Q-Learning for Bandit Problems

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

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 process 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 process-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 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 prob-

lems 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.

The main contribution of this paper appears in Section 5, 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 5 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 prespecified transition matrix. This is the case considered henceforth.

The goal is to choose a sequence of actions $\{a_k\}$ to minimize the expected value of the discounted infinite horizon expected 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. [1] 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" (Figure 1).

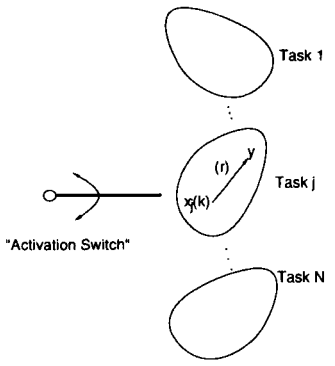


Figure 1: Bandit problem schematic.

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).

In a highly-influential paper, Robbins [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.

The term "multi-armed bandit" refers 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 ban-

[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 3 (note that for M sufficiently small, the retirement option can be excluded).

dit 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.

One final observation: Consider a bandit problem with N component processes, or "tasks," each with state space S. The overall state of the bandit problem process is then an element of S^N, and the action of activating a given task generates an immediate reward and causes the state to change in a Markovian fashion. Hence, ignoring the special structural constraints satisfied by the bandit problem process, it is simply a standard Markov decision process (MDP) with a potentially rather large state set. Transition matrices are $|S|^N$-by-$|S|^N$, and standard methods for computing optimal policies have complexity of order roughly $\boldsymbol{O(|S|^{cN})}$, 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 decompositionally-defined optimal policies whose determination requires work on the order of only $\boldsymbol{O(N|S|^c)}$. 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].

3 THE GITTINS INDEX

Consider a version of the multi-armed bandit problem in which the decision maker has the added option at each stage k of permanently retiring and receiving retirement reward $\gamma^k M$. 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 task states to numbers, or "indices," such that optimal policies have the form:

Retire if $M > \max_i\{g_i(x_i(k))\}$

Activate task j 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

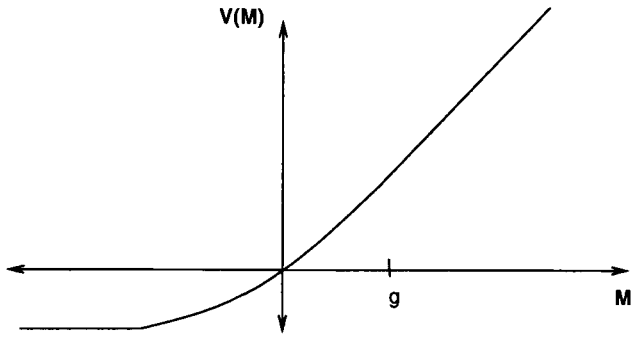


Figure 2: The Gittins index as an indifference threshold (after [Bertsekas, 1987], p. 262).

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.

Rigorous 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 considering the *single*-task problem in intial 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 [2])

$$g_i(x_i) \geq E[\text{discounted return prior to } \tau] + g_i(x_i)E[\gamma^\tau], \tag{1}$$

with equality holding under the optimal continuation policy. Therefore,

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

[2] 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 $t = 1, 2, \ldots$

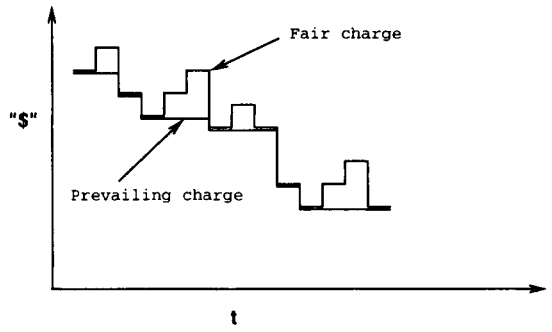


Figure 3: Fair charge and prevailing charge associated with an example bandit process trajectory.

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 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,

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

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.

