Opportunistic Learning for Markov Decision Systems with Application to Smart Robots

Michael J. Neely
University of Southern California
https://viterbi-web.usc.edu/~mjneely/

***Abstract*— This paper presents an online method that learns optimal decisions for a discrete time Markov decision problem with an opportunistic structure. The state at time t is a pair $(S(t), W(t))$ where $S(t)$ takes values in a finite set $\mathcal{S}$ of *basic states*, and $\{W(t)\}_{t=0}^{\infty}$ is an i.i.d. sequence of random vectors that affect the system and that have an unknown distribution. Every slot t the controller observes $(S(t), W(t))$ and chooses a control action $A(t)$. The triplet $(S(t), W(t), A(t))$ determines a vector of costs and the transition probabilities for the next state $S(t+1)$. The goal is to minimize the time average of an objective function subject to additional time average cost constraints. We develop an algorithm that acts on a corresponding virtual system where $S(t)$ is replaced by a decision variable. An equivalence between virtual and actual systems is established by enforcing a collection of time averaged global balance equations. For any desired $\epsilon > 0$, we prove the algorithm achieves an ϵ-optimal solution on the virtual system with a convergence time of $O(1/\epsilon^2)$. The actual system runs at the same time, its actions are informed by the virtual system, and its conditional transition probabilities and costs are proven to be the same as the virtual system at every instant of time. Also, its unconditional probabilities and costs are shown in simulation to closely match the virtual system. Our simulations consider online control of a robot that explores a region of interest. Objects with varying rewards appear and disappear and the robot learns what areas to explore and what objects to collect and deliver to a home base.**

I. Introduction

This paper considers a Markov decision system that operates in slotted time $t \in \{0, 1, 2, \ldots\}$. The state of the system is given by a pair $(S(t), W(t))$, where $S(t)$ takes values in a finite set $\mathcal{S} = \{1, \ldots, n\}$ of *basic states* (where n is a positive integer), and $\{W(t)\}_{t=0}^{\infty}$ is a sequence of independent and identically distributed (i.i.d.) random vectors that take values in a (possibly infinite) set $\mathcal{W}$. The value $W(t)$ can represent a random fluctuation or event that affects the system on slot t, such as a random vector of rewards, costs, or side information. The distribution of $W(t)$ is unknown to the system controller. This is an *opportunistic Markov decision problem* because the controller can observe the value of $W(t)$ at the start of slot t and can use this knowledge to inform its action. Specifically, every slot t the controller observes $(S(t), W(t))$ and chooses an action $A(t)$ from an action set $\mathcal{A}$. The triplet $(S(t), W(t), A(t))$ determines a vector of costs incurred on slot t and also the transition probability associated with the next basic state $S(t+1)$.

The action has the form $A(t) = (A_1(t), \ldots, A_n(t))$, where $A_i(t) \in \mathcal{A}_i$ is a *contingency action* given that $S(t) = i$. Assume $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$. For $i, j \in \mathcal{S}$ define a *transition probability function* $p_{i,j}(w, a_i)$ so

$$P[S(t+1) = j | S(t) = i, W(t), A(t), H(t)] = p_{i,j}(W(t), A_i(t)) \quad (1)$$

where $H(t)$ is the history before slot t. The Markov property holds: The value $S(t+1)$ is conditionally independent of $H(t)$ given the current $(S(t), W(t), A(t))$.

Fix k as a nonnegative integer. For $i \in \mathcal{S}$ and $l \in \{0, 1, \ldots, k\}$ define *cost functions* $c_{i,l}(w, a_i)$. The cost vector for slot t is $C(t) = (C_0(t), C_1(t), \ldots, C_k(t))$ where

$$C_l(t) = c_{S(t),l}(W(t), A_{S(t)}(t)) \quad (2)$$

for $l \in \{0, \ldots, k\}$, where 1_E is an indicator function that is 1 when event E is true and 0 else. The goal is to make decisions over time to produce random processes $\{S(t)\}_{t=0}^{\infty}$ and $\{C(t)\}_{t=0}^{\infty}$ that solve:

$$\text{Minimize:} \quad \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[C_0(t)\right] \quad (3)$$

$$\text{Subect to:} \quad \limsup_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[C_l(t)\right] \leq 0 \quad \forall l \in \{1, \ldots, k\} \quad (4)$$

$$A(t) \in \mathcal{A} \quad \forall t \in \{0, 1, 2, \ldots\} \quad (5)$$

There are two main challenges: First, the dimension of $W(t)$ can be large and its corresponding set $\mathcal{W}$ can be infinite, so the full state space $\mathcal{S} \times \mathcal{W}$ is overwhelming. Second, the distribution of the i.i.d. random vectors $\{W(t)\}_{t=0}^{\infty}$ is unknown. It is not always possible to estimate the distribution in a timely manner. This paper develops a low complexity algorithm that learns to make efficient decisions that drive the system close to optimality. Our algorithm depends on n, the number of basic states, and its implementation and convergence time is independent of the dimension of $W(t)$ and the size of $\mathcal{W}$. The idea is that, rather than learn the full distribution, it is sufficient to learn certain *max weight functionals*. The algorithm can be viewed as a Markov chain based generalization of the drift-plus-penalty algorithm in [1] for opportunistic network scheduling. This paper focuses simulations on a toy example of a roving robot, described in the next subsection.

