1 Introduction

The multi-arm bandit (MAB) problem is a classic sequential decision model used to optimize the resource allocation among multiple projects (or bandits) over time. Activity on a project can earn a reward based on the current state and can affect the state transition of the project. The unselected projects will earn no rewards and their states remain unchanged. The objective of the optimization is to find a policy for sequential selection of active projects based on the current states such that expected total discounted reward is maximized over an infinite time horizon. The problem was initially formulated during World War II, but remained unsolved until the 1970s by Gittins [1, 2]. Gittins showed that each project has an index only depending on its current state, and activating the project with the largest index will be the optimal policy of the MAB problem. The index is now referred as the well-known Gittins index.

As the MAB can be solved by the index policy in a favorably simplistic form, one may be interested in tackling the problems in a more generalized class than the MAB. Whittle proposed a new class of problems [3], which generalized the MAB in the following aspects: (1) the unselected projects can continue to change their states, (2) the unselected projects can earn rewards, and (3) one can choose more than one project each time. The new class of problem is called the restless multi-arm bandits problem. In fact, the title of Whittle’s paper pointed out the insights behind the model: “Restless bandits: activity allocation in a changing world”.

Whittle proposed an index based on the Lagrangian relaxation of the restless bandit problem, which is referred as the Whittle index now. Note that Whittle proposed the index policy as a heuristic policy with no optimality guaranteed in his paper. Weber and Weiss (1990) [4] showed the asymptotical optimality of Whittle index policy for time average reward problem in certain form. In fact, the restless bandit problem has been shown to be PSPACE-hard by Papadimitrious and Tsitsiklis (1999) [5]. Practically, the Whittle index policy has shown good performance in queueing control problems (Ansell et al., 2003 [6]), machine maintenance problems (Glazebrook et al., 2005 [7]), and etc. However, it is still challenging to implement Whittle index policy. The closed form of the Whittle index is only available for specific problems with specialized structures.

In the literature, most related work focuses on the theoretical results in the conditions of indexability and closed-form solutions of Whittle index in specific applications. However, to the best of our knowledge, no study has presented and compared the empirical results of different heuristic policies in restless bandit problems. The objective of this study is to review the existing heuristic index policies and to compare their performance under different problem settings.
The remainder of the paper is organized as follows. The formal definition of restless multi-arm bandit problem and its equivalent Markov decision process model are provided in Section 2. Different index policies and the algorithms are described in detail in Section 3. Design and results of numerical experiments based on simulated cases are discussed in Section 4. Different policies are compared in the context of a real application problem in Section 5. Concluding remarks are presented in Section 6.

2 Restless multi-arm bandit (RMAB) model

The RMAB model is well defined by \(\left\{\left(S_i, \alpha_i, P_i^1, r_i^1, P_i^2, r_i^2\right)\right\}, 1 \leq i \leq N, M, \beta\), which are specified as follows:

- \(N\): The total number of arms (i.e., bandits or projects).
- \(M\): The number of arms that must be selected at each decision period. These arms are called the active arms, and rest \(N - M\) arms are called the passive arms.
- \(\beta\): The discount factor (when necessary).

For each arm \(i \in \{1, ..., N\},\)

- \(S_i\): The state space of arm \(i\).
- \(\alpha_i\): The initial state of the arm \(i\).
- \(P_i^1\): The transition probability matrix for arm \(i\) if arm \(i\) is an active arm. We use the superscript 1 and 2 to denote the active and passive arms, respectively.
- \(P_i^2\): The transition probability matrix for arm \(i\) if arm \(i\) is a passive arm.
- \(r_i^1\) (\(r_i^2\), resp.): The intermediate reward of arm \(i\) if it is active (passive, resp.).

The objective can be defined in different ways. Let random variable \(R_i^1(s_i, a_i)\) denote the reward of arm \(i\) in state \(s_i\) given the action \(a_i\) (i.e., active or passive), and \(\pi\) denote the policy:

- Total discounted reward:
  \[
  \max_{\pi} \mathbb{E}_{\pi}^\pi \left[ \sum_{t=1}^{T} \beta^{t-1} \left( R_i^1(s_1, a_1) + \cdots + R_i^N(s_N, a_N) \right) \right],
  \]
  where \(T\) can be either a finite number or \(\infty\).
- Time average reward:
  \[
  \max_{\pi} \lim_{T \to \infty} \frac{1}{T} \mathbb{E} \left[ \sum_{t=1}^{T} \beta^{t-1} \left( R_i^1(s_1, a_1) + \cdots + R_i^N(s_N, a_N) \right) \right].
  \]

In this study, we focus on the total discounted reward (finite or infinite) criterion. Of course, one can find the counterpart of each algorithm examined in this study for the time average criterion.

