Towards Q-learning the Whittle Index for Restless Bandits

Jing Fu[1], Yoni Nazarathy[2], Sarat Moka[2] and Peter G. Taylor[1]

***Abstract*— We consider the multi-armed restless bandit problem (RMABP) with an infinite horizon average cost objective. Each *arm* of the RMABP is associated with a Markov process that operates in two modes: *active* and *passive*. At each time slot a controller needs to designate a subset of the arms to be active, of which the associated processes will evolve differently from the passive case. Treated as an optimal control problem, the optimal solution of the RMABP is known to be computationally intractable. In many cases, the *Whittle index policy* achieves near optimal performance and can be tractably found. Nevertheless, computation of the *Whittle indices* requires knowledge of the transition matrices of the underlying processes, which are sometimes hidden from decision makers. In this paper, we take first steps towards a tractable and efficient reinforcement learning algorithm for controlling such a system. We setup parallel Q-learning recursions, with each recursion mapping to individual possible values of the Whittle index. We then update these recursions as we control the system, learning an approximation of the Whittle index as time evolves. Tested on several examples, our control outperforms naive priority allocations and nears the performance of the fully-informed Whittle index policy.**

I. Introduction

The multi-armed bandit problem (MABP) is a decision making problem that sequentially *activates* one out of I parallel Markov processes and leaves the remaining $I-1$ *passive*: the passive processes will be frozen (with no state transition) until they become active. Gittins [1] proved the optimality of a simple *index policy* for the MABP, subsequently referred to as the *Gittins index policy*, which always activates the process with the highest state-dependent *index*.

Whittle [2] extended the conventional MABP to a *restless* case: the state of each process can change even when it is not activated. This extended version is referred to as the *restless multi-armed bandit problem* (RMABP), which further allows there to be $K \geq 1$ active processes at each decision making epoch. In general, RMABP are provably PSPACE-hard [3]. Still, like the Gittins case there is the possibility of using an index policy. In [2], Whittle proposed an index policy, referred to as the *Whittle index policy*, and conjectured it to be asymptotically optimal as the number of parallel processes tends to infinity. Later in [4], Weber and Weiss proved the asymptotic optimality of Whittle index policy under certain conditions. See [5] for detailed survey.

[1]Jing Fu and Peter G. Taylor are with School of Mathematics and Statistics, the University of Melbourne, Parkville, VIC 3010, Australia jing.fu;taylorpg@unimelb.edu.au

[2]Yoni Nazarathy and Sarat Moka are with the School of Mathematics and Physics, The University of Queensland, Australia y.nazarathy;s.babumoka@uq.edu.au

The Whittle index policy, similar to Gittins index policy, always prioritizes the processes with the highest state-dependent indices, referred to as the *Whittle indices*, which are calculated by solving *sub-problems* with remarkably reduced state spaces. The (R)MABP has been widely applied in resource allocation problems such as job scheduling in cloud computing [6], channel detection in communications [7], the health care system [8], and dynamic posted pricing [9]. However, calculation or approximation of Whittle indices generally requires full knowledge of the transition matrices of all the Markov processes. In practice, this is rarely known by system controllers.

Problems based on Markov decision process (MDP) with hidden transition matrices have been analyzed through Q-learning (or deep Q-learning) for decades [10]–[13]. Although (deep) Q-learning has been demonstrated to be powerful in approximating value functions, the convergence time to an optimal solution for a problem with a large state space is still an open question. Hence Q-learning of RMABP using a straightforward approach is not applicable.

Borkar and Chadha [14] adapted a Q-learning approach (temporal difference method), which was proposed in [12], to approximate the Whittle index in a RMABP with continuous state space. Given a fixed Whittle index, they proved the convergence of the differential cost function (or bias function). In particular, they focused on a specific case where the Whittle index acted as a monotonic function of the state of each of the parallel processes of the RMABP, and the transition matrices of processes were assumed to be known. Our work in this paper focuses on a RMABP with discrete state space but does not assume a specific class of Whittle indices nor any knowledge of the transition matrices. We rather adapt a Q-learning algorithm for the Whittle sub-problem, presenting an algorithm which experimentally appears to work well in several RMABP settings.

To the best of our knowledge, this is the first work that adapts the Q-learning technique to a generic RMABP with completely concealed transition matrices. We demonstrate by simulations that our policy, referred to as *Q-learning the Whittle Index Controller* (QWIC), significantly outperforms all benchmark policies except the fully-informed Whittle index policy, which assumes knowledge of the transition matrices. Note that this fully-informed Whittle index is illustrated as a performance upper bound, which is not applicable when the transition matrices are hidden.

II. Model

We use $\mathbb{N}_+$ and $\mathbb{N}_0$ as the set of positive and non-negative integers, respectively, and for any $N \in \mathbb{N}_+$, let $[N]$, represent the set $\{1, 2, \ldots, N\}$. Let $\mathbb{R}$ be the set of all reals.

