \right]. \end{aligned} \tag{6}$$

Theory [Bertsekas & Tsitsiklis, 1989] implies that, if each action is tried in each state an infinite number of times, then $Q^{(k)}$ converges to Q^*. In

practice, the evolving current estimate of Q^* is often used to guide the selection of actions, resulting in a gradual focusing of backup operations to states encountered along sample paths generated through the application of actions comprising optimal policies ([Barto *et al*, 1991] includes a discussion of the ensuing computational efficiency). The goal of action-selection is to strike a balance between refining estimates for actions currently thought to be optimal for states that lie on optimal trajectories and exploring regions of the state-action space for which there is a high degree of value-function uncertainty.

A generic Q-learning computational cycle proceeds as follows (assuming that the system is currently in state i):

- Choose an action, a, via some designated action-selection procedure.
- Observe the state transition, $i \rightarrow j$, under action a, generating immediate reward r.
- Perform the backup operation (Equation 6) to update the Q-value associated with state i / action a.

A number of practical action-selection mechanisims have been proposed to accomplish the desired sampling of state-action values. A widely-used method, proposed by Watkins [Watkins, 1989], suggests choosing an action, a, via the Boltzman distribution:

$$Pr\{a = a_j\} = \frac{e^{Q(i,a_j)/T}}{\sum_j e^{Q(i,a_j)/T}}$$

—where the "Boltzman temperature," T, is initialized to a relatively high value, resulting in a uniform distribution for prospective actions. As computation proceeds, temperature is gradually lowered, in effect raising the probability of selecting actions with higher Q-values; in the limit, the action that is "greedy" with respect to Q is selected. The details of one way (the method used in later examples here) of defining the exact manner in which the temperature is lowered is given in Appendix D of [Barto *et al*, 1991]. The step size α_k is also decreased over time in accordance with the requirements for stochastic approximation convergence. In the examples that appear in Section 7, this was achieved by implementing the "search-then-converge" schedule recommended in [Darken & Moody, 1991].

In summary, reinforcement learning methods, such as Q-learning, are adaptive, model-free algorithms that can be applied online for computing optimal policies for MDP's. Q-learning was originally advanced as a sample-based, Monte-Carlo extension of successive approximation for solving Bellman's equation; alternative motivation and justification for the algorithm, as well as rigorous proofs of convergence, appeal to results from the theory of stochastic approximation, see [Jaakkola *et al*, 1994] and [Tsitsiklis, 1994].

Acknowledgements

Thanks to Professor Andrew Barto, and to the members of the Adaptive Networks Laboratory. This work was supported, in part, by the National Science Foundation under grant ECS-9214866 to Professor Barto.

References

A. Barto, R. Sutton, & C. Watkins. (1990) "Learning and Sequential Decision Making" in M. Gabriel & J. Moore, eds. *Learning and Computational Neuroscience: Foundations of Adaptive Networks,* MIT Press, pp. 539-602

A. Barto, S. Bradtke, & S. Singh. (1991) "Real-Time Learning and Control Using Asynchronous Dynamic Programming." Computer Science Department, University of Massachusetts, Tech. Rept. 91-57.

A. Barto & M. Duff. (1994) "Monte-Carlo Matrix Inversion and Reinforcement Learning" in *Neural Information Processing Systems — 6,* 687-694.

R. Bellman. (1956) "A Problem in the Sequential Design of Experiments," *Sankhya,* 16: 221-229.

D. Bertsekas. (1987) *Dynamic Programming: Deterministic and Stochastic Models,* Prenice-Hall.

D. Bertsekas & J. Tsitsiklis. (1989) *Parallel and Distributed Computation: Numerical Methods,* Prentice-Hall.

E. Barnard. (1992) "Temporal-Difference Methods and Markov Models," *IEEE Trans. Syst. Man. & Cybern.*, 22: 357-365.

M. Brady. (1985) "Artificial Intelligence and Robotics," *Artificial Intelligence*, 26: 79-121.

R. Crites. (1995) "Multiagent Reinforcement Learning Applied to Elevator Control. /em In Preparation.

C. Darken, J. Chang, & J. Moody. (1992) "Learning Rate Schedules for Faster Stochastic Gradient Search," *Neural Networks for Signal Processing 2.*

L.E. Dubins & Savage. (1976) *Inequalities for Stochastic Processes: How to Gamble if you Must*, Dover.