2.1 Equivalent Markov decision process (MDP) model

Since the RMAB is a sequential decision making problem, we can cast it into the standard form of an MDP model as follows:
- Decision epochs: $t = \mathbb{N}_+$ (for infinite horizon, or $\{1, \ldots, T\}$ for finite horizon).
- State space: $\times_{i=1}^{N} S_i$, which is a Cartesian product of state space of each individual arm. The state of the system at time $t$ becomes $(s_1(t), s_2(t), \ldots, s_N(t))$ which is an $N$-dimensional vector.
- Action space: There are in total $\binom{N}{M}$ ways of choosing $M$ out of $N$ arms. Equivalently, we let $a = (a_1, \ldots, a_N)$ denote the decision, in which $a_i = 1$ if arm $i$ is an active arm and $a_i = 2$ if a passive arm. The action space $\mathcal{A} = \{(a_1, \ldots, a_N): a_i \in \{1,2\}, \forall i; \{i: a_i = 1\} = M\}$.
- Transition matrix: As $N$ arms evolved independently, the transition probability of state $(s_1(t), s_2(t), \ldots, s_N(t))$ is essentially the product of state transition probability of each arm. Specifically,
\[ P^t((s'_1, s'_2, \ldots, s'_N)|(s_1, \ldots, s_N), a) = \prod_{i=1}^{N} p_i^{a_i}(s'_i|s_i), \]
where $p_i^1$ and $p_i^2$ are the active and passive transition matrix, respectively.
- Reward function: The sum of rewards of each arm.
\[ r((s_1, \ldots, s_N), a) = \sum_{i=1}^{N} r_i^{a_i}(s_i). \]

Then a Markov policy $\pi$ of this MDP model will be a function $\pi$: $\times_{i=1}^{N} S_i \mapsto \mathcal{A}$. The optimal Markov policy $\pi^*$ can be solved by standard algorithm for MDP: value iteration or policy iteration for infinite horizon problem, and backward induction for finite horizon problem.

Although RMAB can be formulated as an MDP, it does not mean that we can always solve the RMAB using the standard algorithms for MDP. In fact, the size of the MDP can quickly become unreasonably large, as the well-known phenomena called the curse of dimensionality. In particular, to specify the transition probability for any combination $(s, a, s')$, $s, s' \in S, a \in \mathcal{A}$, we need $(|S|^N)^2 \binom{N}{M}$ values in total. Based on the calculation results in Table 1, we can see that, an ordinary computer’s memory does not have enough space for the transition matrix even if we merely increase the state space of each arm and the number of arms by a small number.

<table>
<thead>
<tr>
<th>S</th>
<th>N</th>
<th>M</th>
<th># of states</th>
<th>Space for transition Matrix (Mb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>5</td>
<td>2</td>
<td>243</td>
<td>4.5</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>2</td>
<td>1024</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>2</td>
<td>4096</td>
<td>1,920 ~ 2Gb</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>2</td>
<td>16384</td>
<td>43,008 ~ 43Gb</td>
</tr>
</tbody>
</table>

Therefore, we can solve the RMAB as an MDP to optimality only for small instances. For large instances, we will compute an upper bound for the optimal objective value.

### 2.2 First-order relaxation: an upper bound (infinite horizon)

In the RMAB, although each arm appears to evolve according to its own dynamics, they are not completely independent of each other. This is because, at each decision period, we must select exactly
M active arms. That is, we have \( m(t) = M \) for each period \( t \), where \( m(t) \) represents the number of active arms at time period \( t \).

To have an upper bound by relaxing the constraint, we can consider a \textit{total-discounted} version of this constraint. Instead of requiring \( m(t) = M \) holds true for each period, we only need the total expected discounted (or average) number of active arms to be the same, which means

\[
E \left[ \sum_{t=1}^{\infty} \beta^{t-1} m(t) \right] = \sum_{t=1}^{\infty} \beta^{t-1} M.
\]  

Then, constraint (1) can be easily incorporated in a linear program (Bertsimas and Nino-Mora, 2000). The formulation is derived as follows.