Consider I parallel *projects*, each of which is associated with a discrete-time Markov process, denoted by $\{X_i(t), t \in \mathbb{N}_0\}$, $i \in [I]$. A project can behave in two modes, *active* and *passive*, with potentially different transition matrices. At time t, we can choose to activate a project $i \in [I]$ or not, indicated by an *action variable* $A_i(t) \in \{0,1\}$: if $A_i(t) = 1$, the project is active; otherwise, passive.

In this paper, all the I projects are identical in that they have the same state space, $\mathscr{X}$, and the same state transition matrices, $\mathcal{P}^1, \mathcal{P}^0 \in [0,1]^{|\mathscr{X}|\times|\mathscr{X}|}$ for active and passive modes, respectively. The state space $\mathscr{X}$ is assumed to be finite with states labeled by $x = 1, 2, \ldots, |\mathscr{X}|$. Let $P^1_{x,x'}$ and $P^0_{x,x'}$ $(x, x' \in \mathscr{X})$ represent the entries of $\mathcal{P}^1$ and $\mathcal{P}^0$ at row x and column x', respectively. Let $\boldsymbol{X}(t) \coloneqq (X_i(t) : i \in [I])$ and $\boldsymbol{A}(t) \coloneqq (A_i(t) : i \in [I])$.

The process of the entire system, $\{\boldsymbol{X}(t), \boldsymbol{A}(t)\}_{t=0}^{\infty}$, acts as a *Markov decision process* (MDP) with $\mathscr{X}^I$ and $\{0,1\}^I$ the state and action spaces, respectively. The rule that decides the probability of $\boldsymbol{A}(t)$ taking values in $\{0,1\}^I$ is referred to as a *policy*. In particular, we limit this work to stationary policies; that is, $A_i(t) \coloneqq a_i(\boldsymbol{X}(t))$, which is a function of state vector, and we refer to $a_i(\cdot)$ as the action variable for project $i \in [I]$. In this context, all action variables $a_i(\boldsymbol{x})$ for $i \in [I]$ and $\boldsymbol{x} \in \mathscr{X}^I$ determine a policy, denoted by ϕ. By slightly abusing notation, we rewrite the action and state variables as $a_i^\phi(\cdot)$ and $\boldsymbol{X}^\phi(t)$ to indicate their dependencies on policy ϕ. Let $\boldsymbol{a}^\phi(\boldsymbol{x}) \coloneqq (a_i^\phi(\boldsymbol{x}) : i \in [I])$ for any $\boldsymbol{x} \in \mathscr{X}^I$, and Φ represent the set of all such policies ϕ.

By taking an action $a_i^\phi(\boldsymbol{x}) = a$ in state $\boldsymbol{X}^\phi(t) = x$ $(\boldsymbol{x} \in \mathscr{X}^I,\ a \in \{0,1\})$ at time t, an expected reward $R(x_i, a)$, which is bounded, for project $i \in [I]$ is generated. The total reward of the system generated at each time slot is the sum of the rewards for each project. At each time t, we are required to activate exactly $K \leq I$ of the I projects, so that

$$\sum_{i\in[I]} a_i^\phi(\boldsymbol{X}^\phi(t)) = K, \tag{1}$$

and we aim to maximize the long-run average reward of the entire system. To this end, for initial state $\boldsymbol{x}(0)$, our optimization problem can be written as

$$\max_{\phi\in\Phi} \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i\in[I]} \mathbb{E}[R(X_i^\phi(t), a_i^\phi(\boldsymbol{X}^\phi(t)))], \tag{2}$$

subject to (1).

This problem is a standard *Restless Multi-Armed Bandit Problem* (RMABP), of which state space size is increasing exponentially with the number of projects I. The RMABP has been proved to be PSPACE-hard [3]. Thus, we resort to simple, tractable policies that approximate optimality. As mentioned in Section I, an *index policy*, which prioritizes projects according to the descending order of state-dependent *indices*, is a promising technique. Given these indices, an index policy will always prioritizes the K projects with the highest K indices, which represent the marginal rewards of activating projects in corresponding states. In particular, we consider the well-known *Whittle indices*.

(a) (b)

Fig. 1. The underlying project Markov chains: (a) active; and (b) passive.

