Computing an Index Policy for Bandits with Switching Penalties

(Invited Paper)

José Niño-Mora
Department of Statistics
Universidad Carlos III de Madrid
C/ Madrid 126
28903 Getafe (Madrid), Spain
jnimora@alum.mit.edu

ABSTRACT

We address the multiarmed bandit problem with switching penalties including both costs and delays. Asawa and Teneketzis (1996) introduced an index for bandits with switching penalties that partially characterizes optimal policies, attaching to each project state a "continuation index" (its Gittins index) and a "switching index," yet only proposed an index algorithm for the case of switching costs. We present a fast decoupled computation method, which in a first stage computes the continuation index and then, in a second stage, computes the switching index an order of magnitude faster in at most $(5/2)n^2 + O(n)$ arithmetic operations for an n-state project. This extends earlier work where we introduced a two-stage index algorithm for the case of switching costs only. We exploit the fact that the Asawa and Teneketzis index is the marginal productivity index of a classic bandit with switching penalties in its semi-Markov restless reformulation, by deploying methods introduced by the author. A computational study demonstrates the dramatic runtime savings achieved by the new algorithm, the near-optimality of the index policy, and its substantial gains against the benchmark Gittins index policy across a wide range of two- and three-project instances.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Markov processes; G.4 [**Mathematical Software**]: Algorithm design and analysis

General Terms

Algorithms

Keywords

Markov decision processes, bandits, restless, switching costs, switching delays, index policies

1. INTRODUCTION

Imagine a firm owning a portfolio of dynamic and stochastic projects, of which it can engage one at a time. To (re)start a project, the firm must incur an upfront lump-sum *startup cost*, as well as a *startup delay*, after which it accrues rewards and operating expenses. The firm can decide, at any time, to abandon the project currently in operation, incurring a lump-sum *shutdown cost*, as well as a *shutdown delay*. It can then switch to another project. Such a firm faces the problem of designing a dynamic project selection policy that maximizes the expected total discounted value of its net earnings. This is an example of the *multi-armed bandit problem with switching penalties* (MABPSP), which is extensively surveyed in [4].

In this and many other applications switching delays play a fundamental role, and should thus be incorporated into corresponding system models. Thus, startup delays may represent, e.g., time to lay up the groundwork or to build up infrastructure, as well as training or learning time for workers. Similarly, shutdown delays may arise, e.g., when dismantling installed infrastructure.

The problem is cast as a *semi-Markov decision process* (SMDP) by modeling projects as *bandits*, i.e., binary-action (active/passive) SMDPs that can only change state while active. In the no switching penalties case, one thus obtains the *multiarmed bandit problem* (MABP), which is optimally solved by the *Gittins index* policy. See [3].

The optimal index solution for the MABP prompted investigation of *priority-index policies* for the MABPSP. As discussed in [2], such policies attach an index $\nu_m(a_m^-, i_m)$ to each bandit m, which is a function of its previous action a_m^- and current state i_m, thus decoupling into a "continuation index" $\nu_m(1, i_m)$ and a "switching index" $\nu_m(0, i_m)$. They observed that "it is obvious that in comparing two otherwise identical arms, one of which was used in the previous period, the one which was in use must necessarily be more attractive than the one which was idle." To be consistent which such a *hysteretic* property, the indices must satisfy

$$\nu_m(1, i_m) \geq \nu_m(0, i_m). \tag{1}$$

Though [2] proved that such policies are not generally optimal under switching costs, [1] introduced an intuitively appealing index for the MABSP, which we refer to henceforth as the *AT index*, both for the case of only switching costs and for that of only switching delays, and showed that it partly characterizes optimal policies. Their continuation index is the bandit's Gittins index, while their switching index is the maximum rate, achievable by stopping rules that engage an initially passive bandit, of expected discounted reward earned minus initial startup cost incurred, per unit of expected discounted time — including the initial delay.

In [1], an index computation method is presented to jointly compute both indices in the case of only switching costs. Yet, no algorithm is given there to compute the index under switching delays. In [12], we have developed a substantially faster two-stage index computation method for bandits with switching costs but no switching delays, drawing on restless bandit indexation. This raises the need to develop an efficient index computation method for bandits with switching penalties that incorporate delays, which is the prime goal of this paper, while the second goal is to investigate empirically the performance of the resulting index policy.

We address such goals in the setting of an extended model allowing state-dependent startup and shutdown costs and delays, through a seemingly indirect route: by exploiting the natural reformulation of a classic bandit with switching penalties as a *semi-Markov restless bandit* — one that can change state while passive — *without* switching penalties, through which the MABSP is cast as a *semi-Markov multiarmed restless bandit problem* (SMARBP).

Such a reformulation allows us to deploy the powerful indexation theory available for restless bandits. This was introduced by Whittle in [14], who introduced an index for restless bandits under the average criterion, albeit only for the limited range that satisfy a so-called *indexability* property. He proposed to use the resulting index policy as a heuristic for the MARBP, which is generally suboptimal. The theory has been developed by the author in work surveyed in [10], grounded on the unifying concept of *marginal productivity index* (MPI).

As mentioned above, such an approach was deployed in [12] for the special case of the model considered herein where there are only switching costs. The mode of analysis was based on establishing that the restless bandits of concern satisfy the *PCL-indexability* conditions in the author's earlier work. Yet, the extension to the present setting including switching delays is not straightforward, as we have found that the resulting restless bandits need not be PCL-indexable.

We will draw on the more recent results announced in [9], where the tractable class of *LP-indexable* bandits — as they are based on *linear programming* (LP) analyses — is introduced, for which the MPI is efficiently computed by an *adaptive-greedy algorithm.* The scope of such an algorithm is thus extended from the class of PCL-indexable bandits to the larger class of LP-indexable bandits.

We deploy here such a theory, by proving and exploiting the fact that the AT index of a bandit with switching costs and delays is precisely the bandit's MPI in its semi-Markov restless reformulation. We establish that such restless bandits are LP-indexable, relative to the family of hysteretic policies consistent with (1), which allows us to compute the index using the adaptive-greedy algorithm referred to above. A work-reward analysis then reveals that such an algorithm decouples into two stages: a first stage that computes the Gittins index and required extra quantities; and a second stage, which is fed the first-stage's output and computes the switching index.

To implement such a scheme, one can use for the first stage any of several $O(n^3)$ algorithms introduced in [8]. For the second stage, we present here a fast switching-index algorithm that performs *at most* $(5/2)n^2 + O(n)$ arithmetic operations, thus achieving an order-of-magnitude improvement that renders negligible the marginal effort to compute the switching index. Such an algorithm is the main contribution of this paper. In the no-switching-delays case addressed in [12], the complexity count dropped to (at most) $n^2+O(n)$.

The paper further reports on a computational study demonstrating that such an improved complexity translates into dramatic runtime savings. The study is complemented by a set of experiments that demonstrate the near-optimality of the index policy and its substantial gains against the benchmark Gittins index policy across an extensive range of two- and three-bandit instances.

Section 2 describes the model, shows how to reduce it to the normalized no shutdown penalties case, defines the AT index, and gives the SMARBP reformulation. Section 3 reviews the indexation theory to be deployed. Section 4 carries out a work-reward analysis of reformulated restless bandits. Section 5 develops the new decoupled index algorithm. Section 6 reports the computational study's results. Section 7 concludes.

Due to space constraints, this extended abstract only states and briefly discusses results. For a full working paper version with proofs of all results and additional material, we refer the reader to [11].

2. MODEL, AT INDEX AND RESTLESS REFORMULATION

2.1 The MABPSP

Consider a collection of M finite-state bandi projects, one of which must be engaged (*active*) at each discrete *decision period* $\tau_k \in \mathbb{Z}_+$, with $0 \leq \tau_k \nearrow \infty$ as $k \to \infty$, while the others are rested (*passive*). Switching projects is costly, involving startup and shutdown costs and delays. We assume that a freshly set up project must be *worked on* for at least one period, and say that a project is *engaged* if it is either being worked on, or undergoing a startup or shutdown delay.

A rested project m occupying state i_m — belonging in its state space N_m — accrues no rewards, i.e., $R_m^0(i_m) \equiv 0$, and its state remains frozen. When freshly engaged, it incurs startup cost $c_m(i_m)$, followed by a discrete random startup delay $\xi_m(i_m) \in \mathbb{Z}_+$ having *z-transform* $\phi_m(z; i_m) \triangleq \mathbb{E}\big[z^{\xi_m(i_m)}\big]$, during which no rewards accrue. When the startup is completed, the project must be worked on, yielding an active reward $R_m^1(i_m) = R_m(i_m)$ and changing state at the following period to j_m with probability $p_m(i_m, j_m)$. After one or more periods at which the project is worked on, it may be rested. If this happens in state j_m, shutdown cost $d_m(j_m)$ is incurred, followed by a random shutdown delay $\eta_m \in \mathbb{Z}_+$ having z-transform $\psi_m(z) \triangleq \mathbb{E}\big[z^{\eta_m}\big]$, during which no rewards accrue. Then, the project must be rested for at least one period. We thus allow startup delay distributions to be state-dependent, while shutdown delay's are constant — due to results in Section 2.2. Rewards and costs are time-discounted with factor $0 < \beta < 1$. We will find it convenient to write $\phi_m(\beta; i_m)$ and $\psi_m(\beta)$ as $\phi_m(i_m)$ and ψ_m.

