Efficient Computation of Whittle Index for Partially Observable Restless Bandits

Qizhen Jia
School of Mathematics and Physics
Xi'an Jiaotong–Liverpool University
Suzhou, China
qizhen.Jia21@student.xjtlu.edu.cn

Keqin Liu*
School of Mathematics and Physics
Xi'an Jiaotong–Liverpool University
Suzhou, China
keqin.liu@xjtlu.edu.cn

***Abstract*—Restless multi-armed bandit (RMAB) models a sequential allocation problem when arms evolve even while passive (not selected) and observations may be partial. Finite-state RMABs with perfect observation are computationally intractable (PSPACE-hard), which motivates scalable index-based decision rules. *Gittins* indices solve the classical bandit, whereas *Whittle*'s relaxation extends indexation to the restless setting and has enabled broad applications in communications, queueing, and public health. However, in partially observable models the *cost of computing* Whittle indices can dominate runtime due to (i) belief-space blowup and (ii) repeated linear solvers within partial conservation law (PCL)/adaptive-greedy (AG) workflows. In this paper, we present an implementation-oriented optimization pipeline that leaves interfaces and index logic unchanged yet substantially accelerates end-to-end evaluation: (i) hash-guided belief deduplication with ε-radius merging, (ii) shared linear factorization with multi-RHS reuse, and (iii) batch vectorization with memoized intermediates. We observe large speedups while keeping Whittle–myopic runtime gaps within a small, controlled band. Our study complements modeling and theory by focusing on numerical and systems aspects that make index computation efficient for larger instances and tighter time budgets. We position our work alongside classic results on complexity and index policies [1]–[4] and recent advances in RMAB algorithms and applications [5]–[12].**

***Index Terms*—Restless bandit, Whittle index, partial observability, partial conservation law, adaptive-greedy algorithm, numerical optimization, hashing, vectorization, linear solvers.**

I. Introduction

Restless multi-armed bandits (RMABs) model sequential allocation when each arm follows Markovian dynamics and continues to evolve even when *passive (not selected)*. Closely related stochastic control problems—e.g., queueing network control—are PSPACE-hard, and even finite-state restless bandits with perfect observation are PSPACE-hard. This motivates the use of index-based heuristics instead of exact dynamic programming at scale [1], [2].

This research was supported by Leadership Talent Program (Science and Education) of SIP (Grant No. KJQ2024202).

*Corresponding author.

Index policies address this complexity. In classical multi-armed bandits with perfect observation, the Gittins index gives an optimal policy [3]; Whittle extended indexation to restless dynamics via a *subsidy for passivity* and a relaxation that decouples arms [4]. For finite-state RMABs, the *partial conservation law* (PCL) framework re-expresses indexability and index computation through linear "work" and "reward" equations on a single-arm Markov model, and the associated *adaptive-greedy* (AG) procedure uses these equations to test indexability and compute Whittle indices [2], [13]. Such index rules and near-myopic structures support applications in communications, queueing, and public-health intervention planning [5]–[8].

In partially observed RMABs, the controller does not see the underlying Markov state but maintains a *belief state* (a probability vector over latent states). Under general observation models, the set of reachable beliefs proliferates as we simulate actions and observations to build a finite embedded belief graph for a single arm, whose nodes are belief vectors. On top of this belief graph, the AG procedure repeatedly solves closely related linear systems to evaluate the performance quantities required by PCL. In straightforward implementations, this yields quadratic (or worse) scans for belief deduplication and repeated dense solves with near-identical coefficients, making even moderate state sizes ($M \geq 6$) computationally onerous despite simple per-arm logic [9]–[11]. The *policy form* remains attractive, but the *numerics of index computation* become the bottleneck.

In this paper, we propose a drop-in numerical pipeline for PCL/AG-based index computation: we expand reachable beliefs into a finite belief graph, build passive and active belief-transition kernels on this graph, and solve the linear performance equations that yield the discounted work and reward functions used in PCL and their increments. Our implementation accelerates belief expansion with a hash-guided ε-merge, reuses a single factorization of $(\mathbf{I} - C^{\mathcal{P}})$ for multiple right-hand sides within each AG step, and applies batch vectorization with

memoized intermediates to the resulting linear systems and index updates. These changes leave the external interface and index logic unchanged while materially reducing wall-clock time and preserving the discounting scheme, algorithm semantics, and outputs (indices and indexability flags).