III. A Basic Example

We start with an simple problem with state space $\mathscr{X} = \{1,2,3,4\}$ and reward $R(1,a) = -1$, $R(4,a) = 1$, $R(2,a) = R(3,a) = 0$, for $a \in \{0,1\}$. The process $\{X_i^\phi(t), t \in \mathbb{N}_0\}$ (that is, project $i \in [I]$) either remains in its current state or increments positively if it is active, while it either remains in its current state or increments negatively if it is passive. State increments and decrements are with 'wrap-around' as in Figure 1. These dynamics follow,

$$\mathcal{P}^1 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & & \\ & \frac{1}{2} & \frac{1}{2} & \\ & & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & & & \frac{1}{2} \end{bmatrix} \quad \text{and} \quad \mathcal{P}^0 = \begin{bmatrix} \frac{1}{2} & & & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & & \\ & \frac{1}{2} & \frac{1}{2} & \\ & & \frac{1}{2} & \frac{1}{2} \end{bmatrix}.$$

To gain intuition for this problem consider a relatively big system with $I = 500$ projects where only a tenth of the projects are active at any time. That is, $K = 50$. This implies that at every time step, 450 passive projects are subject to the 'decrementing action' while only 50 active projects are incremented. Since both $\mathcal{P}^1$ and $\mathcal{P}^0$ are doubly-stochastic (columns sum up to 1 in addition to rows), the individual stationary distribution is uniform. Hence in particular, a project that has been passive for some time is likely to have a state that is approximately uniformly distributed on $\mathscr{X}$. Also, since most of the projects are passive (nine tenths), the average reward per time step is approximately 0 since the -1 reward from $x = 1$ is offset by the $+1$ reward from $x = 4$. However, as the controller is able to activate 50 projects, it finds an average of about $450/4 = 112.5$ projects in state 3 at any time. From these, it can arbitrarily select 50 projects and cause an average of 25 (since the transition probability is $1/2$) to transition to state 4 in the next time step. Hence any policy that chooses to activate projects in state 3 whenever possible will almost always have 50 such possibilities at its disposal and will move 25 projects on average into state 4. Then, each project that has just transitioned to 4 by incrementing, will stay an average of 2 time steps (expected sojourn time with transition probability $1/2$) at state 4 until returning to state 3. This yields an average reward of 50 per time step. This means that the average reward per arm is $50/500 = 0.1$.

This analysis of the 'give priority to projects in state 3' policy is not more than an intuitive explanation. In fact, if we increase K to for example $K = 250$ the analysis is not as simple as further prioritization needs to take place between the states. Nevertheless, it serves our purpose as we now use this example in two ways. First we consider how the Whittle index can be computed for such an example and then we show how our QWIC policy can obtain similar behavior even when $\mathcal{P}^0$ and $\mathcal{P}^1$ are not known.

IV. Whittle Indices

The Whittle indices are real values assigned to each state in $\mathscr{X}$ and are computed offline by optimizing all the I project processes independently. This idea remarkably reduces the computational complexity of solving a problem over state space $\mathscr{X}^I$ to that over $\mathscr{X}$.

The Whittle index policy has later been proved to be asymptotically optimal under certain conditions [4]. Also, the asymptotic optimality of the Whittle index policy or Whittle-like index policies has been proved or discussed in recent work [6], [15], [16]. In particular, for value functions that solve

$$g+V(x,\lambda)=\max\Bigg\{R(x,1)-\lambda+\sum_{x'\in\mathscr{X}}P^1_{x,x'}V(x',\lambda),\\ R(x,0)+\sum_{x'\in\mathscr{X}}P^0_{x,x'}V(x',\lambda)\Bigg\} \quad (3)$$

with $x\in\mathscr{X}$ and $g,\lambda\in\mathbb{R}$, the Whittle index (if it exists) of a state $x\in\mathscr{X}$ is a specific value $\lambda(x)\in\mathbb{R}$, such that,

$$R(x,1)-\lambda(x)+\sum_{x'\in\mathscr{X}}P^1_{x,x'}V(x',\lambda(x))\\ =R(x,0)+\sum_{x'\in\mathscr{X}}P^0_{x,x'}V(x',\lambda(x)). \quad (4)$$

If the $\lambda(x)$ in (4) is not unique, then we can take any one.

We briefly explain the notation in equation (3). For a given $\lambda\in\mathbb{R}$, consider a MDP that evolves in the same way as a project defined in Section II, except that this MDP gains reward $R(x,1)-\lambda$ if it is active in state $x\in\mathscr{X}$. We refer to such a MDP as a λ-*project*. For a given $\lambda\in\mathbb{R}$, the $g\in\mathbb{R}$ in (3) is a given parameter that is equal to the maximized long-run average revenue of this λ-project. An optimal policy, which maximizes the right hand side of (3) for all $x\in\mathscr{X}$, will maximize the long-run average revenue of the λ-project [17]. The Whittle index (if it exists) for state $x\in\mathscr{X}$ is the specific value of λ such that active and passive actions in state x make no difference in terms of the average revenue of the λ-project, as described in (4).

Nevertheless, the existence of Whittle indices, referred to as *indexability*, for a RMABP is hard to verify. To compute these indices, conventional optimization techniques, such as value or policy iteration, require the transition matrices $\mathcal{P}^1$ and $\mathcal{P}^0$ to be known by system controllers. In this paper, we consider a more practical case where $\mathcal{P}^1$ and $\mathcal{P}^0$ are unknown.