Actions are chosen by adoption of a *scheduling policy* $\boldsymbol{\pi}$, drawn from the class $\boldsymbol{\Pi}$ of *admissible policies*, which are nonanticipative relative to the history of states and actions, and engage one project at a time. Focus on such a version, instead of on that where *at most* one project can be engaged, is without loss of generality. The MABPSP is to find an admissible policy that maximizes the expected total discounted value of rewards earned minus switching costs incurred.

We denote by $X_m(t)$ and $a_m(t) \in \{0,1\}$ the prevailing state and action for project m at period t, respectively, where $a_m(t) = 1$ (resp. $a_m(t) = 0$) means that the project is engaged (resp. rested). Since it must be specified whether each project m is initially set up, we denote such status by $a_m^-(0)$. We define the project's *augmented state* to be $\hat{X}_m(t) \triangleq (a_m^-(t), X_m(t))$, which moves over the *augmented state space* $\hat{N}_m \triangleq \{0,1\} \times N_m$. The *joint augmented state* is thus $\hat{\mathbf{X}}(t) \triangleq \big(\hat{X}_m(t)\big)_{m=1}^M$, and the *joint action process* is $\mathbf{a}(t) \triangleq \big(a_m(t)\big)_{m=1}^M$.

2.2 Reduction to No Shutdown Penalties Case

We show in this section that it suffices to restrict attention to the no shutdown penalties case, without loss of generality. Suppose that, at a certain time, which we take to be $t = 0$, a project is freshly engaged for a random duration given by a stopping-time rule τ. Let us drop the project label m, and denote by $\mathbf{R} = (R_j)$, $\mathbf{c} = (c_j)$ and $\mathbf{d} = (d_j)$ the project's state-dependent active reward, startup and shutdown cost vectors. Let us further denote by $\boldsymbol{\phi} = (\phi_j)$ the project's state-dependent startup z-transform vector, evaluated at $z = \beta$, and let ψ denote the corresponding constant shutdown z-transform value. We can thus write the expected discounted net reward earned on the project during such a time span, starting at $X(0) = i$, as

$$f_i^\tau\big(\mathbf{R}, \mathbf{c}, \mathbf{d}, \boldsymbol{\phi}, \psi\big) \triangleq \mathbb{E}_i^\tau\left[-c_i + \sum_{t=0}^{\tau-1} R_{X(t)}\beta^{t+\xi_i} - d_{X(\tau)}\beta^{\xi_i+\tau}\right], \tag{2}$$

where ξ_i is the startup delay starting at i. The discounted amount of *work* expended on the project is

$$g_i^\tau\big(\boldsymbol{\phi}, \psi\big) \triangleq \mathbb{E}_i^\tau\left[\frac{1-\beta^{\xi_i}}{1-\beta} + \sum_{t=0}^{\tau-1}\beta^{t+\xi_i} + \frac{1-\beta^\eta}{1-\beta}\beta^{\xi_i+\tau}\right], \tag{3}$$

where, as mentioned above, both the startup and shutdown delays ξ_i and η are counted as "work."

We have the following result, where $\mathbf{I}$ is the identity matrix indexed by the state space N, $\mathbf{P} = (p_{ij})_{i,j\in N}$ is the transition probability matrix, and $\mathbf{0}$ is a vector of zeros. Let

$$\widetilde{c}_j \triangleq c_j + \phi_j d_j, \quad \widetilde{\phi}_j \triangleq \psi\phi_j, \quad \widetilde{\mathbf{R}} \triangleq \frac{1}{\psi}\big\{\mathbf{R} + (\mathbf{I} - \beta\mathbf{P})\mathbf{d}\big\}. \tag{4}$$

LEMMA 2.1.

(a) $f_i^\tau\big(\mathbf{R}, \mathbf{c}, \mathbf{d}, \boldsymbol{\phi}, \psi\big) = f_i^\tau\Big(\widetilde{\mathbf{R}}, \widetilde{\mathbf{c}}, \mathbf{0}, \widetilde{\boldsymbol{\phi}}, 1\Big)$.

(b) $g_i^\tau\big(\boldsymbol{\phi}, \psi\big) = g_i^\tau\big(\widetilde{\boldsymbol{\phi}}, 1\big)$.

Lemma 2.1 shows how to eliminate shutdown penalties: one incorporates them into modified startup costs and delay transforms, as well as active rewards, given by the transformations in (3). Note that, in the case $c_j \equiv c$ and $d_j \equiv d$, one obtains $\widetilde{c}_j \equiv c + d\phi_j$ and $\widetilde{R}_j = \{R_j + (1-\beta)d\}/\psi$.

We hence focus our discussion henceforth in the *normalized* no shutdown penalties case.

2.3 The AT Index

We next define the AT index for a project, whose label m we drop from the notation, extending the definitions in [1] to the present setting. The continuation AT index is

$$\nu_{(1,i)}^{\mathrm{AT}} \triangleq \max_{\tau>0} \frac{\mathbb{E}_i^\tau\left[\sum_{t=0}^{\tau-1} R_{X(t)}\beta^t\right]}{\mathbb{E}_i^\tau\left[\sum_{t=0}^{\tau-1}\beta^t\right]}, \tag{5}$$

where τ is a stopping time/rule that engages a project starting at state i needing no setup; hence, $\nu_{(1,i)}^{\mathrm{AT}}$ is precisely the project's Gittins index. The switching AT index is

$$\begin{aligned}\nu_{(0,i)}^{\mathrm{AT}} &\triangleq \max_{\tau>0} \frac{-c_i + \mathbb{E}_i^\tau\left[\beta^{\xi_i}\sum_{t=0}^{\tau-1} R_{X(t)}\beta^t\right]}{\mathbb{E}_i^\tau\left[\sum_{t=0}^{\xi_i-1}\beta^t + \beta^{\xi_i}\sum_{t=0}^{\tau-1}\beta^t\right]} \\ &= \max_{\tau>0} \frac{-c_i + \phi_i\mathbb{E}_i^\tau\left[\sum_{t=0}^{\tau-1} R_{X(t)}\beta^t\right]}{\dfrac{1-\phi_i}{1-\beta} + \phi_i\mathbb{E}_i^\tau\left[\sum_{t=0}^{\tau-1}\beta^t\right]},\end{aligned} \tag{6}$$

where now τ is a stopping time/rule that engages a project starting at i which needs to be set up.

Notice that, writing $f_i^\tau = \mathbb{E}_i^\tau\left[\sum_{t=0}^{\tau-1} R_{X(t)}\beta^t\right]$ and $g_i^\tau = \mathbb{E}_i^\tau\left[\sum_{t=0}^{\tau-1}\beta^t\right]$, we have

$$\frac{f_i^\tau}{g_i^\tau} - \frac{-c_i + \phi_i f_i^\tau}{\dfrac{1-\phi_i}{1-\beta} + \phi_i g_i^\tau} = \frac{1}{g_i^\tau}\,\frac{(1-\beta)c_i g_i^\tau + (1-\phi_i)f_i^\tau}{1-\phi_i + (1-\beta)\phi_i g_i^\tau} \geq 0,$$

provided that $c_j \geq 0$ and $R_j \geq 0$ for $j \in N$. In such a case, on which we focus our analyses, it follows from the above that $\nu_{(1,i)}^{\mathrm{AT}} \geq \nu_{(0,i)}^{\mathrm{AT}}$, consistently with (1).

2.4 Semi-Markov Restless Reformulation

Taking $\hat{X}_m(t)$ as the state of each project m yields a reformulation of the MABPSP as a SMARBP *without* switching penalties, having joint state and action processes $\hat{\mathbf{X}}(t)$ and $\mathbf{a}(t)$, where actions can only be taken at the sequence τ_k of decision periods discussed above. The rewards and dynamics for restless project m in such a reformulation are as follows. If at period τ_k the project occupies (augmented) state $(1, i_m)$ and is engaged, the active reward $\hat{R}_m^1(1, i_m) \triangleq R_m(i_m)$ is earned, and the state moves at the next decision period $\tau_{k+1} = \tau_k + 1$ to $(1, j_m)$ with active transition probability $\hat{p}_m^1\big((1, i_m), (1, j_m)\big) \triangleq p_m(i_m, j_m)$. If the project is instead rested, no passive reward is earned, i.e., $\hat{R}_m^0(1, i_m) \equiv 0$, and the state moves at the next decision period $\tau_{k+1} = \tau_k + 1$ to $(0, i_m)$ with a unity passive transition probability, i.e., $\hat{p}_m^0\big((1, i_m), (0, i_m)\big) \equiv 1$.