II. Problem Setup and Background

We study an N-arm restless bandit. Each arm admits M latent states $\mathcal{M} = \{1, \dots, M\}$ with passive and active dynamics, and a belief state $\boldsymbol{\omega} \in \Delta^{M-1}$ (a probability vector over $\mathcal{M}$). When *passive* (a=0), the belief evolves via a Markov kernel $P \in \mathbb{R}^{M \times M}$. When *active* ($a$=1), an observation $o \in \{1, \dots, M\}$ (and possibly an instantaneous reward symbol r) is generated according to the observation/error matrix $E \in \mathbb{R}^{M \times M}$ and the reward table $R \in \mathbb{R}^{M \times M}$. We assume rows are *row-stochastic* (sum to 1) for E and, when introduced below, for the induced kernels P^0, P^1. Let $\beta \in (0, 1)$ denote the discount factor for future rewards.

Belief updates (observation then transition): We adopt the timing convention "observe first, then transition," which is equivalent (up to an index shift) to the standard predict–update Bayes filter in POMDPs [14]. For a=0 (passive),

$$\boldsymbol{\omega}^+ = \boldsymbol{\omega} P. \tag{1}$$

For a=1 (active), given (o, r) generated from the *current* latent state,

$$\mathrm{den}(o, r; \boldsymbol{\omega}) = \sum_{i=1}^{M} \omega_i \, E_{i,o} \, \mathbf{1}\{R_{i,o} = r\}, \tag{2}$$

$$(\boldsymbol{\omega}^+)_j = \frac{\sum_{i=1}^{M} \omega_i \, E_{i,o} \, \mathbf{1}\{R_{i,o} = r\} \, P_{i,j}}{\mathrm{den}(o, r; \boldsymbol{\omega})}. \tag{3}$$

Zero-probability branches are discarded: we enumerate (o, r) only when $\mathrm{den} > 0$.

Single-arm relaxation and AG algorithm setup: Following Whittle's relaxation, we introduce a passivity subsidy and work with a *single* arm evolving on a finite embedded belief graph. Let

$$\Omega = \{\boldsymbol{\omega}_1, \dots, \boldsymbol{\omega}_S\}, \qquad \mathcal{P} \subseteq \{1, \dots, S\}, \qquad \mathcal{A} = \mathcal{P}^c,$$

denote, respectively, the finite belief graph, the *passive* set, and its complement, the *active* set. Rows with $i \in \mathcal{P}$ use the passive kernel, while rows with $i \in \mathcal{A}$ use the active kernel. Define the controlled kernel

$$C^{\mathcal{P}}(i, \cdot) = \begin{cases} \beta \, P^0(i, \cdot), & i \in \mathcal{P}, \\ \beta \, P^1(i, \cdot), & i \in \mathcal{A}, \end{cases}$$

where P^0, P^1 are the passive/active belief-transition kernels on Ω (constructed below), and let $r \in \mathbb{R}^S$ collect the expected immediate rewards upon activation at each node.

Performance systems and AG marginal increments: Let $(\mathbf{1}_{\mathcal{A}})_i = \mathbf{1}\{i \in \mathcal{A}\}$ and $(r_{\mathcal{A}})_i = r_i \, \mathbf{1}\{i \in \mathcal{A}\}$. The performance vectors $T, W \in \mathbb{R}^S$ are defined as the unique solutions of

$$(\mathbf{I} - C^{\mathcal{P}}) \, T = \mathbf{1}_{\mathcal{A}}, \qquad (\mathbf{I} - C^{\mathcal{P}}) \, W = r_{\mathcal{A}}. \tag{4}$$