For our example problem, it can be shown that

$$\lambda(1)=-1/2,\quad \lambda(2)=1/2,\quad \lambda(3)=1,\quad \lambda(4)=-1.$$

Hence according to the Whittle index and in agreement with the discussion above, top priority is given to projects with state 3. Then, Whittle gives second priority to projects with 2, then to state 1 and finally to 4.

Such priorities of states, indicated by Whittle indices, coincide with our intuition (presented in Section III) about the transition matrices and reward functions. Simulations on this example will be provided later in Section V-B, after we introduce our algorithm that does not assume any knowledge of transition matrices.

V. Q-Learning for Whittle Indices

By time t, we can retrieve information of the stochastic process with unknown transition matrices from the history of states and actions to solve the optimization problem described by (2) and (1). Let $\mathcal{H}(t):=\{\boldsymbol{X}^\phi(\tau),\boldsymbol{A}^\phi(\tau),\tau\in[t]\}$ represent the history of states and actions of the system up to time $t\in\mathbb{N}_0$.

A. Q-Learning

We adapt the *Q-learning* technique to our problem. The Q-learning technique often applies to a MDP without specific knowledge of its transition rules, and is used to learn the value functions according to observations. For instance, consider the value functions described in (3), and for $x\in\mathscr{X}$, $a\in\{0,1\}$, $\lambda\in\mathbb{R}$, define the *Q function* as,

$$g+Q^\lambda(x,a)=R(x,a)-\lambda a+\sum_{x'\in\mathscr{X}}P^a_{x,x'}V(x',\lambda), \quad (5)$$

and $V(x,\lambda)=\max_{a\in\{0,1\}}Q^\lambda(x,a)$, where λ is considered as a given constant.

Also, define the *estimated Q function* as $\hat{Q}^\lambda_t(x,a)$ for $x\in\mathscr{X}$ and $a\in\{0,1\}$, which is the estimated value of $Q^\lambda(x,a)$ according to the observed history $\mathcal{H}_t$. Since all project processes $i\in[I]$ are stochastically identical, we start our discussion with any of them and write the state variable as $X^\phi(t)$ and $A^\phi(t)$ by omitting the subscript i.

Q-learning is an iterative process that starts out with a set of $\hat{Q}^\lambda_0(x,a)$ for all $x\in\mathscr{X}$ and $a\in\{0,1\}$ and updates them. The process is operated in a way that optimizes the estimated Q function. For $t\in\mathbb{N}_+$, it takes the action $A^\phi(t)=\arg\max_{a\in\{0,1\}}\{\hat{Q}^\lambda_t(X^\phi(t),a)\}$ and updates according to the iteration in (6), where β and $\{\alpha_t\}_{t=0}^\infty$ are hyper-parameters. Given the observation $X^\phi(t+1)$ at time $t+1$ after taking the action $A^\phi(t)$, equation (6) is a standard method in Q-learning that updates the estimated Q functions. Note that in (6), the transition matrices are not required. However it increments slowly; for $x\in\mathscr{X}$ and $a\in\{0,1\}$, if $x\neq X^\phi(t)$ or $a\neq A^\phi(t)$, the estimated Q function $\hat{Q}^\lambda_{t+1}(x,a)=\hat{Q}^\lambda_t(x,a)$.

The parameter $\beta\in[0,1)$ is a discount parameter that is close to 1 and used to guarantee the convergence of our estimated Q function. From the Blackwell optimality theorem [18], there exists an optimal solution of the β-discounted cumulative reward of the λ-project for all β near 1, and this solution will also be long-run average optimal. We can then translate the role of the unknown parameter g in (5) into this β by setting it close to 1 for all our numerical experiments. The $\{\alpha_t\}_{t=0}^\infty$ is a sequence of reals in $[0,1]$ acting as the *learning rate* of the estimated Q function. If $\sum_{t=0}^\infty\alpha_t=\infty$ and $\sum_{t=0}^\infty\alpha_t^2<\infty$, then, from [10, Proposition 5.5], $\lim_{t\to\infty}\hat{Q}^\lambda_t(x,a)=Q^\lambda(x,a)$ for any $x\in\mathscr{X}$ and $a\in\{0,1\}$.