First, we model the dynamics within each arm separately. Define \( x_i(s, a) \) as the occupancy measure of state \( s_i \) and action \( a_i \), which can be interpreted as the total expected discounted number of periods when action \( a_i \) is selected in state \( s_i \). More specifically, \( x_i(s_i, 1) \) represents the total expected discounted number of periods when arm \( i \) is selected to be active given the state \( s_i \). The occupancy measures need to satisfy

\[
Q_i = \left\{ x_i(s, a) \geq 0: x_i(s, 1) + x_i(s, 2) = \alpha(s) + \beta \sum \sum p_i^{s}(s|j)x_i(j, a), s \in S_i \right\}, \quad \forall i.
\]

Based on the interpretation of \( x_i(s_i, 1) \), constraint (1) can be formulated with occupancy measures by

\[
\sum_{i=1}^{N} \sum_{s_i \in S_i} x_i(s_i, 1) = \sum_{t=1}^{\infty} \beta^{t-1} M = \frac{M}{1 - \beta}.
\]  

Thus, the relaxation is formulated as the following linear programming (LP):

\[
UB = \max \left\{ \sum_{i=1}^{N} \sum_{s_i \in S_i} \sum_{a_i \in \{1,2\}} r_i(s_i, a_i)x_i(s_i, a_i): \text{s.t. } x_i \in Q_i, \forall i; \right\} \sum_{i=1}^{N} \sum_{s_i \in S_i} x_i(s_i, 1) = \frac{M}{1 - \beta}.
\]  

Remark: The LP has constraints in the form of \( Ax = b \). In the matrix \( A \), each column corresponds to a decision variable. The variables \( x_i(s, a) \) are arranged by the index of arm. Since each \( Q_i \) is independent of the \( x_i \)'s in any other arm, the matrix corresponding to constraints \( x_i \in Q_i, \forall i \) should appear to be a diagonal matrix consisting of \( N \) matrix blocks (as illustrated in Figure 1). The reason why the problem is not decomposable is that the additional constraint (2) links \( x_i \) variables in each block altogether, as shown in the last row at the bottom of the matrix. The left figure in Figure 1 is a schematic representation, while the right figure is obtained from the actual constraint matrix of LP based on one of our simulated test cases. Gray and dark cells represent the non-zero values in the matrix.
3 Heuristic index policies and algorithms

Generalized from the Gittins index policy, a general index policy means that we select the arms with the $M$ largest/smallest indices at each period. Then, one large decision problem for $N$ arms is reduced to $N$ small problems for each individual arm separately, which makes the computation much more tractable. For some problem instances, the index that has a good performance may reveal some intuitive insights of the problem itself. Of course, this decomposition will result in some loss in the optimality. The next question is that how to design the index with performance as close to the optimal as possible.

3.1 The Whittle index

The Whittle index was originally proposed for the time-average reward problem in Whittle (1988) \cite{3}. Glazebrook et al. (2006) \cite{8} provided the formulation for the total discounted reward problem following the same idea of the original Whittle index.

The derivation of Whittle index is based on the notion of passive subsidy $W$, which means that an arm will receive a subsidy $W$ (possible to be negative) if it is passive. That is, the passive reward $r_i(s_i, 2)$ is now replaced by $r_i(s_i, 2) + W$ for each $s_i \in S_i$. For each arm $i$, we define the subsidy-$W$ problem with following optimality equation (we suppress the arm index in the subscript for simplicity):

$$V(s, W) = \max \left\{ r^1(s) + \beta \sum_{s'} P^1(s'|s)V(s', W), r^2(s) + W + \beta \sum_{s'} P^2(s'|s)V(s', W) \right\}, \; s \in S.$$  

The value function now depends on the value of subsidy $W$. Then the Whittle index $W: S \rightarrow \mathbb{R}$ is defined by

$$W(s) = \inf \left\{ W: r^1(s) + \beta \sum_{s'} P^1(s'|s)V(s', W) \leq r^2(s) + W + \beta \sum_{s'} P^2(s'|s)V(s', W) \right\} = \inf \left\{ W: \text{passive arm is better than active arm for state } s \right\}.$$  

In other words, the Whittle index of state $s$ can be interpreted as the subsidy such that being active and passive are indistinguishable in the subsidy-$W$ problem.

Although some studies established the closed-form solution of the Whittle index in specific application problems, no generic algorithm to compute the Whittle index numerically is directly available from the
To be able to test the performance of the Whittle index policy and compare with other policies on our simulated cases, we propose the following algorithm to compute the Whittle index numerically.

**The Whittle index**

**PHASE-1 (Identify the range)**

STEP1. Initialize the value of subsidy \( W \) (e.g., \( W \leftarrow 0 \)). Specify initial step size \( a_0, \bar{a} \leftarrow 0, k \leftarrow 0 \).