Here T gives the discounted *expected number of active visits* and W the discounted *expected cumulative reward* under the given passive set $\mathcal{P}$. Since $C^{\mathcal{P}} = \beta M$ with row-stochastic M and $\beta \in (0, 1)$, we have $(\mathbf{I} - C^{\mathcal{P}})^{-1} = \sum_{t \geq 0} (C^{\mathcal{P}})^t \geq 0$, hence $T \geq 0$ componentwise (inverse positivity) [15], [16]. The (PCL/AG) marginal increments—*action-usage/work* and *reward/gain*—are

$$A = \mathbf{1}_{\mathcal{A}} + \beta \, (P^1 - P^0) \, T, \qquad U = r_{\mathcal{A}} + \beta \, (P^1 - P^0) \, W, \tag{5}$$

and the index ratio U_i / A_i is formed only for $i \in \mathcal{A}$ (roman A denotes the increment vector, not the active set $\mathcal{A}$). Under PCL-indexability, $A_i > 0$ wherever ratios are formed; see [13], [17]. Once P^0 and P^1 are specified on Ω, Whittle indices follow directly from these linear systems.

Finite embedding and kernels: From an initial belief $\boldsymbol{\omega}_0$, we expand beliefs to depth T=6 and apply the quantize-and-merge procedure of Sec. III to obtain the finite node set Ω. The passive kernel P^0 maps the one-step passive update $\boldsymbol{\omega}_i P$ to its nearest neighbor in Ω. For the active kernel, we define

$$\begin{aligned} p_i(o, r) &:= \sum_{m=1}^{M} \omega_{i,m} \, E_{m,o} \, \mathbf{1}\{R_{m,o} = r\}, \\ \tau_i(o, r) &:= \mathrm{BeliefUpdate}(\boldsymbol{\omega}_i, o, r), \\ P^1(i, j) &:= \sum_{(o,r):\, \Pi_\varepsilon(\tau_i(o,r)) = \boldsymbol{\omega}_j} p_i(o, r), \end{aligned}$$

where Π_ε is the quantize-and-merge projection used in Sec. III. We then apply a single *row normalization for numerical hygiene* (theoretical row sums are 1 in exact arithmetic) [18]. This finite embedding via quantize-and-merge is consistent with point-based POMDP practice (e.g., PBVI) [19] and provides the P^0, P^1 needed by the PCL/AG equations above.

Terminology: A *belief-expansion iteration* means one breadth-first *layer* on the embedded belief graph: from the current frontier we apply the passive and active update operators once, collect all successor beliefs, and then perform the ε-merge; after T such iterations we obtain the T-step approximate state space on which we run the PCL/AG procedure [10], [19]. This notion of "iteration" underlies the belief-expansion iterations-per-second metric in Sec. IV.

III. Numerical Optimization Details

Our pipeline preserves the external API and the underlying PCL/AG logic, while improving throughput via three drop-in numerical modifications.

(1) Hash-guided ε-merge on beliefs: We perform a breadth-first expansion of (o, r) and passive branches for T=6 *belief-expansion iterations* (layers; see Sec. II, Terminology) and deduplicate beliefs on a quantization grid (step δ=$\varepsilon/2$, default ε=5×10^{-4}). Each candidate belief is first mapped to a hash bucket via its quantized key; it merges into that bucket if an ℓ_2 representative lies within radius ε, and otherwise we append a new node. Branches with zero mass are not generated. Under standard hashing assumptions, this yields *amortized constant-time* expected lookups per insertion [20], in contrast to quadratic all-pairs distance checks.

(2) Kernel assembly and stochasticity checks: We assemble P^0, P^1 once per instance, based on the finite embedding described in Sec. II. For P^1, we aggregate the probabilities of all (o, r) pairs that land on each neighbor and then normalize *once* per row; for P^0, we project $\boldsymbol{\omega}P$ to its nearest neighbor in Ω. To respect the modeling assumption that P^0, P^1, E have stochastic rows, we enforce row-stochasticity at the numerical level: machine-scale negatives ($< 10^{-15}$) from roundoff are clipped and rows are renormalized on private copies of the data structures [18]. This keeps the implemented kernels consistent with the ideal Markov structure.