However, a RMABP is much more complicated than a single project process with a one-dimensional state space.

$$\hat{Q}^{\lambda}_{t+1}(X^{\phi}(t), A^{\phi}(t)) = (1-\alpha_t)\hat{Q}^{\lambda}_{t}(X^{\phi}(t), A^{\phi}(t)) + \alpha_t \left(R(X^{\phi}(t), A^{\phi}(t)) - \lambda A^{\phi}(t) + \beta \max_{a\in\{0,1\}} \hat{Q}^{\lambda}_{t}(X^{\phi}(t+1), a) \right) \quad (6)$$

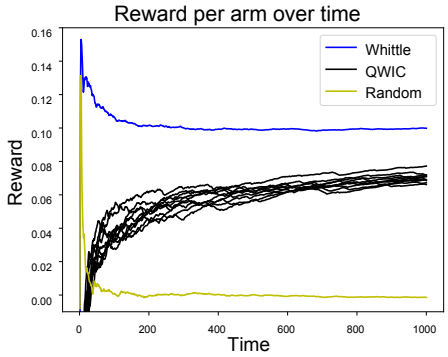


Fig. 2. Accumulated average reward per arm under three policies.

The size of the state space for the RMABP is increasing exponentially in the number of projects, I, with the size of the domain of the Q function increasing accordingly. For most practical problems, the entire learning process cannot be completed within a reasonable time horizon.

B. Q-Learning the Whittle Indices

Recall the definition of the Whittle indices $\lambda(x)$ ($x \in \mathscr{X}$) and the Q function in (4) and (5). We obtain

$$Q^{\lambda(x)}(x, 1) - Q^{\lambda(x)}(x, 0) = 0. \quad (7)$$

For state $x \in \mathscr{X}$, consider a set of candidate values of $\lambda(x)$, denoted by Λ, and, for $\lambda \in \Lambda$, $a \in \{0, 1\}$, $t \in \mathbb{N}_+$, define $Q(\lambda, x, a) = Q^{\lambda}(x, a)$ and $\hat{Q}_t(\lambda, x, a) = \hat{Q}_t(\lambda, x, a)$. In this context, λ becomes part of the state in a Q function, and we approximate the Whittle index $\lambda(x)$ ($x \in \mathscr{X}$) by optimizing

$$\min_{\lambda\in\Lambda} |\hat{Q}_t(\lambda, x, 1) - \hat{Q}_t(\lambda, x, 0)| \quad (8)$$

for each $t \in \mathbb{N}_+$. Define $\hat{\lambda}_t(x)$ as the *estimated Whittle index* for state $x \in \mathscr{X}$ at time t. The $\hat{\lambda}_t(x)$ is updated to be the λ minimizing (8).

On the other hand, these estimated Whittle indices $\hat{\lambda}_t(\cdot)$ will indicate the priorities of projects at time t: the K projects with highest $\hat{\lambda}_t(X_i^{\phi}(t))$ will be activated ($A_i^{\phi}(t) = 1$), and all the others will be passive. The estimated Q function $\hat{Q}_{t+1}(\cdot, \cdot, \cdot)$ and the estimated Whittle indices $\hat{\lambda}_{t+1}(\cdot)$ are then updated according to new observations $\boldsymbol{X}^{\phi}(t+1)$ and updated $\hat{Q}_{t+1}(\cdot, \cdot, \cdot)$, by invoking (8) and (9).

Equation (9) follows the same manner of updating the estimated Q functions as that in (6), except that parameter λ in (6) becomes a state waiting to be explored in (9). Similar to the conventional Q-learning procedure described in Section V-A, for $x \in \mathscr{X}$, $a \in \{0, 1\}$ and $\lambda \in \Lambda$, if $x \neq X^{\phi}(t)$, $a \neq A^{\phi}(t)$ or $\lambda \neq \hat{\lambda}(X^{\phi}(t))$, then $\hat{Q}_{t+1}(\lambda, x, a) = \hat{Q}_t(\lambda, x, a)$. This indicates that the estimated Q function will be updated only for specific states and actions at each time t. The estimated Q function $\hat{Q}_t(\lambda, \cdot, 1)$ with smaller $\lambda \in \Lambda$ are unlikely to be updated, since a smaller index λ has a lower priority and thus has less chance of being activated.

To address this, we introduce an *exploration probability* $\gamma_t \in [0, 1]$, $t \in \mathbb{N}_+$, to guarantee the exploration opportunities of the Q function for all $\lambda \in \Lambda$ of an active project. At each time t, after the action vector $\boldsymbol{A}^{\phi}(t)$ has been determined according to estimated indices $\hat{\lambda}_t(\cdot)$, with probability γ_t, we

- randomly permute the binary values of elements of $\boldsymbol{A}^{\phi}(t)$; and
- for all $x \in \mathscr{X}$, uniformly choose a value from Λ and assign it to $\hat{\lambda}_t(x)$.

The exploration probability γ_t acts as a lever that balances the time spent on exploration and exploitation, which helps updates of these less popular λ values.

We refer to above described policy as *Q-learning the Whittle Index Controller* (QWIC). For a better explanation, we also provide the pseudo-code of QWIC in Algorithm 1. Later, we replace the superscript ϕ (of the state and action variables) by QWIC if the stochastic process is controlled under QWIC, and retain ϕ to indicate a non-specified policy.