J. Gelfand, J. Shi, D. Handelms, & S. Lane. (1994) "Learning Robotic Sensor Integration Through Practice," *Proc. 8th Yale Workshop on Adaptive & Learning Systems*, pp. 27-32.

J.C. Gittins. (1979) "Bandit Processes and Dynamic Allocation Indices," *J.Royal Stat. Soc. Ser. B* 41: 148-177.

J.C. Gittins. (1989) *Multi-armed Bandit Allocation Indices*, Wiley.

J.C. Gittins & D.M. Jones. (1974) "A Dynamic Allocation Index for the Sequential Design of Experiments," in *Progress in Statistics,* J.Gani *et al*, eds., pp.241-266.

K.D. Glazebrook. (1979) "Scheduling Tasks with Exponential Service Times on Parallel Processors" *J.Appl.Prob* 16:685-689.

K.D. Glazebrook. (1980) "On Stochastic Scheduling with Precedence Relations and Switching Costs," *J.Appl.Prob* 17: 1016-1024.

K.D. Glazebrook. (1983) "Optimal Strategies for Families of Alternative Bandit Processes," *IEEE-TAC* 28: 858-861.

K.D. Glazebrook & J.C. Gittins. (1981) "On single-machine scheduling with precedence relations and linear or discounted costs," *Oper. Res.* 29:289-300.

R. Grupen, M. Huber, J. Coelho, & K. Souccar. "A Basis for Distributed Control of Manipulation Tasks," *Forthcoming.*

V. Gullapalli, & A. Barto. (1994) "Convergence of Indirect Adaptive Asynchronous Value Iteration Algorithms," *Neural Information Processing Systems -6*, 695-702.

T Ishikida & P. Varaiya. (1994) "Multi-Armed Bandit Problem Revisited," *J. Opt. Thry. & Applic.*, 83: 113-154.

T. Jaakkola, M. Jordan, & S. Singh. (1994). "Convergence of Stochastic Iterative Dynamic Programming Algorithms," *Neural Information Processing Systems -6*, 703-710.

M.H. Katehakis and A.F. Veinott. (1987) "The Multi-armed Bandit Problem: Decomposition and Computation." *Math. OR.* 12: 262-268.

R.E. Korf. (1985) *Learning to Solve Problems by Searching for Macro-Operators*, Pitman.

A. Mandelbaum. (1986) "Discrete Multi-armed Bandits and Multiparameter Processes," *Prob. Thry. & Related Fields*, 71: 129-147.

J. McNamara & A. Houston. (1985) "Optimal Foraging and Learning." *Journal of Theoretical Biology*, 117: 231-249.

H. Robbins. (1952) "Some Aspects of the Sequential Design of Experiments," *Bull. Amer. Math. Soc.*, 58: 527-535.

S. Ross. (1983) *Introduction to Stochastic Dynamic Programming*, Academic Press.

R. Sutton. (1988) "Learning to Predict by the Method of Temporal Differences," *Machine Learning* 3:9-44.

G. Tesauro. (1992) "Practical Issues in Temporal Difference Learning," *Machine Learning* 8:257-277.

W.R. Thompson. (1933) "On the Likeliehood that one unknown probability exceeds another in view of the evidence of two samples" *Biometrika* 25: 275-294

J. Tsitsiklis. (1986) "A Lemma on the Multiarmed Bandit Problem" *IEEE-TAC* 31: 576-577.

J. Tsitsiklis. "Asynchronous Stochastic Approximation and Q-learning," *Machine Learning* 16:185-202.

P. Varaiya, J. Walrand, & C. Buyukkoc. (1985) "Extensions of the Multi-armed Bandit Problem: The Discounted Case" *IEEE-TAC* 30: 426-439.

J. Walrand. (1988) *An Introduction to Queueing Networks*, Prentice Hall.

C. Watkins. (1989) *Learning from Delayed Rewards.* PhD Thesis Cambridge University.

R. Weber. (1982) "Scheduling Jobs with Stochastic Processing Requirements on Parallel Machines to Minimize Makespan or Flowtime," *J.Appl.Prob.* 19 167-182.

R. Weber. (1992) "On the Gittens Index for Multi-armed Bandits," *Annals of Applied Probability*, 1024-1033.

P. Whittle. (1981) "Arm-acquiring bandits" *Ann. Prob.* 9: 284-292.

P. Whittle. (1982) *Optimization over Time: Dynamic programming and Stochastic Control, Vol. 1*, Wiley.