STEP2. Solve the optimal value function \( V(s, W) \) of subsidy-W problem (using value iteration).

\[
k \leftarrow k + 1.
\]

STEP3. Calculate \( \Delta = [r^2(s) + W + \beta \sum_{s'} P^2(s' | s)V(s', W') - [r^1(s) + \beta \sum_{s'} P^1(s' | s)V(s', W')].
\]

STEP4. If \( \Delta \) and \( \bar{\Delta} \) have different signs, STOP and go to PHASE-2 with input \( W \) and \( \bar{W} \).

Otherwise, \( \bar{\Delta} \leftarrow \Delta \).

STEP5. \( W \leftarrow W \). If \( \Delta > 0 \), \( W \leftarrow W - 2^{k-1}a_0 \). Otherwise, \( W \leftarrow W + 2^{k-1}a_0 \). Go to STEP2.

**PHASE-2 (Binary search)**

STEP1. \( LB=\min(W, \bar{W}), UB=\max(W, \bar{W}) \).

STEP2. \( W'' \leftarrow \frac{LB + UB}{2} \). Solve the optimal value function \( V(s, W'') \) using value iteration.

STEP3. Calculate \( \Delta = [r^2(s) + W'' + \beta \sum_{s'} P^2(s' | s)V(s', W'') - [r^1(s) + \beta \sum_{s'} P^1(s' | s)V(s', W'')].
\]

STEP4. If \( \Delta > \epsilon, LB \leftarrow W'' \), go to STPE 2; else if \( \Delta < -\epsilon, LB \leftarrow W'' \), got to STEP 2; otherwise STOP and return \( W'' \).

The algorithm consists of two phases. The first is to identify the plausible range of the Whittle index \( W(s) \). If subsidy \( W \) is too large, the value-to-go of being passive will be much higher than being active in the subsidy-W problem, and thus \( \Delta \) will be positive. We should reduce the value of subsidy \( W \). On the other hand, if subsidy \( W \) is too large, \( \Delta \) will be negative, and needs to be increased. The range search will stop at the first time when \( \Delta \) reverses its sign. The value of subsidy \( W \) in the last and current iteration will be used as the end points of the range for the Whittle index. Once we identify the range, we can apply binary search to search for the \( W \) value such that \( \Delta \) is close to 0 at any desired precision.

### 3.2 The Primal-dual heuristic and the index

The Primal-Dual heuristic proposed by Bertsimas and Nino-Mora (2000) [9] is based on the solution of the first-order relaxation (3). Let \( \{x_i(s_i, a_i)\} \) denote the optimal primal solution, \( \{\lambda_i(s_i), \bar{\lambda}\} \) denote the optimal dual solution, and \( \gamma_i(s_i, a_i) \) denote the optimal reduced costs coefficients, where the optimal reduced costs can be determined by

\[
\gamma_i(s_i, 1) = \lambda_i(s_i) - \beta \sum_{j \in S_i} P^1(j | s_i) \lambda_i(j) + \lambda - \tau_i(s_i, 1), \quad \forall \ i = 1, \ldots, N.
\]

\[
\gamma_i(s_i, 2) = \lambda_i(s_i) - \beta \sum_{j \in S_i} P^2(j | s_i) \lambda_i(j) - \tau_i(s_i, 2), \quad \forall \ i = 1, \ldots, N.
\]

A heuristic approach of selecting active arms is based on following interpretation of the optimal primal solutions and the optimal reduced costs. In the first-order relaxation LP, by the interpretation of occupancy measures, for any given state \( s_i \), a positive \( x_i(s_i, 1) \) implies the arm \( i \) is selected at this state.
with positive probability, which can be regarded as a candidate active arm. The optimal reduced cost $y_i(s_i, a_i)$ is the rate of decrease in the objective value as the value of $x_i(s_i, a_i)$ increases by 1 unit, which essentially describes the “penalty” of having a high value of $x_i(s_i, a_i)$.

Next, we can select $M$ active arms according to following scheme:

<table>
<thead>
<tr>
<th>Primal-Dual heuristic</th>
</tr>
</thead>
<tbody>
<tr>
<td>For given current state $s_i$ in each arm $i$. Compute $p = \text{how many arms with positive } x_i(s_i, 1)$.</td>
</tr>
<tr>
<td>If $p=M$: choose these $M$ arms.</td>
</tr>
<tr>
<td>If $p&gt;M$: remove $(p-M)$ arms with lowest reduced cost $y_i(s_i, 2)$ from these $p$ candidate arms.</td>
</tr>
<tr>
<td>If $p&lt;M$: add $(M-p)$ more arms with lowest reduced cost $y_i(s_i, 1)$ from the $(N-p)$ non-candidate arms.</td>
</tr>
</tbody>
</table>