If the restless project occupies at τ_k state $(0, i_m)$ and is engaged, the expected active reward

$$\begin{aligned}\hat{R}_m^1(0, i_m) &\triangleq \mathbb{E}[-c_m(i_m) + \beta^{\xi_m(i_m)}R_m(i_m)] \\ &= -c_m(i_m) + \phi_m(i_m)R_m(i_m)\end{aligned} \tag{7}$$

accrues up to the next decision period $\tau_{k+1} = \tau_k + \xi_m(i_m) + 1$, at which its state moves to $(1, j_m)$ with active transition probability $\hat{p}_m^1\big((0, i_m), (1, j_m)\big) \triangleq p_m(i_m, j_m)$. If the project is instead rested, no passive reward accrues, i.e., $\hat{R}_m^0(0, i_m) \equiv 0$, and the state remains frozen at the next decision period $\tau_{k+1} = \tau_k + 1$, i.e., $\hat{p}_m^0\big((0, i_m), (0, i_m)\big) \equiv 1$.

We can thus formulate the MABPSP as the SMARBP

$$\max_{\boldsymbol{\pi} \in \boldsymbol{\Pi}} \mathbb{E}_{\hat{\boldsymbol{\imath}}}^{\boldsymbol{\pi}} \left[\sum_{k=0}^{\infty} \sum_{m=1}^{M} \hat{R}_m^{a_m(\tau_k)}\big(\hat{X}_m(\tau_k)\big)\beta^{\tau_k} \right], \tag{8}$$

where $\mathbb{E}_{\hat{\boldsymbol{\imath}}}^{\boldsymbol{\pi}}[\cdot]$ denotes expectation under policy $\boldsymbol{\pi}$ conditional on the initial joint state $\hat{\mathbf{X}}(0) = \hat{\boldsymbol{\imath}}$.

3. RESTLESS PROJECT INDEXATION

We discuss in this section the indexation theory referred to in Section 1, as it applies to a single project m as above — in its restless reformulation. We hence drop again the project label m henceforth, so that, e.g., N and $\hat{N} \triangleq \{0, 1\} \times N$ denote the project's original and augmented state spaces. We denote by Π the space of admissible project operating policies π. We assume that (normalized) startup costs and active rewards are nonnegative.

Assumption 3.1. *For $i \in N$:*

(i) $c_i \geq 0$*; and*

(ii) $R_i \geq 0$.

3.1 Indexability and the MPI

We use two criteria to evaluate a policy π, relative to an initial state (a_0^-, i_0): the *reward measure*

$$f_{(a_0^-, i_0)}^{\pi} \triangleq \mathbb{E}_{(a_0^-, i_0)}^{\pi} \left[\sum_{k=0}^{\infty} \hat{R}\big(\hat{X}(\tau_k)\big)\beta^{\tau_k} \right],$$

which gives the expected total discounted value of *net rewards* — net of switching costs — that accrue on the project; and the *work measure*

$$g_{(a_0^-, i_0)}^{\pi} \triangleq \mathbb{E}_{(a_0^-, i_0)}^{\pi} \left[\sum_{t=0}^{\infty} a(t)\beta^{t} \right],$$

which gives the expected total discounted work expended. We will actually consider the average measures f^π and g^π obtained by drawing the initial state from an arbitrary positive probability mass function $p_{(a^-, i)} > 0$ for $(a^-, i) \in \hat{N}$.

Imagining that work is paid for at *wage* rate ν leads us to consider the *ν-wage problem*

$$\max_{\pi \in \Pi} f^\pi - \nu g^\pi, \tag{9}$$

which is to find an admissible project operating policy achieving the maximum value of net rewards earned minus labor costs incurred. We use (9) to *calibrate* the *marginal value of work* at each state, by analyzing the structure of optimal policies as ν varies.

MDP theory ensures that for every wage $\nu \in \mathbb{R}$ there exists an optimal policy that is stationary deterministic and independent of the initial state. Any such policy is characterized by its *active set*, or subset of states where it prescribes to engage the project. We write active sets as

$$S_0 \oplus S_1 \triangleq \{0\} \times S_0 \cup \{1\} \times S_1, \quad S_0, S_1 \subseteq N.$$

Thus, the policy that we denote by $S_0 \oplus S_1$ engages the project when it was previously rested (resp. engaged) if the original state $X(t)$ lies in S_0 (resp. in S_1).

Hence, to any wage ν there corresponds a *unique minimal optimal active set* $S_0^*(\nu) \oplus S_1^*(\nu) \subseteq \hat{N}$, which consists of all augmented states where engaging the project is the only optimal action. We say that the project is *indexable* if there exists an *index* $\nu_{(a^-, i)}^*$ for $(a^-, i) \in \hat{N}$ such that, for $\nu \in \mathbb{R}$,

$$S_0^*(\nu) = \big\{(0, i) \colon \nu_{(0,i)}^* > \nu\big\} \text{ and } S_1^*(\nu) = \big\{(1, i) \colon \nu_{(1,i)}^* > \nu\big\}.$$

We then say that $\nu_{(a^-, i)}^*$ is the project's *marginal productivity index* (MPI), terming $\nu_{(1,i)}^*$ the *continuation MPI*, and $\nu_{(0,i)}^*$ the *switching MPI*.

Thus, the project is indexable with MPI $\nu_{(a^-, i)}^*$ if it is optimal in (9), to engage (resp. rest) the project when it occupies state (a^-, i) iff $\nu_{(a^-, i)}^* \geq \nu$ (resp. $\nu_{(a^-, i)}^* \leq \nu$). Note that [14]'s original definition of indexability was stated in an equivalent form in terms of optimal passive sets.

To establish indexability and compute the MPI, we have developed in [5, 6, 7, 9] an approach based on positing and then establishing the structure of optimal active sets, as an *active-set family* $\hat{\mathscr{F}} \subseteq 2^{\hat{N}}$ that *contains* all sets $S_0^*(\nu) \oplus S_1^*(\nu)$ as ν varies, under a possibly restricted range of reward/cost parameters. The intuition that, if startup costs satisfy Assumption 3.1, optimal policies should have the hysteretic property that, if it is optimal to engage a project when it was previously rested, then, other things being equal, it should be optimal to engage it when it was previously active, leads us to guess that the right choice of $\hat{\mathscr{F}}$ should be

$$\hat{\mathscr{F}} \triangleq \big\{S_0 \oplus S_1 \colon S_0 \subseteq S_1 \subseteq N\big\}. \tag{10}$$

Notice that $\hat{\mathscr{F}}$ represents a family of policies consistent with (1), which we posit to contain the optimal policies for (9). When $S_0 \neq S_1$, such policies present the *hysteresis region* $S_1 \setminus S_0$, on which project dynamics depend on the previous action. We thus aim to establish indexability relative to such a family, meaning that the project is indexable and $S_0^*(\nu) \oplus S_1^*(\nu) \in \hat{\mathscr{F}}$ for $\nu \in \mathbb{R}$.

3.2 LP-Indexability and Index Algorithm

We next discuss the approach we will deploy to establish indexability and compute the MPI of the restless projects of concern herein, based on showing that they are LP-indexable relative to $\hat{\mathscr{F}}$, and using the adaptive-greedy index algorithm that is valid for such projects.