Remark Unlike conventional Q-learning algorithms, here, we learn the Q function and the Whittle indices, which mutually affect the learning results of each other, simultaneously. The estimated Q function summarizes the information collected through exploration, while the estimated Whittle indices stand for the exploitation decisions determined by the information collected so far. As mentioned in Section I, this is the first work that approximates Whittle indices with general but unknown transition matrices.

For the simple example discussed in Section III, Figure 2 reports the results for the long-run average reward obtained from simulations of the process operating under the Whittle Index policy, QWIC (10 trajectories) and a third policy that

$$\hat{Q}_{t+1}(\hat{\lambda}(X^{\phi}(t)), X^{\phi}(t), A^{\phi}(t)) = (1-\alpha_t)\hat{Q}_t(\hat{\lambda}(X^{\phi}(t)), X^{\phi}(t), A^{\phi}(t)) + \alpha_t \left(R(X^{\phi}(t), A^{\phi}(t)) - \hat{\lambda}(X^{\phi}(t))A^{\phi}(t) + \beta \max_{a\in\{0,1\}} \hat{Q}_t(\hat{\lambda}(X^{\phi}(t)), X^{\phi}(t+1), a) \right) \quad (9)$$

Input : $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathscr{X}^I$, $Q_0 \in \mathbb{R}^{\Lambda \times \mathscr{X} \times \{0,1\}}$, $\hat{\lambda}_0 \in \Lambda^{\mathscr{X}}$, $\boldsymbol{a}_0 \in \{0,1\}^I$ and $t \in \mathbb{N}_+$

/* $\boldsymbol{x}_1$ is the newly observed state vector for a tested time slot t. */

/* $\boldsymbol{x}_0, Q_0, \hat{\lambda}_0, \boldsymbol{a}_0$ are the state vector, estimated Q function and Whittle index */

/* and action vector for the previous time slot, respectively. */

Output: $Q_1 \in \mathbb{R}^{\Lambda \times \mathscr{X} \times \{0,1\}}$, $\hat{\lambda}_1 \in \Lambda^{\mathscr{X}}$ and $\boldsymbol{a}_1 \in \{0,1\}^I$

/* $Q_1, \hat{\lambda}_1, \boldsymbol{a}_1$ are the updated Q function, the resulting estimated Whittle */

/* index and the resulting action vector for the tested time slot. */

Function $(Q_1, \hat{\lambda}_1, \boldsymbol{a}_1) \leftarrow$ `QWICPolicy`$(\boldsymbol{x}_0, Q_0, \hat{\lambda}_0, \boldsymbol{a}_0, \boldsymbol{x}_1)$**:**

  $Q_1(\lambda, x, a) \leftarrow Q_0(\lambda, x, a)$ for all $\lambda \in \Lambda$, $x \in \mathscr{X}$ and $a \in \{0,1\}$

  for $i \in [I]$ **do**

    $Q_1(\hat{\lambda}_0(x_{0,i}), x_{0,i}, a_{0,i}) \leftarrow$
    $(1-\alpha_t)\hat{Q}_0(\hat{\lambda}_0(x_{0,i}), x_{0,i}, a_{0,i}) + \alpha_t \left(R(x_{0,i}, a_{0,i}) - \hat{\lambda}(x_{0,i})a_{0,i} + \beta \max_{a\in\{0,1\}} \hat{Q}_0(\hat{\lambda}_0(x_{0,i}), x_{1,i}, a)\right)$

    /* Update Q_1 by plugging parameters in (9). */

  end

  for $x \in \mathscr{X}$ **do**

    $\hat{\lambda}_1(x) \leftarrow \arg\min_{\lambda\in\Lambda} |Q_1(\lambda, x, 1) - Q_1(\lambda, x, 0)|$

  end

  $\mathscr{I} \leftarrow$ a subset of $[I]$: if $i \in \mathscr{I}$, then $\hat{\lambda}(x_{1,i})$ is one of the K largest $\hat{\lambda}(x_{1,i'})$ among all $i' \in [I]$

    /* Guarantee $|\mathscr{I}| = K$ by breaking tie cases randomly. */

  for $i \in [I]$ **do**

    if $i \in \mathscr{I}$ **then**

      $a_{1,i} \leftarrow 1$

    else

      $a_{1,i} \leftarrow 0$

    end

  end

  With probability γ_t, randomizes the elements of $\boldsymbol{a}_1$ in place, and, for all $x \in \mathscr{X}$, $\hat{\lambda}_1(x) \leftarrow$ a uniformly randomly selected value from Λ.

  return $(Q_1, \hat{\lambda}_1, \boldsymbol{a}_1)$

End

Algorithm 1: Pseudo-code for QWIC.

randomly activates 100 out of the 500 processes at each time point. For QWIC, we choose hyper-parameters $\beta = 0.99$, $\gamma_t = \min(2t^{-1/2}, 1)$, $\alpha_t = t^{-1/2}$ and Λ as a uniform grid with 10 points ranging from -1.25 to 1.25.