In fact, the choice of $(p - M)$ arms to be removed or $(M - p)$ arms to be added has an intuitive explanation. When we need to remove $(p - M)$ arms, these arms will become passive, which implies that the value of $x_i(r_i, 2)$ will increase. The increasing $x_i(r_i, 2)$ will lead to some reduction in the objective value. Although the reduction is not exactly equal to $x_i(r_i, 2)y_i(s_i, 2)$, the reduced cost $y_i(s_i, 2)$ can still reflect which arm has a higher impact (penalty) on the objective value. Intuitively, one may want to choose the $(p - M)$ arms with the lowest penalty, i.e., the lowest reduced cost $y_i(s_i, 2)$. The argument is similar for the case of adding $(M - p)$ more arms.

Bertsimas and Nino-Mora showed that above heuristic procedure has an equivalent form under the assumption that the Markov chain in each arm is connected.

<table>
<thead>
<tr>
<th>The Primal-Dual Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>For given current state $s_i$ in each arm $i$. Compute $\delta(s_i) = y_i(s_i, 1) - y_i(s_i, 2)$.</td>
</tr>
<tr>
<td>Choose the arms with the $M$ smallest $\delta(s_i)$ indices. Break ties by selecting arms with positive $x_i(s_i, 1)$ value.</td>
</tr>
</tbody>
</table>

### 3.3 Other heuristic indices

Greedy policies (or myopic policies) represent another important class of heuristic policies not only for RMAB problems but also for more general sequential decision making problems. They have strong intuition but not necessarily optimal. However, for certain special problem structure, they can have very good performance even be optimal (see examples in Liu and Zhao (2010) [10], Deo et al. (2013) [11]).

We can define the index $v_i(s_i)$ in the following different ways, and the policy will simply select the arms with the $M$ largest indices.

- Absolute greedy index: $v_i(s_i) = r_i(s_i, 1), \ \forall i$. 
- Relative greedy index: $v_i(s_i) = r_i(s_i, 1) - r_i(s_i, 2), \ \forall i$. Instead of ranking based on the active rewards, we consider the incremental benefit between passive to active arm in the current period.
- Rolling horizon ($H$-period look-ahead) index:

\[
v_i(s) = [r(s_i, 1) + \beta P^1(\cdot | s_i)V^H] - [r(s_i, 2) + \beta P^2(\cdot | s_i)V^H],
\]
in which $V^H$ represents the optimal value function for the next $H$ periods. Instead of comparing the incremental benefit in the current period, we will compare the overall incremental benefits over the next $H$ periods.

4 Numerical results: simulated cases

4.1 Experimental Design

There are 5 key questions we would like this study to address:

1. How do different policies compare under different problem structures?
2. How do different policies compare under different problem sizes?
3. Does discount factor play a significant role in algorithm performance?
4. Does time horizon play a significant role in algorithm performance?
5. Can a rolling horizon look-ahead policy improve the relative greedy policy?

To answer these questions, we will use a series of numerical simulations. In all cases, we assume the arms are not identical; that is, each arm has different rewards and transition matrices. We also assume the reward for activating an arm is greater than the reward for leaving the arm passive. This assumption is common in most modeling frameworks. For each arm in each state, we generate two uniform (0,1) random variables and set the maximum of these to be the active reward, and the minimum to be the passive reward. Except for the special cases described below, the active and passive transition matrices of each arm are uniformly sampled from the space of all transition matrices. We do this sampling via the standard procedure: For each row of the transition matrix, we generate $|S|$ exponential (1) random variables and scale each by their sum so that each entry is nonnegative and the row sums to 1. Thus the transition matrices are irreducible aperiodic and $p_{ij} > 0$ for all $i, j$. Except where mentioned, we consider the infinite horizon case with a discount factor of 0.9.

To answer question 1, we consider 4 special structures the transition matrices can take on:

a) The uniform case described above
b) The less connected (LC) case
   In this structure, each state can only transition to adjacent states. Namely, state 1 can transition to state 1 or 2, state 2 can transition to state 2 or 3, and so on.
c) Increasing failure rate (IFR) case
   For both the active and passive transition matrices,
   $$\sum_{j=k}^{|S|} P(j|i)$$
   is non-decreasing in $i$ for all arms. Together with non-increasing rewards in the state space, this condition implies higher states are more likely to deteriorate faster. This modeling framework is useful for many problems such as machine maintenance and health care: once a machine starts breaking, it is more likely to continue deteriorating to worse conditions.
d) $P^1$ is stochastically smaller than $P^2$ ($P1 SS P2$)
A form of stochastic ordering, this condition imposes

\[ \sum_{j=k}^{\mid S \mid} p^1(j|i) \leq \sum_{j=k}^{\mid S \mid} p^2(j|i), \quad 1 \leq i, k \leq \mid S \mid \]

for every arm. We also impose non-increasing rewards in the state space, so this condition says we are more likely to stay in a lower, more beneficial state under the active transition matrix than under the passive transition matrix.