Given an action $a \in \{0, 1\}$ and an active set $S_0 \oplus S_1 \in \hat{\mathscr{F}}$, denote by $\langle a, S_0 \oplus S_1 \rangle$ the policy that initially takes action a and adopts the $S_0 \oplus S_1$-*active policy* thereafter. Now, for an augmented state (a^-, i) and an active set $S_0 \oplus S_1 \in \hat{\mathscr{F}}$, define the *marginal work measure*

$$w_{(a^-, i)}^{S_0 \oplus S_1} \triangleq g_{(a^-, i)}^{\langle 1, S_0 \oplus S_1 \rangle} - g_{(a^-, i)}^{\langle 0, S_0 \oplus S_1 \rangle}, \tag{11}$$

along with the *marginal reward measure*

$$r_{(a^-, i)}^{S_0 \oplus S_1} \triangleq f_{(a^-, i)}^{\langle 1, S_0 \oplus S_1 \rangle} - f_{(a^-, i)}^{\langle 0, S_0 \oplus S_1 \rangle}, \tag{12}$$

and, when $w_{(a^-, i)}^{S_0 \oplus S_1} \neq 0$, the *marginal productivity measure*

$$\nu_{(a^-, i)}^{S_0 \oplus S_1} \triangleq \frac{r_{(a^-, i)}^{S_0 \oplus S_1}}{w_{(a^-, i)}^{S_0 \oplus S_1}}. \tag{13}$$

We deploy the LP-indexability approach to indexation introduced in [9], which extends the earlier PCL-indexability approach introduced and developed in [5, 6, 7]. For an active set $\hat{S} = S_0 \oplus S_1 \in \hat{\mathscr{F}}$, let

$$\begin{aligned}\partial^{\text{out}}_{\hat{\mathscr{F}}} \hat{S} &\triangleq \{(a^-, i) \in \hat{S}^c \colon \hat{S} \cup \{(a^-, i)\} \in \hat{\mathscr{F}}\} \\ &= \{(0, i) \colon i \in S_1 \setminus S_0\} \cup \{(1, i) \colon i \in S_1^c\},\end{aligned} \tag{14}$$

where $\hat{S}^c \triangleq \hat{N} \setminus \hat{S}$ and $S_1^c \triangleq N \setminus S_1$, be the *outer boundary of $\hat{S}$ relative to $\hat{\mathscr{F}}$*; and let

$$\begin{aligned}\partial^{\text{in}}_{\hat{\mathscr{F}}} \hat{S} &\triangleq \{(a^-, i) \in \hat{S} \colon \hat{S} \setminus \{(a^-, i)\} \in \hat{\mathscr{F}}\} \\ &= \{(1, i) \colon i \in S_1 \setminus S_0\} \cup \{(0, i) \colon i \in S_0\}\end{aligned} \tag{15}$$

be the corresponding *inner boundary*. Note that the rightmost identities in (14)–(15) follow from (10). Now, we require that *set system* $(\hat{N}, \hat{\mathscr{F}})$ be *monotonically connected*, which in the present setting means that:

(i) $\emptyset, \hat{N} \in \hat{\mathscr{F}}$;

(ii) for $\hat{S}, \hat{S}' \in \hat{\mathscr{F}}$ with $\hat{S} \subset \hat{S}'$, $\hat{S}' \cap \partial^{\text{out}}_{\hat{\mathscr{F}}} \hat{S} \neq \emptyset$ and $\hat{S}^c \cap \partial^{\text{in}}_{\hat{\mathscr{F}}} \hat{S}' \neq \emptyset$; and

(iii) for $\hat{S}, \hat{S}' \in \hat{\mathscr{F}}$, $\hat{S} \cap \hat{S}' \in \hat{\mathscr{F}}$ and $\hat{S} \cup \hat{S}' \in \hat{\mathscr{F}}$.

As the reader can immediately verify, the $\hat{\mathscr{F}}$ defined in (10) satisfies indeed such conditions.

We further write below

$$\underline{r}^{\hat{S}} \triangleq \max_{(a^-, j) \in \hat{S}^c, w^{\hat{S}}_{(a^-, j)} = 0} r^{\hat{S}}_{(a^-, j)}$$

$$\overline{r}^{\hat{S}} \triangleq \min_{(a^-, j) \in \hat{S}, w^{\hat{S}}_{(a^-, j)} = 0} r^{\hat{S}}_{(a^-, j)},$$

adopting the convention that the maximum (resp. minimum) over an empty set is $-\infty$ (resp. $+\infty$).

Now, we say that the project is *LP-indexable* relative to $\hat{\mathscr{F}}$, or *LP($\hat{\mathscr{F}}$)-indexable*, if:

(i) $w^{\emptyset}_{(a^-, i)}, w^{\hat{N}}_{(a^-, i)} \geq 0$ for $(a^-, i) \in \hat{N}$, and $\underline{r}^{\emptyset} \leq 0 \leq \overline{r}^{\hat{N}}$;

(ii) for each active set $\hat{S} \in \hat{\mathscr{F}}$, $w^{\hat{S}}_{(a^-, i)} > 0$ for $(a^-, i) \in \partial^{\text{in}}_{\hat{\mathscr{F}}} \hat{S} \cup \partial^{\text{out}}_{\hat{\mathscr{F}}} \hat{S}$; and

(iii) for every wage $\nu \in \mathbb{R}$ there exists an optimal policy for (9) with active set $\hat{S} \in \hat{\mathscr{F}}$.

We further refer to the *adaptive-greedy algorithmic scheme* $\mathrm{AG}_{\hat{\mathscr{F}}}$ shown in Table 1, where $n \triangleq |N|$ denotes the number of project states in the original (nonrestless) formulation. The algorithm produces an output consisting of a string $\{(a_k^-, i_k)\}_{k=1}^{2n}$ of distinct augmented states spanning $\hat{N}$, with $\hat{S}^k \triangleq \{(a_1^-, i_1), \ldots, (a_k^-, i_k)\} \in \hat{\mathscr{F}}$, for $1 \leq k \leq 2n$, along with corresponding index values $\{\nu^*_{(a_k^-, i_k)}\}_{k=1}^{2n}$. Ties for picking the (a_k^-, i_k)'s are broken arbitrarily. We use the term *algorithmic scheme* as it is not yet specified how to compute the required marginal productivity rates.

We will later invoke the following key result, introduced in [9], which refers to a generic restless project and active-set family $\mathscr{F}$.

Theorem 3.2. *An LP($\mathscr{F}$)-indexable project is indexable and algorithm $\mathrm{AG}_{\mathscr{F}}$ computes its MPI.*

Table 1: Version 1 of Algorithmic Scheme $\mathrm{AG}_{\hat{\mathscr{F}}}$.

```
ALGORITHM $\mathrm{AG}_{\hat{\mathscr{F}}}$:
Output: $\{(a_k^-, i_k), \nu^*_{(a_k^-, i_k)}\}_{k=1}^{2n}$

$\hat{S}^0 := \emptyset \oplus \emptyset$
for $k := 1$ to $2n$ do
    pick $(a_k^-, i_k) \in \arg\max\{\nu^{\hat{S}^{k-1}}_{(a^-, i)} \colon (a^-, i) \in \partial^{\text{out}}_{\hat{\mathscr{F}}} \hat{S}^{k-1}\}$
    $\nu^*_{(a_k^-, i_k)} := \nu^{\hat{S}^{k-1}}_{(a_k^-, i_k)}$;  $\hat{S}^k := \hat{S}^{k-1} \cup \{(a_k^-, i_k)\}$
end { for }
```

Using the definition of $\hat{\mathscr{F}}$ in (10) yields the more explicit *Version 2* of the algorithm shown in Table 2, where the output is decoupled. We use in this and later versions a less unwieldy algorithm-like notation, writing, e.g., $\nu^{S_0^{k_0-1} \oplus S_1^{k_1-1}}_{(0,j)}$ as $\nu^{(k_0-1, k_1-1)}_{(0,j)}$. Notice that the active sets constructed in both versions are related by $\hat{S}^{k-1} \triangleq S_0^{k_0-1} \oplus S_1^{k_1-1}$, with $k = k_0 + k_1 - 1$ and $k_0 \leq k_1$. Version 2 draws on the fact that, at each step, the algorithm augments the current active set by a state that can be of the form $(1, i)$ or $(0, i)$. Sets $S_0^{k_0}$ and $S_1^{k_1}$ in the algorithm are $S_0^{k_0} = \{i_0^1, \ldots, i_0^{k_0}\}$ and $S_1^{k_1} = \{i_1^1, \ldots, i_1^{k_1}\}$, and satisfy $S_0^{k_0} \subset S_1^{k_1}$ for $1 \leq k_0 < k_1 \leq n$, consistently with (10).

3.3 Optimality of Hysteretic $\hat{\mathscr{F}}$-Policies

The following result states that LP($\hat{\mathscr{F}}$)-indexability condition (ii) above holds for the model of concern, namely that *$\hat{\mathscr{F}}$-policies*, i.e., those with active sets $\hat{S} \in \hat{\mathscr{F}}$, solve (9).

Proposition 3.3. *For every wage $\nu \in \mathbb{R}$ there exists an optimal active set $\hat{S} \in \hat{\mathscr{F}}$ for (9), i.e., if it is optimal to rest the project in state $(1, i)$ then it is optimal to rest it in $(0, i)$.*

In order to further establish the remaining conditions (i, ii) and to simplify the index algorithm we will have to draw on the work-reward analysis carried out in the next section.

4. WORK-REWARD ANALYSIS

We set out in this section to carry out a work-reward analysis of a single project with startup penalties as above, in its semi-Markov restless project reformulation, and to establish its LP-indexability.