4 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 with n states 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,

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

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 $V_i^i = r_i + \gamma\sum_k P_{ik}V_k^i$. V_i^i may also be interpreted as the maximum value in state i for the corresponding embedded single-state semi-Markov decision chain; that is, 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 (see Figure 2). Comparing this last equation with Equation 1 in the previous section under an optimal 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

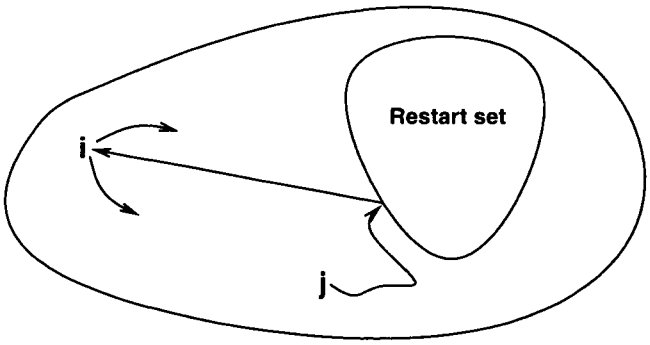


Figure 4: The restart-in-i problem.

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.

5 ON-LINE ESTIMATION OF GITTINS INDICES VIA Q-LEARNING

The multi-armed bandit problem was stated in Section 2, and Section 3 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 4 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].

Thus, 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.

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 updates, or "backups," be performed for states that arise naturally along sample paths of the bandit problem process.

Consider, then, one step 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 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. It follows that 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 an adequate sampling of states and actions. There are many reasonable ways of doing this; a 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."

In summary, at each stage:

- Select a task to activate via the Boltzman distribution.

- Observe the state-transition $i \rightarrow j$ 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,

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

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

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 × 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.

6 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 $\gamma = .7$. The optimal policy may be calculated by solving a four-state MDP, as discussed in Section 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

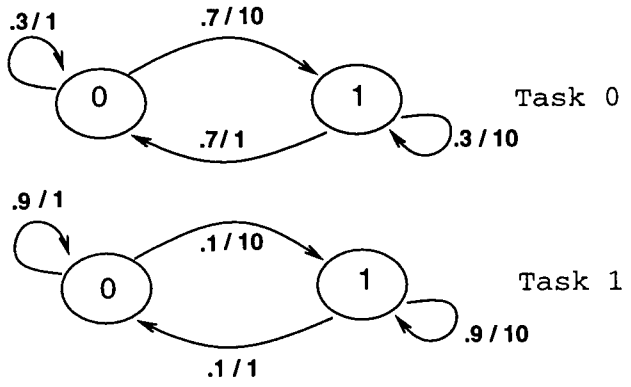


Figure 5: A simple bandit problem.

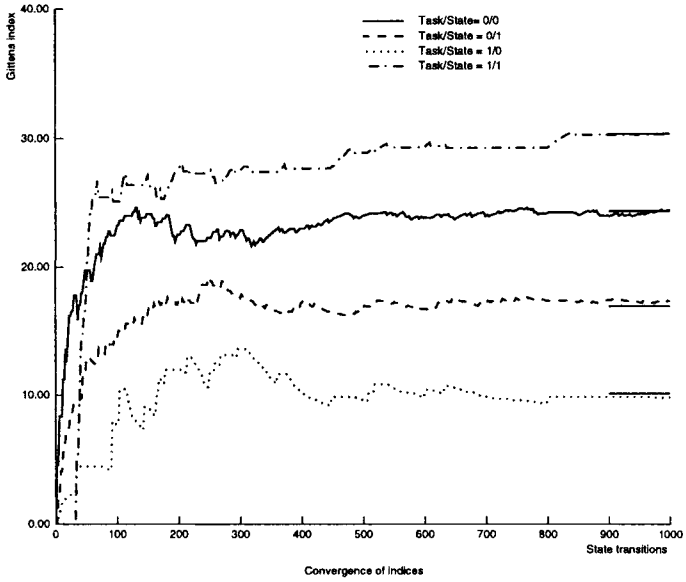


Figure 6: Convergence of Gittins Indices.

algorithm proposed in Section 5. 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 reduction have not been optimized in any sense; 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 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

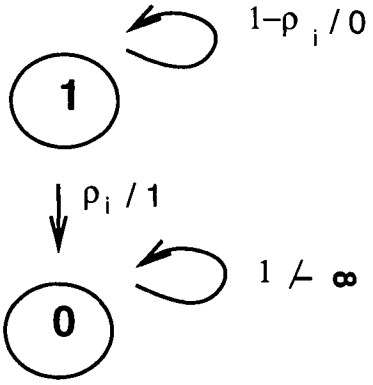


Figure 7: Task model with constant hazard rate.