To answer question 2, we will first fix N and M and increase S, then fix S and M and increase N. For question 3, we consider a range of discount factors from 0.4 to 0.99. In question 4, we consider different finite horizons ranging from 10 to 300, and compare this to the infinite horizon. For question 5, we consider 2, 5, 10, and 50 period look-ahead policies for both the uniform and less connected case.

When the problem instance is small and we can solve the dynamic programming problem to optimality, we compute this optimal solution, evaluate the optimal value exactly, and compare each algorithm’s performance to optimality. For larger instances, we compute the Lagrange upper bound, evaluate each policy via Monte-Carlo simulation, and compare the performance to the upper bound.

4.2 Results

1. How do different policies compare under different problem structures?

Figure 2 demonstrates the results of the numerical experiments for question 1. Since this is a relatively small problem size, we can solve each to optimality with dynamic programming. Each cluster of bars represents a particular algorithm, and each bar within the cluster represents the percentage gap from optimality of that algorithm on a specified problem structure.

The first observation we draw from these results is that across all problem structures, Whittle index and the primal dual method are the most effective. For all instances except for the less connected case, both algorithms perform within 0.1% of optimality. We also observe the absolute greedy policy performs unacceptably poorly in all structures. For the increasing failure rate structure, the absolute greedy policy performs only marginally better than the baseline policy of completely random choice of arms.

Our next observation is that all algorithms except for primal dual perform substantially worse on the less connected case than for any other structure. This can be interpreted as the algorithms failing to capture the special fact that the chain cannot transition between any arbitrary pair of states. Whittle index only considers the value function of adjacent states, and the greedy policy does not consider any future progression, so both perform poorly when the chain may not quickly reach an advantageous state.
A final observation is the surprising result that for $P^1$ stochastically smaller than $P^2$, the relative greedy policy is very close to optimal (0.07% from optimality). For several instances of the 40 repetitions of the given problem size, the relative greedy policy is exactly optimal. One possible intuition is since the active arm is always more likely to stay in a beneficial state, picking the arms with the largest immediate net benefit also ensures the largest future net benefit. We would like to analytically explore this phenomenon, as we believe a proof may be possible to establish optimality of the relative greedy policy under certain conditions.

2. How do different policies compare under different problem sizes?

Recall that the total state space of our process is $|S|^N$. In Figure 3 we will first fix $N$ and $M$ and increase $S$, and in Figure 4 we fix $S$ and $M$ and increase $N$. Since we are dealing with large problem instances, we can only calculate the Lagrange upper bound and compare algorithm performance to this. In Figure 3, we see every algorithm’s performance is unaffected by the choice of $S$. The computation time to compute and evaluate the policy increases, but performance is unaffected. However, Figure 4 shows a surprising result – every policy’s performance improves as $N$ increases. The explanation of this result is the fraction $M/N$. When $M/N$ is large ($N$ is small), a large fraction of the arms will be chosen, and there is only a small difference between the reward from optimal and near optimal decisions. When $M/N$ is small ($N$ is large), there is a larger gap between the rewards from optimal and suboptimal decisions, so the algorithms can gain a larger margin of improvement by selecting optimal decisions. Finally, we note no one algorithm improves at a faster relative rate than the others as $N$ increases.

3. Does the discount factor matter?

Figure 5 displays the results for experiments run to examine the effects of discount factor on the performance of the various algorithms in the infinite horizon setting. Every algorithm decreases in
performance as the discount factor increases. As the discount factor increases, the future becomes more important, and the problem becomes larger and therefore harder to solve to optimality. The performance significantly deteriorates as the discount factor approaches 1: From discount factor 0.9 to 0.99, Whittle index decreases from 0.057% to 0.10%, and primal dual decreases from 0.01% to 0.023% (a 100% increase in error in both cases, although both are still very close to optimal).

Figure 3. Uniform problem structure, fixed N and M, increasing S

Figure 4. Uniform problem structure, fixed S and M, increasing N
4. Does the time horizon matter?

Figure 6 displays the results of experiments testing the effects of increasing time horizon in the finite horizon case. Both greedy policies are unaffected by the time horizon. However, both Whittle index and primal dual improve as the time horizon increases. Both algorithms are designed for infinite horizon, so it is not surprising both would perform better as the horizon approaches infinity.