4.1 Work and Marginal Work Measures

We start by addressing calculation of marginal work measures $w^{S_0 \oplus S_1}_{(a^-, i)}$. We show that they are closely related to their counterparts w_i^S for the underlying nonrestless project, where stationary deterministic policies are represented by their active sets $S \subseteq N$. See [8].

Lemma 4.1. *For $a^- \in \{0, 1\}$, $S_0 \oplus S_1 \in \hat{\mathscr{F}}$:*

(a) $w^{S_0 \oplus S_1}_{(1,i)} = w_i^{S_1}$, *for $i \in S_1^c$.*

(b) $w^{S_0 \oplus S_1}_{(0,i)} = \dfrac{1 - \phi_i}{1 - \beta} + w_i^{S_1}$, *for $i \in S_1^c$.*

(c) $w^{S_0 \oplus S_1}_{(1,i)} = \dfrac{1 - \beta\phi_i}{1 - \beta}\left\{w_i^{S_1} - \beta\dfrac{1 - \phi_i}{1 - \beta\phi_i}\right\}$, *for $i \in S_0$.*

Table 2: Version 2 of Algorithmic Scheme $\mathrm{AG}_{\hat{\mathscr{F}}}$.

```
ALGORITHM $\mathrm{AG}_{\hat{\mathscr{F}}}$:
Output: $\{(0, i_0^{k_0}), \nu^*_{(0,i_0^{k_0})}\}_{k_0=1}^n$, $\{(1, i_1^{k_1}), \nu^*_{(1,i_1^{k_1})}\}_{k_1=1}^n$

$S_0^0 := \emptyset$;  $S_1^0 := \emptyset$;  $k_0 := 1$;  $k_1 := 1$
while $k_0 + k_1 \leq 2n + 1$ do
    if $k_1 \leq n$ pick $j_1^{\max} \in \arg\max\{\nu_{(1,j)}^{(k_0-1,k_1-1)} : j \in N \setminus S_1^{k_1-1}\}$
    if $k_0 < k_1$ pick $j_0^{\max} \in \arg\max\{\nu_{(0,j)}^{(k_0-1,k_1-1)} : j \in S_1^{k_1-1} \setminus S_0^{k_0-1}\}$
    if $k_1 = n+1$ or $\{k_0 < k_1 \leq n$ and $\nu_{(1,j_1^{\max})}^{(k_0-1,k_1-1)} < \nu_{(0,j_0^{\max})}^{(k_0-1,k_1-1)}\}$
        $i_0^{k_0} := j_0^{\max}$;  $\nu^*_{(0,i_1^{k_0})} := \nu_{(0,i_1^{k_0})}^{(k_0-1,k_1-1)}$;  $S_0^{k_0} := S_0^{k_0-1} \cup \{i_0^{k_0}\}$;  $k_0 := k_0 + 1$
    else
        $i_1^{k_1} := j_1^{\max}$;  $\nu^*_{(1,i_1^{k_1})} := \nu_{(1,i_1^{k_1})}^{(k_0-1,k_1-1)}$;  $S_1^{k_1} := S_1^{k_1-1} \cup \{i_1^{k_1}\}$;  $k_1 := k_1 + 1$
    end { if }
end { while }
```

(d) $w_{(0,i)}^{S_0 \oplus S_1} = 1 - \phi_i + \phi_i w_i^{S_1}$, *for* $i \in S_0$.

(e) $w_{(1,i)}^{S_0 \oplus S_1} = \dfrac{w_i^{S_1}}{1-\beta}$, *for* $i \in S_1 \setminus S_0$.

(f) $w_{(0,i)}^{S_0 \oplus S_1} = \dfrac{1-\phi_i}{1-\beta} + \dfrac{\phi_i}{1-\beta} w_i^{S_1}$, *for* $i \in S_1 \setminus S_0$.

Note that, at this point in the corresponding analysis in [12] — for the no startup delay case $\phi_i \equiv 1$ — we could immediately prove positivity of marginal workloads, i.e., $w_{(a^-,i)}^{\hat{S}} > 0$, for $(a^-, i) \in \hat{N}, \hat{S} \in \hat{\mathscr{F}}$, which is a prerequisite for PCL-indexability. In the present setting, however, it is clear from Lemma 4.1(c) that $w_{(1,i)}^{S_0 \oplus S_1}$, for $i \in S_0$, can become negative if $w_i^{S_1} < \beta$ and ϕ_i is close enough to zero. This is why we cannot use here the same argument in that paper to prove indexability, and use instead the more powerful LP-indexability conditions.

4.2 Marginal Reward Measures

We continue by addressing calculation of required marginal reward measures $r_{(a^-,i)}^{S_0 \oplus S_1}$. Again, we show that they are closely related to their counterparts r_i^S for the underlying nonrestless project wit no startup costs. See [8].

Lemma 4.2. *For* $S_0 \oplus S_1 \in \hat{\mathscr{F}}$:

(a) $r_{(1,i)}^{S_0 \oplus S_1} = r_i^{S_1}$, *for* $i \in S_1^c$.

(b) $r_{(0,i)}^{S_0 \oplus S_1} = -c_i + r_i^{S_1}$, *for* $i \in S_1^c$.

(c) $r_{(1,i)}^{S_0 \oplus S_1} = \beta c_i + \dfrac{1-\beta\phi_i}{1-\beta} r_i^{S_1}$, *for* $i \in S_0$.

(d) $r_{(0,i)}^{S_0 \oplus S_1} = -(1-\beta)c_i + \phi_i r_i^{S_1}$, *for* $i \in S_0$.

(e) $r_{(1,i)}^{S_0 \oplus S_1} = \dfrac{r_i^{S_1}}{1-\beta}$, *for* $i \in S_1 \setminus S_0$.

(f) $r_{(0,i)}^{S_0 \oplus S_1} = -c_i + \phi_i \dfrac{r_i^{S_1}}{1-\beta}$, *for* $i \in S_1 \setminus S_0$.

4.3 Marginal Productivity Measures

We next address calculation of the marginal productivity measures $\nu_{(a^-,i)}^{S_0 \oplus S_1}$ in (13). Again, we show that they are closely related to their counterparts ν_i^S for the underlying nonrestless project without startup costs, given by

$$\nu_i^S \triangleq \frac{r_i^S}{w_i^S}, \quad i \in N, S \subseteq N. \tag{16}$$

The next result represents $\nu_{(a^-,i)}^{S_0 \oplus S_1}$ in terms of the ν_i^S's.

Lemma 4.3. *For* $S_0 \oplus S_1 \in \hat{\mathscr{F}}$:

(a) $\nu_{(1,i)}^{S_0 \oplus S_1} = \nu_i^{S_1}$, *for* $i \in S_1^c$.

(b) $\nu_{(0,i)}^{S_0 \oplus S_1} = \dfrac{-c_i + r_i^{S_1}}{\frac{1-\phi_i}{1-\beta} + w_i^{S_1}} = \dfrac{w_i^{S_1}}{\frac{1-\phi_i}{1-\beta} + w_i^{S_1}} \{\nu_i^{S_1} - \dfrac{c_i}{w_i^{S_1}}\}$, *for* $i \in S_1^c$.

(c) $\nu_{(1,i)}^{S_0 \oplus S_1} = \dfrac{\beta c_i + \frac{1-\beta\phi_i}{1-\beta} r_i^{S_1}}{\frac{1-\beta\phi_i}{1-\beta}\{w_i^{S_1} - \beta\frac{1-\phi_i}{1-\beta\phi_i}\}} = \dfrac{w_i^{S_1}}{w_i^{S_1} - \beta\frac{1-\phi_i}{1-\beta\phi_i}}\{\nu_i^{S_1} + \dfrac{\beta(1-\beta)}{1-\beta\phi_i}\dfrac{c_i}{w_i^{S_1}}\}$, *for* $i \in S_0$ *such that* $w_i^{S_1} \neq \beta\frac{1-\phi_i}{1-\beta\phi_i}$.

(d) $\nu_{(0,i)}^{S_0 \oplus S_1} = \dfrac{-(1-\beta)c_i + \phi_i r_i^{S_1}}{1-\phi_i+\phi_i w_i^{S_1}} = \dfrac{-(1-\beta)c_i + \phi_i w_i^{S_1} \nu_i^{S_1}}{1-\phi_i+\phi_i w_i^{S_1}}$, *for* $i \in S_0$.

(e) $\nu_{(1,i)}^{S_0 \oplus S_1} = \nu_i^{S_1}$, *for* $i \in S_1 \setminus S_0$.

(f) $\nu_{(0,i)}^{S_0 \oplus S_1} = \nu_i^{S_1} - \dfrac{(1-\beta)c_i + (1-\phi_i)\nu_i^{S_1}}{1-\phi_i+\phi_i w_i^{S_1}}$, $i \in S_1 \setminus S_0$.