(3) Stable linear solves with iterative refinement: At each AG step, the coefficient matrix $(\mathbf{I} - C^{\mathcal{P}})$ is fixed while the right-hand side changes across the two performance systems in (4); we therefore reuse a single factorization *within that step* (across steps, $\mathcal{P}$ changes, so the factorization is rebuilt). As noted in Sec. II, $(\mathbf{I} - C^{\mathcal{P}})$ is a nonsingular M-matrix for $\beta \in (0, 1)$ with row-stochastic M; here we exploit this property to reuse a single factorization per step and to apply two-step iterative refinement [18]. We stop refinement when $\|b - (\mathbf{I} - C^{\mathcal{P}})\hat{x}\|_\infty < 10^{-12}$. To avoid propagating tiny negative artifacts of floating-point arithmetic, we clamp machine-scale negatives in T, W to 0, and clamp U_i only when $|U_i| \le 10^{-14}$, while logging the number of clamps. These safeguards do not change the intended PCL/AG semantics, but improve numerical robustness.

Formal stability statement (projection/merge): We formalize the effect of ε-merge via a standard resolvent bound and record the statements for reference; proofs follow familiar perturbation arguments for linear systems and are omitted for brevity.

Theorem 3.1 (Stability under projection): Fix a passive set $\mathcal{P}$ and let $C = \beta M$ and $\widehat{C} = \beta\widehat{M}$ be the exact and projected controlled kernels, where $M, \widehat{M}$ are row-stochastic and $\beta \in (0, 1)$. Assume the projection $\Pi_\varepsilon : \Delta^{S-1} \to \Delta^{S-1}$ used to form $\widehat{M}$ satisfies, for some $L > 0$,

$$\sup_{v \in \Delta^{S-1}} \|\Pi_\varepsilon v - v\|_1 \;\le\; L\,\varepsilon. \tag{A1}$$

Let $x^\star, \widehat{x}^\star$ solve $(\mathbf{I} - C)x = b$ and $(\mathbf{I} - \widehat{C})\widehat{x} = b$ for the same right-hand side b (e.g., $b = \mathbf{1}_{\mathcal{A}}$ or $b = r_{\mathcal{A}}$). Then

$$\|x^\star - \widehat{x}^\star\|_\infty \;\le\; \frac{\beta\, L\, \varepsilon}{(1-\beta)^2}\, \|b\|_\infty. \tag{6}$$

Remark 1 (From ℓ_2 merge radius to ℓ_1 control): If the merge radius is specified in ℓ_2 on Δ^{S-1}, then by norm equivalence $\|x\|_1 \le \sqrt{S}\,\|x\|_2$ and (A1) holds with $L \leftarrow \sqrt{S}\,L_2$, where L_2 is the induced ℓ_2 Lipschitz constant.

Corollary 3.2 (Decision stability away from ties): Let (U, A) and $(\widehat{U}, \widehat{A})$ be the exact and projected increments built from (4). Suppose $A_i, \widehat{A}_i \ge a_{\min} > 0$ for all $i \in \mathcal{A}$. Then there exists an explicit constant

$$K(a_{\min}, \|r\|_\infty, \beta) = \frac{2\|r\|_\infty}{a_{\min}} + \frac{2\|r\|_\infty}{a_{\min}^2}\Big(1 + \frac{2\beta}{1-\beta}\Big)$$

such that, for every $i \in \mathcal{A}$,

$$\left|\frac{U_i}{A_i} - \frac{\widehat{U}_i}{\widehat{A}_i}\right| \;\le\; K(a_{\min}, \|r\|_\infty, \beta)\; \frac{\beta\, L\, \varepsilon}{(1-\beta)^2}.$$

Consequently, if the gap between the top two indices exceeds $2K(a_{\min}, \|r\|_\infty, \beta)\,\beta L\varepsilon/(1-\beta)^2$, the maximizer is invariant (i.e., stable away from ties). We use this corollary qualitatively: it explains why, in practice, moderate ε does not change the top-ranked arm when index gaps are not vanishingly small.