![Figure 5. Uniform problem structure, increasing discount factor](image)

![Figure 6. Uniform problem structure, increasing time horizon](image)
Figure 7. Rolling horizon discount factor 0.4, uniform and less connected structure

Figure 8. Rolling horizon discount factor 0.98, uniform and less connected structure

5. Does a rolling horizon improve the greedy algorithm?
and Figure 8 display the results of experiments testing whether a rolling horizon look-ahead policy can improve the relative greedy policy. When the discount factor is low, as seen in Table 6, a look-ahead policy provides no benefit in either the uniform or less connected case. As previously discussed, for such a low discount factor the future contribution to the total discounted reward is negligible compared to the current period, so a look-ahead policy does not contribute much value. As the discount factor increases, a look-ahead policy does gain improvement over the relative greedy policy. In Table 7, the discount factor is 0.98 and we see an improvement of the 2 step look-ahead policy in both the uniform and less connected cases. The improvement is meager in the uniform case (8% relative improvement) but significant in the less connected case (34% relative improvement). This agrees with our previous result that greedy policies perform poorly on the less connected case because they fail to consider the future development of the chain. We see no further improvement by looking more than 2 periods into the future, implying a 3 or more step look-ahead policy expends more computational effort but does not return a better solution. Finally, it should be noted that this 2 step look-ahead policy still performs worse than the Whittle index and primal dual policy for both problem structures.

4.3 Validation of algorithm implementation

All algorithms for policies construction and evaluation are implemented in MATLAB®. The first-order relaxation LP is solved by calling the CPLEX library for MATLAB. Experiments are run on the Condor Cluster of ISyE.

To validate the algorithm implementation, we have checked:

- **Policy evaluation:** For small instances, the exact value function can be evaluated based on the value iteration of equivalent MDP model. The results are very close to those based on the Monte-Carlo simulation approach.
- **MDP and Whittle index policy:** For small cases, MDP always provides the optimal objective value. Whittle index can be reduced to Gittins index when passive arms are frozen (i.e., no transition and no rewards). Gittins index is proven to be optimal when only one arm is activated each time. That is, Whittle index can be optimal when passive arms are frozen and \( M = 1 \). We test the two algorithms in simulated cases with different problem size, and observe that the objective values of two algorithms are always identical in such condition. When \( M \) becomes 2, the optimality of Whittle index does not hold any more (see Table 2).

<table>
<thead>
<tr>
<th>( S )</th>
<th>( N )</th>
<th>( M=1 )</th>
<th>( M=2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>5</td>
<td>0% (^\dagger)</td>
<td>25%</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>0%</td>
<td>25%</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0%</td>
<td>40%</td>
</tr>
</tbody>
</table>

\(^\dagger\) Percentage of cases when objective value of Whittle index policy and MDP are not equal.

The observation that MDP provides the best objective value in all of our numerical results (for small cases) also supports the correctness of our algorithm implementation.
5 Application: Capacity allocation problem

In this section, we will test and compare different algorithms and policies on a real application problem. The objective is twofold. First, we will see how real problem can be modeled as an RMAB problem, and know about the challenge of solving the RMAB of a real application size. Second, we will apply and compare the performance of different policies (with some not included in the original paper).

The clinical capacity allocation problem (Deo et al. (2013) [11]) is about how to deliver the school-based asthma care for children given the capacity constraint. A healthcare program visits a school every month, and provides treatment to children with asthma. However, the capacity of appointment/treatment during each visit is limited. Patients who have received treatment will improve their health state while those without treatment may progress to worse state. Given the limited capacity and disease dynamics, the objective is to maximize the total benefit of the community over two years. Within the framework of RMAB, this problem can be modeled as follows:

- **N arms**: number of total patients (typically ~ 50 patients).
- **M arms**: capacity, the available slots in the van during each visit (typically ~ 10-15 patients).
- **$S_i$:** State space for each patient. In the model, each patient has (1) health state at the last appointment $h \in \{controlled, improved, unchanged, worsened\} = \mathcal{H}$ and (2) time since the last appointment $n \in \{1,2,...,24\} = \mathcal{N}$ (in month). Thus, $S_i = \mathcal{H} \times \mathcal{N}$.
- **$P_i^1$:** Transition matrix if the patient receives treatment:
  \[ P_i^1 \left( \left( h', 1 \right) \mid (h, n) \right) = (QP)_{hh'}, \]
  where $Q$ is the matrix of immediate transition after treatment, and $P$ is the matrix of disease progression during one month.
- **$P_i^2$:** Transition matrix if the patient does not receive treatment:
  \[ P_i^2 \left( \left( h', n+1 \right) \mid (h, n) \right) = P_{hh'}. \]
- **$r_i^1$ and $r_i^2$:** reward is the quality-adjusted life years (QALYs) at given state $s_i$.