4.4 LP($\hat{\mathscr{F}}$)-Indexability

We can use the above results to establish (cf. [11]) that the restless projects of concern are LP($\hat{\mathscr{F}}$)-indexable, ensuring the validity of index algorithm $\mathrm{AG}_{\hat{\mathscr{F}}}$ via Theorem 3.2.

Theorem 4.4. *Under Assumption* 3.1, *the restless reformulation of a project with switching penalties is LP*($\hat{\mathscr{F}}$)*-indexable.*

4.5 The MPI is the AT Index

The next result ensures the identity between the MPI and the AT index for the projects of concern in this paper. We find it convenient to reformulate the expressions for the AT index, given in (5)–(6) in terms of stopping times, using instead active sets $S \subseteq N$ to represent the latter — as it suffices to consider stationary deterministic policies. We formulate the continuation and switching AT indices as

$$\nu_{(1,i)}^{\mathrm{AT}} \triangleq \max_{i \in S \subseteq N} \frac{f_i^S}{g_i^S}, \tag{17}$$

and

$$\nu_{(0,i)}^{\mathrm{AT}} \triangleq \max_{i \in S \subseteq N} \frac{-c_i + \phi_i f_i^S}{\dfrac{1-\phi_i}{1-\beta} + \phi_i g_i^S}. \tag{18}$$

Recall that we denote the MPI by $\nu_{(a^-,i)}^*$.

PROPOSITION 4.5. *Under Assumption* 3.1, $\nu_{(1,i)}^* = \nu_{(1,i)}^{\mathrm{AT}}$ *and* $\nu_{(0,i)}^* = \nu_{(0,i)}^{\mathrm{AT}}$, *for* $i \in N$.

5. TWO-STAGE INDEX COMPUTATION

In this section we further simplify the index algorithm, by *decoupling* computation of the continuation and the switching index into a two-stage scheme.

5.1 First Stage: Continuation Index

We start with continuation index $\nu_{(1,i)}^*$, which is the Gittins index ν_i^* of the project. We need further quantities as input for the second-stage algorithm to be discussed later.

Table 3: Gittins-Index Algorithmic Scheme AG^1.

```
ALGORITHM AG^1:
Output: $\{i_1^{k_1}\}_{k_1=1}^n$, $\{\nu_j^* : j \in N\}$, $\{(w_j^{(k_1)}, \nu_j^{(k_1)}) : j \in S_1^{k_1}\}_{k_1=1}^n$

set $S_1^0 := \emptyset$; compute $\{(w_i^{(0)}, \nu_i^{(0)}) : i \in N\}$
for $k_1 := 1$ to $n$ do
    pick $i_1^{k_1} \in \arg\max \{\nu_i^{(k_1-1)} : i \in N \setminus S_1^{k_1-1}\}$
    $\nu_{i_1^{k_1}}^* := \nu_{i_1^{k_1}}^{(k_1-1)}$;  $S_1^{k_1} := S_1^{k_1-1} \cup \{i_1^{k_1}\}$
    compute $\{(w_i^{(k_1)}, \nu_i^{(k_1)}) : i \in N\}$
end
```

To compute such an index and extra quantities, we refer to the algorithmic scheme AG^1 in Table 3. This is a variant of the algorithm of [13], reformulated as in [8]. For actual implementations, one can use several algorithms in the latter paper, such as the *Fast-Pivoting* algorithm with extended output FP(1), performing $(4/3)n^3 + O(n^2)$ arithmetic operations; or the *Complete-Pivoting* (CP) algorithm, performing $2n^3 + O(n^2)$ operations.

5.2 Second Stage: Switching Index

We next address computation of the switching index, *after* having computed the Gittins index and required extra quantities. Consider the algorithm AG^0 in Table 4, which is fed as input the output of AG^1, and produces a sequence of states $i_0^{k_0}$ spanning N, along with corresponding index values $\nu_{(0,i_0^{k_0})}^*$, computed in a *top down* fashion, i.e., from highest to lowest. Notice that we have formulated such algorithms in a form that applies to the case where the startup delay is positive at every state j, so that $\phi_j < 1$.

The following is the main result of this paper.

THEOREM 5.1. *Algorithm* AG^0 *computes index* $\nu_{(0,i)}^*$.

We next assess the arithmetic operation count of the switching index algorithm.

PROPOSITION 5.2. *Algorithm* AG^0 *performs at most* $(5/2)n^2 + O(n)$ *operations.*

6. COMPUTATIONAL EXPERIMENTS

This section reports the results of a computational study, based on the author's MATLAB implementations of the algorithms described herein.

The first experiment investigated the runtime performance of the decoupled index computation method. We made MATLAB generate a random project instance with startup costs for each of the state-space sizes $n = 500, 1000, \ldots, 5000$. For each n, MATLAB recorded the time to compute the continuation index and required extra quantities with algorithm FP(1) in [8], the time to compute the switching MPI by the top-down (index computed from largest to smallest) and bottom-up (index computed from smallest to largest) versions of the switching-index algorithm, in which we denote by $\mathrm{AG}_{\mathrm{TD}}^0$ and $\mathrm{AG}_{\mathrm{BU}}^0$, respectively, and the time to jointly compute both indices using algorithm FPAG in [8, Sec. 6.3], which is a fast-pivoting implementation of the algorithmic scheme $\mathrm{AG}_{\mathscr{F}}$ discussed herein. Note that the switching-index algorithm AG^0 shown above is the top-down version $\mathrm{AG}_{\mathrm{TD}}^0$. This experiment was run under MATLAB R2006b 64-bit on Windows XP x64, on an HP xw9300 254 (2.8 GHz) AMD Opteron workstation.

The results are displayed in Figure 1. The left pane shows total runtimes, in hours, for computing both indices vs. n, along with curves obtained by cubic least-squares fit, which are consistent with the theoretical $O(n^3)$ complexity. Squares correspond to the $\mathrm{AG}_{\mathscr{F}}$ scheme, while circles correspond to our two-stage scheme. The results show that the two-stage method consistently achieved about a 4-fold speedup over the single-stage method.

The right pane shows runtimes, in *seconds*, for the switching index algorithm vs. n, along with curves obtained by quadratic least-squares fit, which are consistent with the theoretical $O(n^2)$ complexity. Now, squares (resp. circles) correspond to the top-down (resp. bottom-up) algorithm $\mathrm{AG}_{\mathrm{TD}}^0$ (resp. $\mathrm{AG}_{\mathrm{BU}}^0$). The change of timescale from hours to seconds demonstrates the order-of-magnitude runtime improvement achieved. Further, the bottom-up algorithm consistently outperformed the top-down one, though the difference is negligible, given the small runtimes.

The following experiments assess the average relative performance of the MPI policy in random samples of two- and three-project instances, both against the optimal policy, and against the benchmark Gittins index policy. For each instance, the optimal performance was computed by solving the LP formulation of the Bellman equations using the CPLEX LP solver, interfaced with MATLAB via TOMLAB. The MPI and benchmark policies were evaluated by solving with MATLAB the corresponding linear evaluation equations.

Table 4: Switching-Index Algorithm AG^0.

```
ALGORITHM $\mathrm{AG}^0$:
Input: $\{i_1^{k_1}\}_{k_1=1}^n$, $\{\nu_j^*: j \in N\}$, $\{(w_j^{(k_1)}, \nu_j^{(k_1)}): j \in S_1^{k_1}\}_{k_1=1}^n$
Output: $\{i_0^{k_0}\}_{k_0=1}^n, \{\nu_{(0,j)}^*: j \in N\}$

$\hat{c}_j := \frac{1-\beta}{1-\phi_j} c_j,\ j \in N;\quad z_j = \phi_j/(1-\phi_j);\quad S_0^0 := \emptyset;\quad S_1^0 := \emptyset;\quad k_0 := 0$
for $k_1 := 1$ to $n$ do
    $S_1^{k_1} := S_1^{k_1-1} \cup \{i_1^{k_1}\}$;  AUGMENT$_1$ := false
    $\nu_{(0,j)}^{(0,k_1)} := \nu_j^{(k_1-1)} - \frac{\hat{c}_j + \nu_j^{(k_1-1)}}{1 + z_j w_j^{(k_1-1)}},\ j \in S_1^{k_1} \setminus S_0^{k_0}$
    while $k_0 < k_1$ and not(AUGMENT$_1$) do
        pick $j_0^{\max} \in \arg\max\{\nu_{(0,j)}^{(0,k_1)} : j \in S_1^{k_1} \setminus S_0^{k_0}\}$
        if $k_1 = n$ or $\nu_{i_1^{k_1}}^* < \nu_{(0,j_0^{\max})}^{(0,k_1)}$
            $i_0^{k_0+1} := j_0^{\max};\ \nu_{(0,i_0^{k_0+1})}^* := \nu_{(0,i_0^{k_0+1})}^{(0,k_1)}$
            $S_0^{k_0+1} := S_0^{k_0} \cup \{i_0^{k_0+1}\};\ k_0 := k_0 + 1$
        else
            AUGMENT$_1$ := true
        end { if }
    end { while }
end { for }
```

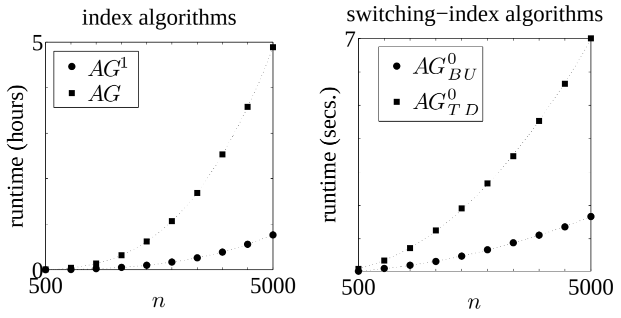

Figure 1: Exp. 1(a):Runtimes of Index Algorithms.