In Figure 2, the reward curves for QWIC and the Whittle index policy are quite close to each other, both significantly outperforming a policy that chooses active arms randomly. As discussed in Section III, the performance of the random policy is expected to be 0. The Whittle index policy does achieve the highest average reward in Figure 2 and is demonstrated to be promising in this example. However the strength of QWIC is that it doesn't rely on any knowledge of the transition matrices or reward structure.

From Figure 2, we can see the quick convergence of QWIC in terms of average revenue, although there is still a gap between QWIC and the Whittle index policy. As the full knowledge of transition matrices must be assumed for implementing the latter, this gap can be regarded as a cost of exploring hidden information.

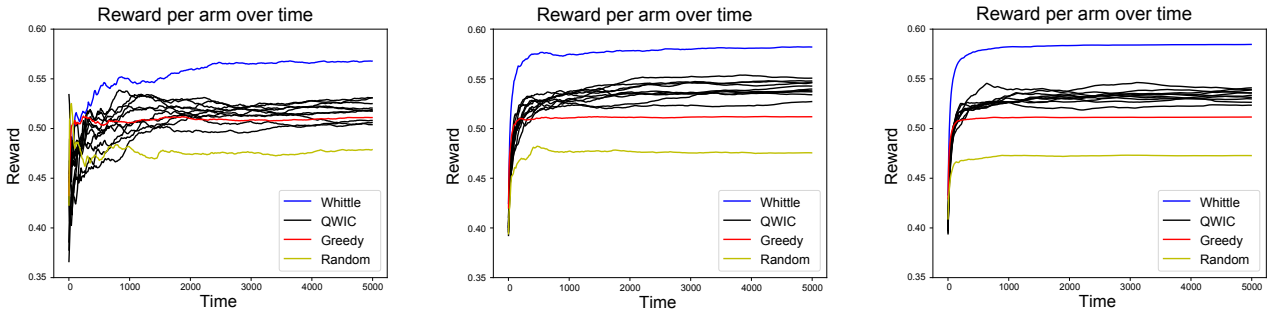
VI. A Further Example: Mentoring Instructions

We now demonstrate the effectiveness of QWIC on a more complicated example. Consider I students with $|\mathscr{X}| = 10$ different study levels. We have $K \in [I]$ mentors available for these students all the time, and aim to maximize the long-run average reward indicated by the study levels of students; that is, let $R(x, a) = c\sqrt{x}$ with $c = 1/\sqrt{10}$ a constant for normalization. Level $x = 1$ is the worst study level, and 10 is the best, however a-priori, the QWIC controller is not aware of that. Student $i \in [I]$ with level $X_i^{\phi}(t) \in \mathscr{X}$ will move into level $X_i^{\phi}(t+1) \in \mathscr{X}$ according to transition matrices

$$\mathcal{P}^1, \mathcal{P}^0 = \begin{bmatrix} 0.3 & 0.7 & & & \\ 0.3 & 0 & \ddots & & \\ & 0.3 & \ddots & 0.7 & \\ & & \ddots & 0 & 0.7 \\ & & & 0.3 & 0.7 \end{bmatrix}, \begin{bmatrix} 0.7 & 0.3 & & & \\ 0.7 & 0 & \ddots & & \\ & 0.7 & \ddots & 0.3 & \\ & & \ddots & 0 & 0.3 \\ & & & 0.7 & 0.3 \end{bmatrix},$$

based on if they get mentoring help (active) or not (passive).

We simulated the student study levels under the Whittle index policy with full knowledge of the transition matrices, QWIC, a policy that randomly selected K students for mentoring at each time point and a greedy policy that always selects students with the best levels. For QWIC, we set $\beta = 0.99$, $\gamma_t = \min(2t^{-1/2}, 1)$, $\alpha_t = 0.3$ and $\Lambda = \{0, 1/15, 2/15, \ldots, 2\}$. In Figure 3(a), we illustrate the long-run average revenue gained by the different policies for a small case with $K = 1$ and $I = 5$. In particular, for the simulation results demonstrated in Figure 3(a), apart from QWIC, all the tested policies are stationary policies for which the long-run average reward. In this context, we simulate one trajectory for each of these stationary policies; and ten trajectories for QWIC by plugging in different seeds of the



(a) (b) (c)

Fig. 3. Accumulated average reward for the Mentoring Instruction system: (a) $K = 1$, $I = 5$; (b) $K = 10$, $I = 50$; and (c) $K = 100$, $I = 500$.

pseudo-random numbers.

In Figure 3(a), although the trajectories of QWIC become almost flat at time $t = 5000$, they diverge from each other. This indicates that the performance of QWIC is sensitive to the exact instances of the simulated trajectory. Also, the trajectories for QWIC form a range of average rewards, all of which outperform Random. The curve for Greedy locates in the range of QWIC trajectories.