Remark: Not surprisingly, the problem size of the real application is far beyond being tractable. In particular, the state space is $|S_1|^N = 96^{50} \sim 1.3 \times 10^{99}$. One may suggest using value function approximation techniques to address the large state space since the states are only two-dimensional. In fact, we can use simulation-based projected equation methods to approximate value function, instead of calculating the exact value for each state. However, these approaches only resolve the challenges in the *policy evaluation* step. In the *policy improvement* step, we need to find the maximum over $\binom{50}{10}$ (in the order of $10^{10} \binom{50}{15} \sim 2 \times 10^{12}$) different actions, which is still prohibitively large for computation. One possible approach is to approximate the policy function using a parametric model proposed in Powell (2010) [12]. Current states and parameter $\theta$ are the inputs to the parametric model. Then, we need to tune up the best parameter $\theta$ using stochastic optimization algorithms (e.g., random search, stochastic approximation), instead of taking the maximum over an intractably large action space.
5.1 Numerical results

In addition to the policies we introduced in Section 3, there are several more problem-specific policies to be evaluated.

- **Fixed-duration policy.** Physicians will recommend the next follow-up time based on patients’ health state (3 months for controlled state, and 1 month for uncontrolled states). The patients who are due back in prior periods have the highest priority, and followed by those who are due back in the current period.

- **H-N priority policy.** It first prioritizes patients based on their health state observed in the last appointment, and then breaks ties based on the time since the last appointment.

- **N-H priority policy.** Similar to H-N priority policy but with a reversed order of prioritization.

- **No-schedule policy.** No patients are scheduled and all follows the natural progression process.

We use the performance measurement proposed by Deo et al., the improvement of each policy $\delta$ is defined as

$$\delta(\pi) = \frac{V_\pi - V_{\text{fixed-duration}}}{V_{\text{fixed-duration}} - V_{\text{no-schedule}}}.$$

The results are shown in Figure 9.

![Figure 9. Policy comparisons in the capacity allocation problem.](image)

**Observation 1.** Relative greedy policy performs well and remains stable in all cases. The rolling-horizon policy performs similar to relative greedy policy, with only marginal improvement to the relative greedy policy. The paper has shown that relative greedy policy can be optimal under certain conditions, while the parameter estimates for the numerical study do not meet all of them, with few violations. It is still reasonable to expect and see that relative greedy policy performs well overall.
Observation 2. The Whittle index policy performs well and very close to the relative greedy policy. However, it takes 1.5-3 hours to compute all Whittle indices for each state of each arm (96 × 50 = 4800 indices in total). As a generic heuristic index policy independent of problem structure, the Whittle index policy has demonstrated its robustness of good performance in all tests that we have done in this study.

Observation 3. Improvement reduces as the capacity becomes less restrictive. This finding is in line with those based on the simulated cases in Section 4.2. As M/N becomes larger, the relative difference between the best selection and the second best selection can be less significant.

Observation 4. As capacity increases, the performance of the primal-dual index policy degrades significantly. We speculate that it may be because that the first-order relaxation LP used to compute the primal-dual index is essentially an infinite horizon problem. We use a large discount factor 0.99 which makes the model takes more care of future rewards. However, the capacity allocation problem is only evaluated for 24 periods (2 years). This disconnect between the settings of the policy and the problem may make the primal-dual index policy behave poorly.

6 Conclusion
From our numerical experiments, we have demonstrated Whittle index and the primal dual policy both work efficiently in all cases. They provide solutions within 1% of optimality when we know the exact solution, and within 5% of the Lagrange upper bound when we cannot calculate the optimal solution. The computation time of the policy is only seconds for even the largest problems we tested, although evaluating the solution may still be costly. These algorithms remain the best choice for all problem sizes and all structures, although the relative greedy policy is close to optimal when $P_1$ is stochastically smaller than $P_2$. Both Whittle index and the primal dual policy perform better when the time horizon is large or infinite. All algorithms perform more poorly when the discount factor is large, due to the added complexity of the optimization.

In the real application, we have seen the relative greedy policy perform very close to optimal, as dictated by the special structure of the problem. The Whittle index was very expensive to compute, and performed less well than the greedy policy. Consistent with our results from the numerical experiments, we found the primal dual policy did not perform well because it is evaluated only on a relatively small time horizon.

Reference