The second experiment assessed how the relative performance of the MPI policy on two-project instances depends on a common constant startup-delay transform's value ϕ and discount factor — there are no shutdown penalties. A sample of 100 instances (with 10-state projects) was randomly generated with MATLAB. In every instance, parameter values for each project were independently generated: transition probabilities (obtained by scaling a matrix with Uniform[0, 1] entries — dividing each row by its sum) and active rewards (Uniform[0, 1]). For each instance $k = 1, \ldots, 100$ and startup cost-discount factor combination in the range $(\phi, \beta) \in [0.5, 0.99] \times [0.5, 0.95]$ — using a 0.1 grid — the optimal objective value $\vartheta^{(k),\mathrm{opt}}$ and the objective values of the MPI ($\vartheta^{(k),\mathrm{MPI}}$) and the benchmark ($\vartheta^{(k),\mathrm{bench}}$) policies were computed, along with the corresponding relative suboptimality gap of the MPI policy $\Delta^{(k),\mathrm{MPI}} \triangleq 100(\vartheta^{(k),\mathrm{opt}} - \vartheta^{(k),\mathrm{MPI}})/|\vartheta^{(k),\mathrm{opt}}|$, and the suboptimality-gap ratio of the MPI over the benchmark policy $\rho^{(k),\mathrm{MPI,bench}} \triangleq 100(\vartheta^{(k),\mathrm{MPI}} - \vartheta^{(k),\mathrm{opt}})/(\vartheta^{(k),\mathrm{bench}} - \vartheta^{(k),\mathrm{opt}})$ — scaled as percentages. The latter were then averaged over the 100 instances for each (c, β) pair, to obtain the average values Δ^{MPI} and $\rho^{\mathrm{MPI,bench}}$.

Ojective values $\vartheta^{(k),\mathrm{opt}}$, $\vartheta^{(k),\mathrm{MPI}}$ and $\vartheta^{(k),\mathrm{bench}}$ were evaluated as follows. First, the corresponding *value functions* $\vartheta^{(k),\mathrm{opt}}_{((a_1^-,i_1),(a_2^-,i_2))}$, $\vartheta^{(k),\mathrm{MPI}}_{((a_1^-,i_1),(a_2^-,i_2))}$ and $\vartheta^{(k),\mathrm{bench}}_{((a_1^-,i_1),(a_2^-,i_2))}$ were computed as mentioned above. Then, the objective values were evaluated as

$$\vartheta^{(k),\pi} \triangleq \frac{1}{n^2} \sum_{i_1,i_2 \in N} \vartheta^{(k),\pi}_{((0,i_1),(0,i_2))}, \quad \pi \in \{\mathrm{opt}, \mathrm{MPI}, \mathrm{bench}\}, \tag{19}$$

where each project has state space $N = \{1, \ldots, n\}$, with $n = 10$. Notice that (19) corresponds to assuming that both projects are initially passive.

Figure 2 plots Δ^{MPI} vs. the ϕ — notice the inverted ϕ-axis we use throughout — for multiple discount factors β, using cubic interpolation. Such a gap starts at 0 as ϕ approaches 1 (as the optimal policy is then recovered), then increases up to a maximum value, which is less than 0.18%, and then decreases to 0 as ϕ gets smaller. Such a pattern is consistent with intuition: for small enough ϕ, both the optimal and the MPI policies initially pick a project and stay on it thereafter. Since the best project can be determined through single-project evaluations, the MPI policy identifies it.

Figure 3 shows corresponding plots for the suboptimality-gap ratio $\rho^{\mathrm{MPI,bench}}$ of the MPI over the benchmark policy. They show that the average suboptimality gap for the MPI policy is in each case less than 45% of that for the benchmark policy. Such a ratio increases with β, and takes the value 0 for ϕ small enough, as the MPI policy is then optimal.

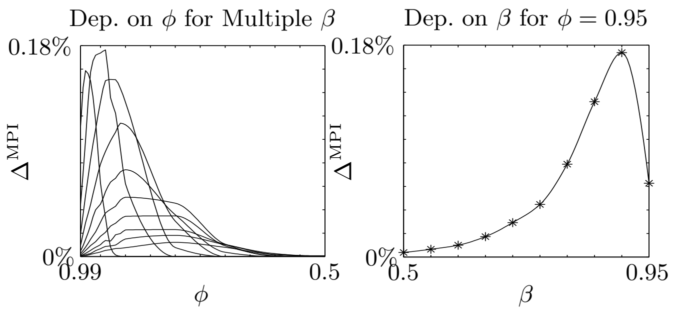

Figure 2: Exp. 2: Average Relative Suboptimality Gap of MPI Policy.

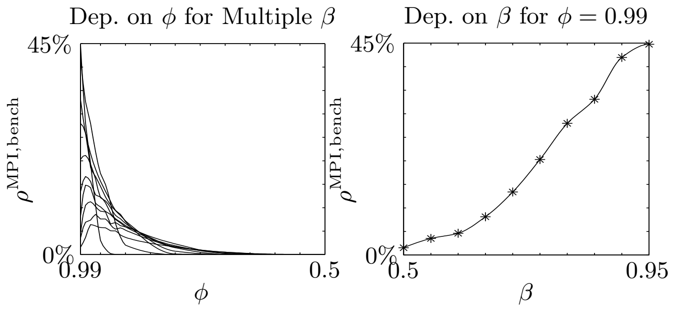

Figure 3: Exp. 2: Average Suboptimality-Gap Ratio of MPI over Benchmark Policy.

The third experiment was setup as the previous one, but considering a constant startup delay T for each project, so that $\phi = \beta^T$. Figures 4 and 5 display the results, showing that the MPI policy was optimal for $T \geq 2$, had a relative suboptimality gap of no more than 0.06%, and improved substantially on the benchmark Gittins-index policy, as the suboptimality-gap ratio remains below 2%.

The fourth experiment investigated the effect of asymmetric constant startup delay transform values, as these vary over the range $(\phi_1, \phi_2) \in [0.8, 0.99]^2$, in two-project instances with $\beta = 0.9$. The left contour plot in Figure 6 shows that the average relative suboptimality gap of the MPI policy, Δ^{MPI}, reaches a maximum value of about 0.14%, vanishing as both ϕ_1 and ϕ_2 approach unity, and as either gets small enough. The right contour plot shows that the suboptimality-gap ratio ρ^{MPI} reaches maximum values of about 50%, vanishing as either ϕ_1 or ϕ_2 gets small enough.

The fifth experiment evaluated the effect of state-dependent

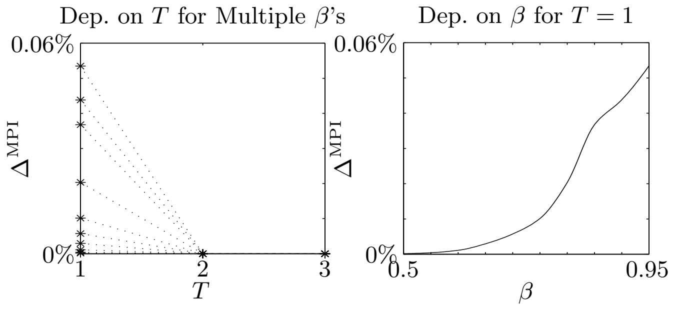

Figure 4: Exp. 3: Average Suboptimality Gap of MPI Policy.