Complexity and memory: Belief expansion visits $|\Omega|$ nodes and, with hash-guided ε-merge, has *amortized* $O(|\Omega|)$ work under standard hashing assumptions [20]. Kernels are built once in $O(|\Omega|^2)$ arithmetic, and each AG step performs a single factorization of $\mathbf{I} - C^{\mathcal{P}}$ plus a small, fixed number of triangular solves for multiple right-hand sides in that step. Memory usage is dominated by storing the belief set Ω, the dense kernels P^0, P^1, and the factorization of $\mathbf{I} - C^{\mathcal{P}}$, which are all $O(|\Omega|^2)$ in the generic dense case.

IV. Experiments

We address two questions. First, for $M \in \{6, 7, 8\}$ and $N \in \{50, 100\}$, does the optimized pipeline still allow the Whittle policy to outperform a myopic policy (Q1)? Second, during Whittle-index computation, how much faster is the optimized pipeline than the unoptimized reference, as measured by *belief-expansion iterations per second (it/s)* and wall-clock time (Q2)?

A. Setup and data generation

We set $\beta = 0.9999$ for all baselines and approximate the state space by capping *belief expansion* at six iterations with tolerance 10^{-3}. All arms are i.i.d. and *start from each arm's stationary prior* ω_0 under the passive transition P; when P is not ergodic, we restrict attention to the relevant communicating class. Each policy is evaluated with 1,000 Monte Carlo simulations per instance to estimate average performance.

Hardware: All timings were obtained on an AMD Ryzen 9 9950X with 48 GB RAM.

Baselines: **Legacy** is the unoptimized reference implementation. **Optimized** is our numerically improved pipeline (Sec. III) with ε-merge (5×10^{-4}), iterative-refinement tolerance 10^{-12}, and protected divisions. Interfaces, discounting, and PCL/AG logic are otherwise identical.

Metrics and fairness controls: We report the **policy performance gain** $100 \cdot \frac{J_{\text{Whittle}} - J_{\text{myopic}}}{|J_{\text{myopic}}|}$, and the **throughput** in *belief-expansion iterations per second (it/s)*, measured inside the belief-expansion routine; we also record per-instance wall-clock time. All runs use the same $(P, E, R, \omega_0, \beta)$, the same six belief-expansion iterations with tolerance 10^{-3}, identical stopping rules and timing methodology, and a 30-minute per-instance timeout.

B. Q1: Effectiveness at $M=6,7,8$ with $N=50,100$

Table II reports Whittle-vs.-myopic percentage gains across all settings. The gaps are consistently positive, showing that our numerical streamlining *preserves decision quality*. For illustration, Figs. 1–3 plot reward trajectories only for N=100 arms; the curves for N=50 arms show the same pattern and are omitted for space.

C. Q2: Throughput (iterations per second) and wall-clock

We use two complementary speed metrics: (i) *belief-expansion* iterations per second (it/s), measured inside the belief-expansion routine, and (ii) end-to-end wall-clock for a complete index computation (state-space approximation with the six-iteration cap and 10^{-3} tolerance, kernel construction, and all linear solves).

Throughput (belief-expansion it/s): On sizes that complete under the 30-minute cap, the optimized pipeline achieves **15.8-fold**, **100.7-fold**, **540.8-fold**, and **2806.3-fold** speedups over the legacy code for the 3×3, 4×4, 5×5, and 6×6 sizes, respectively; see Table I. For 7×7 and 8×8, the legacy code times out, whereas the optimized code reaches 31.604 and 13.416 it/s, respectively. To avoid bias, we treat timeouts as *right-censored* and do not form speedup ratios for those sizes.

Wall-clock: End-to-end wall-clock shows the same pattern: under a 30-minute cap, optimized runs finish across all reported settings, while the legacy 7×7 case times out (cf. Table I). Profiling the legacy implementation on 7×7 *without* a time cap on the same hardware yields a throughput of about **0.0021 it/s**, which is consistent with these timeouts.

TABLE I: Throughput (iterations per second, it/s) by matrix size.