With the same system parameters except K and I, Figures 3(b) and 3(c) illustrate the evolution of the long-run average reward with $K = 10$, $I = 50$ and $K = 100$, $I = 500$, respectively. From Figures 3(a)-3(c), we observe that the range of QWIC trajectories is becoming narrower as the number of students (that is, I) increases. In Figure 3(c), QWIC clearly outperforms all other tested policies except the fully-informed Whittle index policy.

When there are more projects, the system receives more feedback information within the same time horizon, and we would expect QWIC to converge faster. This is consistent with our observations in Figures 3(b) and 3(c).

Recall that the ideas of RMABP and Whittle indices were proposed for large scale problems with high-dimensional state spaces. In particular, under certain conditions, the Whittle index policy has been proved to approach optimality as the number of projects tends to infinity [4]. Also, from our simulation results, our QWIC learning algorithm converges faster in large scale problems. Given that it does not require any prior knowledge of the transition matrices this is very promising.

VII. Conclusions

We have taken first steps towards a general Q-learning algorithm for the Whittle index. While further advances are needed to make this algorithm practical, the numerical examples that we present already illustrate promising results.

Acknowledgment

J. Fu and P.G. Taylor's research is supported by the Australian Research Council (ARC) Laureate Fellowship FL130100039 and the ARC Centre of Excellence for the Mathematical and Statistical Frontiers (ACEMS). S. Moka's research is supported by ACEMS, under grant number CE140100049. Y. Nazarathy's research is supported by ARC grant DP180101602. The authors also thank Prof. Vivek Borkar for preliminary discussions.

References

[1] J. C. Gittins, "Bandit processes and dynamic allocation indices," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 148–177, 1979.

[2] P. Whittle, "Restless bandits: Activity allocation in a changing world," *J. Appl. Probab.*, vol. 25, pp. 287–298, 1988.

[3] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of optimal queuing network control," *Math. Oper. Res.*, vol. 24, no. 2, pp. 293–305, May 1999.

[4] R. R. Weber and G. Weiss, "On an index policy for restless bandits," *J. Appl. Probab.*, no. 3, pp. 637–648, Sep. 1990.

[5] J. Niño-Mora, "Dynamic priority allocation via restless bandit marginal productivity indices," *TOP*, vol. 15, no. 2, pp. 161–198, Sep. 2007.

[6] J. Fu, B. Moran, J. Guo, E. W. M. Wong, and M. Zukerman, "Asymptotically optimal job assignment for energy-efficient processor-sharing server farms," *IEEE J. Select. Areas Commun.*, 2016.

[7] K. Liu and Q. Zhao, "Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access," *IEEE Trans. Inform. Theory*, vol. 56, no. 11, pp. 5547–5567, Oct. 2010.

[8] S. Deo, S. Iravani, T. Jiang, K. Smilowitz, and S. Samuelson, "Improving health outcomes through better capacity allocation in a community-based chronic care model," *Operations Research*, vol. 61, no. 6, pp. 1277–1294, Nov.–Dec. 2013.

[9] A. V. den Boer, "Dynamic pricing and learning: historical origins, current research, and new directions," *Surveys in operations research and management science*, vol. 20, no. 1, pp. 1–18, Jun. 2015.

[10] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. MA: Athena Scientific, 1996, vol. 5.

[11] V. S. Borkar, *Stochastic approximation: a dynamical systems viewpoint*. Springer, 2009, vol. 48.

[12] H. Yu and D. P. Bertsekas, "Convergence results for some temporal difference methods based on least squares," *IEEE Transactions on Automatic Control*, vol. 54, no. 7, pp. 1515–1531, Jun. 2009.

[13] T. Jaksch, R. Ortner, and P. Auer, "Near-optimal regret bounds for reinforcement learning," *Journal of Machine Learning Research*, vol. 11, no. Apr, pp. 1563–1600, 2010.

[14] V. S. Borkar and K. Chadha, "A reinforcement learning algorithm for restless bandits," in *2018 Indian Control Conference (ICC)*. IEEE, 2018, pp. 89–94.

[15] I. M. Verloop, "Asymptotically optimal priority policies for indexable and non-indexable restless bandits," *Ann. Appl. Probab.*, vol. 26, no. 4, pp. 1947–1995, Aug. 2016.

[16] J. Fu, B. Moran, and P. G. Taylor, "Restless bandits in action: Resource allocation, competition and reservation," *arXiv: 1804.02100*, Apr. 2018. [Online]. Available: https://arxiv.org/abs/1804.02100

[17] S. M. Ross, *Applied probability models with optimization applications*. Dover Publications (New York), 1992.

[18] D. Blackwell, "Discrete dynamic programming," *The Annals of Mathematical Statistics*, pp. 719–726, 1962.