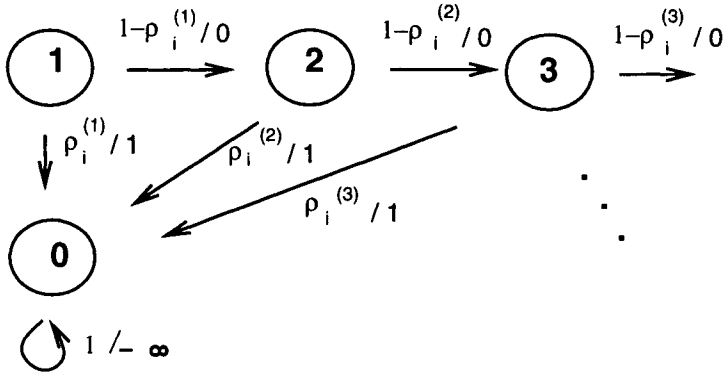


Figure 8: Task model with non-constant hazard rate.

a geometric service time: $Pr\{\tau_i = s\} = \rho_i(1-\rho_i)^{s-1}$ (see Figure 7).

Each task i thus has a constant hazard rate, ρ_i, and it is known that, in order to minimize either mean flow time [3] or mean waiting time, an optimal policy is to activate the tasks in decreasing order of this parameter.

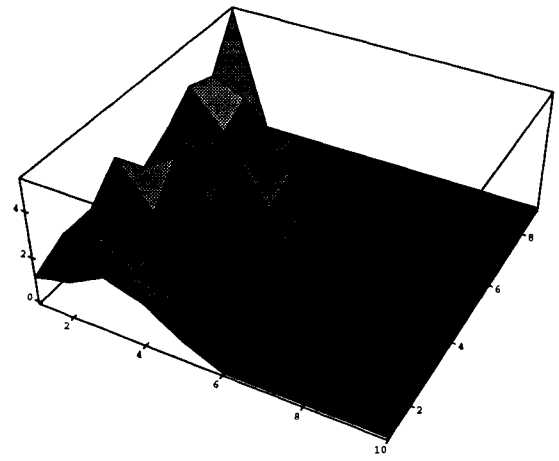
It may be unreasonable to presume that the service-time distributions are known *a priori*. In this case, the reinforcement-learning algorithm of Section 5 can be applied, online, to calculate the respective Gittins indices directly, without building an explicit task model.

Non-constant hazard rate cases can be handled by defining the task models as suitable Markov reward chains, see Figure 8.

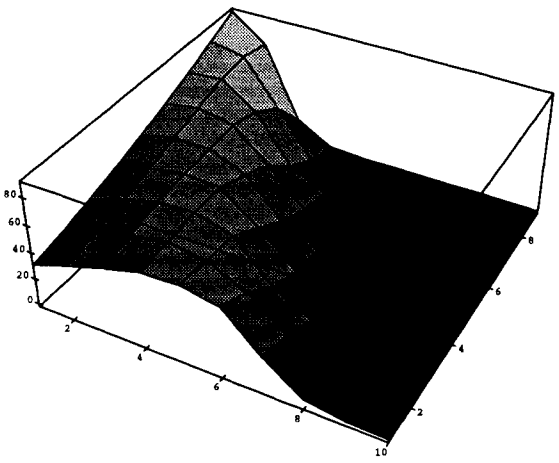
For example, consider a task model for a case in which each task has increasing hazard rate:

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

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



10 trials



1000 trials

Figure 9: Gittins index surface plotted as function of state (x-axis) and task (y-axis).

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 $\rho_i^{(1)} = .1i, \quad i = 1, 2, ..., 9, \lambda = .8$, and the discount factor, γ, was set to 0.99. Again, the reinforcement learning algorithm was applied online in a trial-based way, and the results are presented in Figure 9, 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 two stages of the learning process.

It may be seen that the index values gradually "un-

roll" 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 schedule the tasks non-preemptively, highest hazard-rate first. The plots of Figure 9 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 mentioned at the end of Section 2, it would entail a state-set size on the order of fifty million ($\approx 9\times9\times8\times8\times7\times7\times6\times6\times5$) 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.

The book by Gittins [Gittins, 1989] contains many further applications of the bandit formulation to job scheduling, resource allocation, sequential random sampling, and random search.

7 CONCLUSION

This paper has traced the following chain of reasoning:

- A multi-armed bandit is a Markov decision process (MDP) possessing a special decompositional (Cartesian-product) structure.
- Optimal policies for bandit 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 5.)

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 reinforcement learning algorithm presented in Section 5 has undergone only preliminary empirical testing; its convergence could be accellerated through the utilization of function approximators for representing Q-values or through thoughtful selection of learning rate parameters, which raises an interesting issue:

An example in Section 6 considered a problem of stochastic scheduling as a specific instance 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 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."

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.

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

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

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

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.

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.

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

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