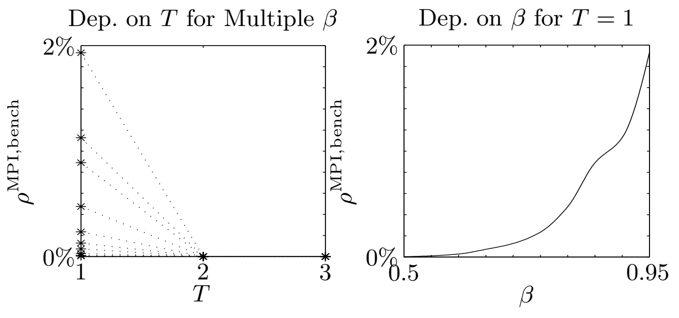

Figure 5: Exp. 3: Average Suboptimality-Gap Ratio of MPI over Benchmark Policy.

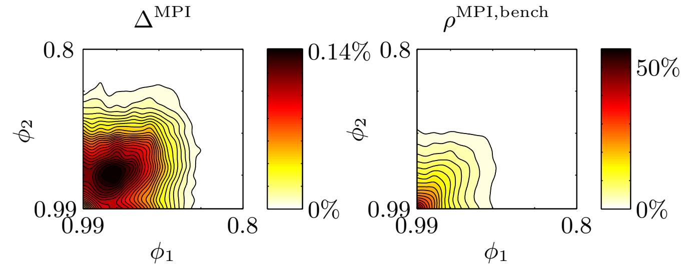

Figure 6: Exp. 4: Average Relative Performance of MPI Policy vs. (ϕ_1, ϕ_2), for $\beta = 0.9$.

startup delay parameters ϕ_i, as the discount factor varies. Uniform[0.9, 1] i.i.d. state-dependent startup costs were randomly generated for each instance. The left pane in Figure 7 plots the average relative suboptimality gap vs. the discount factor, which shows that such a gap remains below 0.14%. The right pane shows that the average suboptimality-gap ratio $\rho^{\mathrm{MPI,bench}}$ remains below 20%.

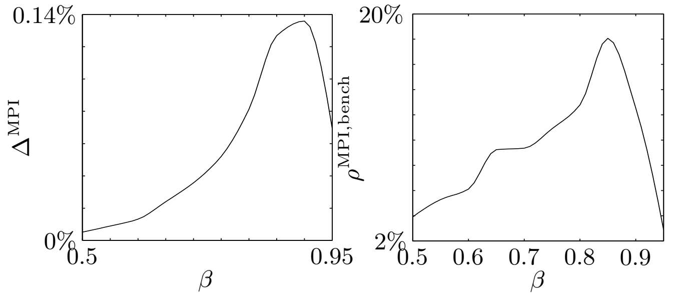

Figure 7: Exp. 5: Average Performance of MPI Policy for State-Dependent Startup Delays.

The sixth and last experiment evaluated the relative performance of the MPI policy on three-project instances as a function of a common startup delay parameter ϕ and discount factor, based on a random sample of 100 instances of three 8-state projects each. For each instance, the startup cost-discount factor combination was varied over the range $(\phi, \beta) \in [0.5, 0.99] \times [0.5, 0.95]$. The results are shown in Figures 8 and 9, which are the counterparts of experiment 2's Figures 2 and 3. Comparison of Figures 2 and 8 reveals a slight performance degradation of the MPI policy's performance in the latter, though the average gap Δ^{MPI} remains quite small, below 0.25%. Comparison of Figures 3 and 9

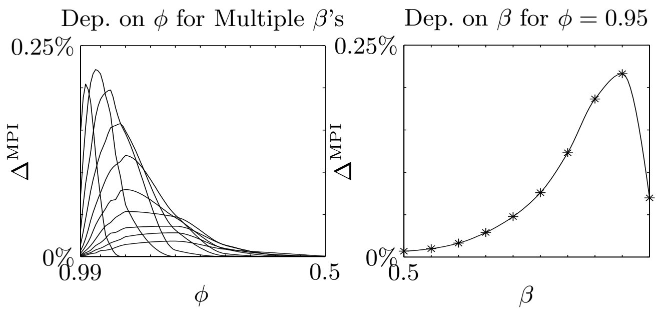

Figure 8: Exp. 6: Counterpart of Figure 2 for Three-Project Instances.

reveals similar values for the ratio $\rho^{\mathrm{MPI,bench}}$.

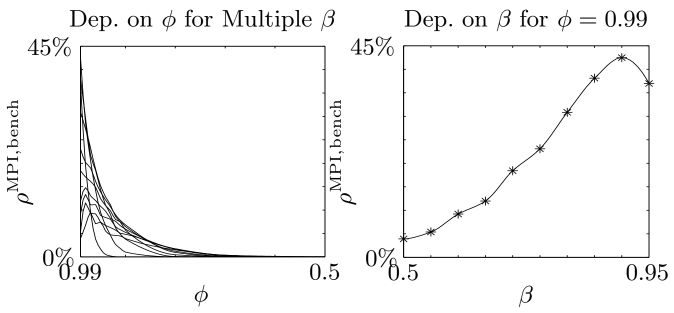

Figure 9: Exp. 6: Counterpart of Figure 3 for Three-Project Instances.

7. CONCLUDING REMARKS

We have addressed the important extension of the classic multiarmed bandit problem that incorporates both costs and delays for switching projects. The paper has demonstrated the practical applicability of the index policy based on the index introduced in [1], by introducing an efficient index algorithm and providing experimental evidence of the near optimality of such a policy. The mode of analysis has been based on deploying the powerful indexation theory for restless projects introduced in [14] and developed by the author in recent work surveyed in [10]. Thus, the AT index has been shown to be precisely the MPI of the projects of concern in their natural restless reformulation. To establish indexability and compute the index we have deployed the LP-indexability approach recently introduced in [9], which extends the earlier PCL-indexability approach in the author's earlier work. This paper demonstrates the relevance of such an extension, since the restless projects analyzed herein are LP-indexable, yet not necessarily PCL-indexable.

Acknowledgments

The author's work has been supported in part by the Spanish Ministry of Education & Science under project MTM2004-02334 and an I3 faculty endowment grant, by the European Union's Networks of Excellence EuroNGI and EuroFGI, and by the Autonomous Community of Madrid under grants UC3M-MTM-05-075 and CCG06-UC3M/ESP-0767.

8. REFERENCES

[1] M. Asawa and D. Teneketzis. Multi-armed bandits with switching penalties. *IEEE Trans. Automat. Control*, 41:328–348, 1996.

[2] J. S. Banks and R. K. Sundaram. Switching costs and the Gittins index. *Econometrica*, 62:687–694, 1994.

[3] J. C. Gittins. Bandit processes and dynamic allocation indices (with discussion). *J. Roy. Statist. Soc. Ser. B*, 41:148–177, 1979.

[4] T. Jun. A survey on the bandit problem with switching costs. *De Economist*, 152:513–541, 2004.

[5] J. Niño-Mora. Restless bandits, partial conservation laws and indexability. *Adv. Appl. Probab.*, 33:76–98, 2001.

[6] J. Niño-Mora. Dynamic allocation indices for restless projects and queueing admission control: a polyhedral approach. *Math. Program.*, 93:361–413, 2002.

[7] J. Niño-Mora. Restless bandit marginal productivity indices, diminishing returns and optimal control of make-to-order/make-to-stock $M/G/1$ queues. *Math. Oper. Res.*, 31:50–84, 2006.

[8] J. Niño-Mora. A $(2/3)n^3$ fast-pivoting algorithm for the Gittins index and optimal stopping of a Markov chain. *INFORMS J. Comput.*, 19:pp. to be assigned, 2007.

[9] J. Niño-Mora. Characterization and computation of restless bandit marginal productivity indices. In *SMCtools '07: Proceedings from the 2007 Workshop on Tools for Solving Structured Markov Chains.* ACM, New York, NY, 2007.

[10] J. Niño-Mora. Dynamic priority allocation via restless bandit marginal productivity indices (with discussion). *Top*, 15, 2007. In press.

[11] J. Niño-Mora. Two-stage index computation for bandits with switching penalties II: switching delays. Working Paper 07-42, Statistics and Econometrics Series 10, Univ. Carlos III de Madrid, Spain, 2007.

[12] J. Niño-Mora. Faster index computation and a computational study for bandits with switching costs. *INFORMS J. Comput.*, 2008. In press.

[13] P. P. Varaiya, J. C. Walrand, and C. Buyukkoc. Extensions of the multiarmed bandit problem: the discounted case. *IEEE Trans. Automat. Control*, 30:426–439, 1985.

[14] P. Whittle. Restless bandits: Activity allocation in a changing world. In J. Gani, editor, *A Celebration of Applied Probability*, volume 25A of *J. Appl. Probab.*, pages 287–298. Applied Probability Trust, Sheffield, UK, 1988.