Size ($M\times M$)	Optimized (it/s)	Legacy (it/s)	Notes
3×3	1302.256	82.210	
4×4	437.194	4.340	
5×5	175.038	0.3236	
6×6	73.668	0.02625	
7×7	31.604	–	legacy timed out
8×8	13.416	–	legacy timed out

TABLE II: Whittle vs. myopic under the common setup. Entries show final average reward (higher is better) and relative improvement (%).

	N=50 arms			N=100 arms		
$M\times M$	Myopic	Whittle	% gain	Myopic	Whittle	% gain
6×6	2.3371	2.5768	10.26	2.3791	2.5906	8.89
7×7	2.3110	2.4732	7.02	2.3525	2.5053	6.50
8×8	2.3044	2.4384	5.82	2.2586	2.4088	6.65

V. Conclusion and Outlook

We studied how to speed up Whittle-index computation for restless multi-armed bandits with general observation models from a *numerical* viewpoint. Without changing the problem setup, discounting scheme, or PCL/AG algorithmic logic, we proposed a practical pipeline that combines hash-guided ε-merge during belief expansion, reuse of a single factorization of $(\mathbf{I} - C^{\mathcal{P}})$ for multiple right-hand sides within each AG step, and numerically robust linear algebra for the PCL/AG performance equations. These changes reduce overhead while keeping the public interface and the definition of the Whittle index intact.

Empirically, across state sizes $M \in \{6, 7, 8\}$ and arm counts $N \in \{50, 100\}$, the optimized implementation preserves decision quality in our testbed: the Whittle policy consistently beats the myopic baseline (Table II). It also shortens end-to-end runtime: with belief expansion fixed at T=6, the index-computation pipeline finishes reliably where the legacy code often times out and delivers 1–3 orders of magnitude higher belief-expansion throughput (Table I). For N=100 arms, reward trajectories remain stable as M increases (Figs. 1–3), and the curves for N=50 arms show the same pattern (omitted for space), indicating that the numerical changes do not degrade policy performance in these scenarios.

The main limitations are numerical rather than conceptual. The ε-merge introduces a bias–variance trade-off: aggressive merging risks hiding indexability, while conservative merging gives up speed. Here we fix a small ε that is empirically safe for the model families studied, but an adaptive choice would be preferable. Iterative refinement improves robustness of the linear solves, yet ill-conditioned instances can still inflate runtime, suggesting value in lightweight preconditioning and structure-aware factorizations. Our experiments use synthetic RMAB instances; evaluating the pipeline on domain-specific real-world workloads is left for future

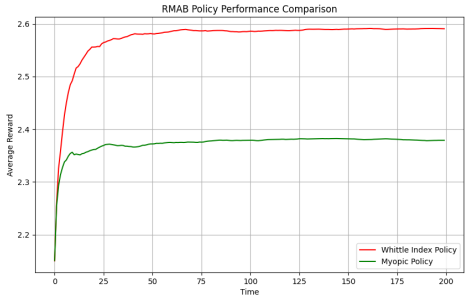

Fig. 1: Simulation for M=6, N=100 arms.

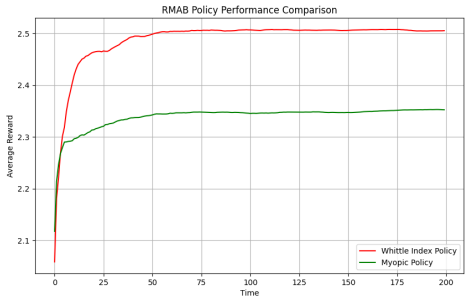

Fig. 2: Simulation for M=7, N=100 arms.

work.

Future work includes adaptive belief compression (instance-aware ε with explicit measures of index-order stability), more advanced solver engineering (preconditioners, sparse or block-structured factorizations, warm starts across AG steps), and exploiting parallelism and hardware acceleration (parallel (o, r) branches and arms, GPU-backed linear algebra) under the same API. We also aim to validate the approach on additional structured domains such as communications and age-of-information (AoI) systems, queueing, and public health scheduling. Overall, the numerical pipeline acts as a drop-in replacement for legacy implementations: it preserves policy quality and indexability checks while making previously impractical cases ($M \geq 7$ in our experiments) feasible under tight time budgets, and the underlying ideas should transfer to a broad range of RMAB solvers beyond the specific setting studied here.

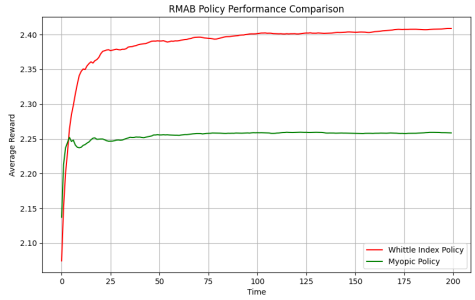

Fig. 3: Simulation for M=8, N=100 arms.

REFERENCES

[1] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of optimal queuing network control," *Mathematics of Operations Research*, vol. 24, no. 2, pp. 293–305, 1999.

[2] D. Bertsimas and J. Niño-Mora, "Restless bandits, linear programming relaxations, and a primal-dual index heuristic," *Operations Research*, vol. 48, no. 1, pp. 80–90, 2000.

[3] J. C. Gittins, "Bandit processes and dynamic allocation indices," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 41, no. 2, pp. 148–177, 1979.

[4] P. Whittle, "Restless bandits: Activity allocation in a changing world," *Journal of Applied Probability*, vol. 25A, pp. 287–298, 1988, special Vol. 25A.

[5] S. H. A. Ahmad, M. Liu, T. Javidi, Q. Zhao, and B. Krishnamachari, "Optimality of myopic sensing in multichannel opportunistic access," *IEEE Transactions on Information Theory*, vol. 55, no. 9, pp. 4040–4050, 2009.

[6] S. Aalto, "Whittle index approach to the multi-class queueing systems with convex holding costs and ihr service times," *Mathematical Methods of Operations Research*, vol. 100, no. 3, pp. 603–634, 2024.

[7] A. Mate, J. A. Killian, H. Xu, A. Perrault, and M. Tambe, "Collapsing bandits and their application to public health interventions," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS '20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[8] K. Liu and Q. Zhao, "Distributed learning in multi-armed bandit with multiple players," *IEEE Transactions on Signal Processing*, vol. 58, no. 11, pp. 5667–5681, 2010, preprint: arXiv:0910.2065.

[9] K. Liu, "Relaxed indexability and index policy for partially observable restless bandits," *Management Science*, 2025.

[10] K. Liu, Q. Jia, and C. Zhang, "PCL-indexability and whittle index for restless bandits with general observation models," arXiv:2307.03034, 2023.

[11] K. Liu, R. Weber, and C. Zhang, "Low-complexity algorithm for restless bandits with imperfect observations," *Mathematical Methods of Operations Research*, vol. 100, no. 2, pp. 467–508, 2024.

[12] K. Liu, Y. Zhang, and Z. Ding, "Efficient algorithm design of dynamic spectrum access by whittle index," arXiv:2501.00236, 2025.

[13] J. Niño-Mora, "Restless bandits, partial conservation laws and indexability," *Advances in Applied Probability*, vol. 33, no. 1, pp. 76–98, 2001.

[14] S. Särkkä, *Bayesian Filtering and Smoothing*, ser. Institute of Mathematical Statistics Textbooks. Cambridge, UK: Cambridge University Press, 2013, no. 3.

[15] A. Berman and R. J. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*. Philadelphia, PA: SIAM, 1994, sIAM Classics in Applied Mathematics.

[16] E. Seneta, *Non-negative Matrices and Markov Chains*, revised ed. New York, NY: Springer, 2006.

[17] J. Niño-Mora, "Dynamic allocation indices for restless projects and queueing admission control: a polyhedral approach," *Mathematical Programming*, vol. 93, no. 3, pp. 361–413, 2002.

[18] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, 2nd ed. SIAM, 2002.

[19] J. Pineau, G. Gordon, and S. Thrun, "Point-based value iteration: An anytime algorithm for pomdps," in *Proc. IJCAI*, 2003, pp. 725–732.

[20] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA: MIT Press, 2009.