A. Robot example

Consider a robot that seeks out valuable objects over a region and delivers them to a home base (see Fig. 1). Ob-

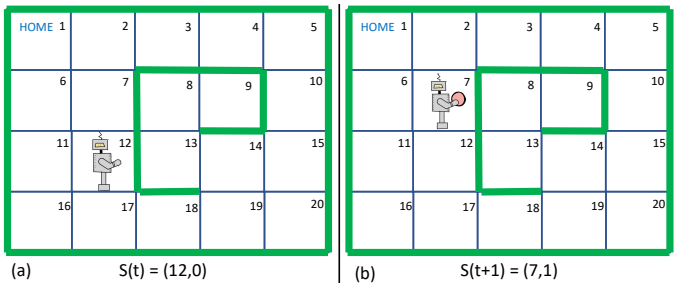

Fig. 1: (a) The robot has current state $S(t) = (12, 0)$ because it is in location 12 and is not holding an object. Suppose there is an object at location 12, and the robot decides to collect this object (earning reward $W_{12}(t)$) and then move to location 7; (b) The robot transitions to state $S(t+1) = (7, 1)$ because it is now in location 7 and is holding an object. It will continue to hold the object (and hence is blocked from accumulating more rewards) until it returns to the home location 1 (where it "deposits" the object at home and transitions to state $(1, 0)$).

jects randomly appear and disappear in each location. Define $W(t) = (W_1(t), \ldots, W_{20}(t))$, where $W_i(t)$ is the value of the object in location i at time t (if there is currently no object in location i then $W_i(t) = 0$). Assume $\{W(t)\}_{t=0}^{\infty}$ are i.i.d. random vectors with a joint distribution that is unknown to the robot. On each slot t, the robot has a satellite view of the full reward vector $W(t)$.

The basic state is $S(t) = (Location(t), Hold(t))$, where $Location(t) \in \{1, \ldots, 20\}$ indicates the current location of the robot, and $Hold(t) \in \{0, 1\}$ indicates whether or not the robot is currently holding an object. Thus, there are $20 \times 2 = 40$ basic states ($n = 40$). The rules are as follows: The robot can hold at most one object at a time; The robot can only drop an object at the home base (its state $Hold(t)$ moves to 0 when entering location 1); On each slot t: If the robot is not currently holding an object, it decides whether or not to pick up the object (if any) at its current location; Then, the robot decides to either stay in its current location or move to an adjacent location in any feasible direction to the North, West, South, or East. Thus, given $S(t) = (i, h)$, the action $A_{i,h}(t)$ is always one of 10 actions of the form $(collect, move)$ where $collect \in \{0, 1\}$ and $move \in \{Stay, N, S, W, E\}$. Actions within this set of 10 are removed from consideration if they are impossible in state (i, h), such as moving in a direction blocked by a wall, or collecting a new object when the robot is already holding one.

Since every object must be brought home before new ones are collected, the robot must be careful not to waste time carrying low-valued objects. The goal is to maximize time average reward, equivalent to minimizing $\overline{C}_0$ where: $C_0(t) = -W_{Location(t)}(t)1_{collect}(t)$; $1_{collect}(t)$ is a binary variable that is 1 if and only if the robot collects an object at time t. For this example, the transition probability functions $p_{(i,h),(j,h')}(\cdot)$ are binary valued and depend only on $A_{i,h}(t)$, so the next state $(j, h') \in \mathcal{S}$ is determined by the current state and action. This example has $k = 0$, so there are no additional cost constraints $\overline{C}_l \leq 0$. Our technical report [2] also considers an additional time average power constraint.

Let r^* denote the optimal time average reward that can be achieved, considering all decision strategies. The value r^* (and the optimal strategy) depends on the distribution of $W(t)$ rather than just its mean value $\mathbb{E}[W(t)] = (\mathbb{E}[W_1(0)], \ldots, \mathbb{E}[W_{20}(0)])$. It is not possible to approximately learn the full distribution in a timely manner. Something more efficient must be done. Further, it is not obvious how to achieve r^* even if the distribution were fully known.

B. Comparing against distribution-aware strategies

To illuminate the structure of the robot problem, consider the following distribution on $W(t)$: Entries of $W(t)$ are mutually independent; $W_1(t) = 0$ surely (no object appears at the home location); for $i \in \{2, \ldots, 20\}$, $W_i(t) = B_i(t)R_i(t)$ with $B_i(t) \sim Bern(1/2)$ being 1 if and only if an object appears in cell i on slot t, $R_i(t)$ is the reward of the object that appears; $R_9(t) \sim$ Unif$[0, 20]$, $R_{16}(t) \sim$ Unif$[0, 4]$, $R_i(t) \sim$ Unif$[0, 1]$ for $i \in \{2, \ldots, 20\} \setminus \{9, 16\}$. With these parameters, the most valuable objects tend to appear in location 9, which is the most difficult location to reach (see Fig. 1). The second most valuable location is 16, while all other locations tend to have low valued objects.

In the algorithm of our paper, the robot must *learn* to return home as quickly as possible after it collects an object. However, it is useful to compare performance of our algorithm against heuristic strategies that have a frame-based renewal structure and that are fine tuned with knowledge of the problem structure and probability distributions. Several heuristics are considered in our technical report [2], we describe one below (called "heuristic 2").

Heuristic 2: Fix a parameter $\theta \in [0, 20)$; Starting from location 1, move to location 9 over any path that takes exactly 10 steps (ignoring all rewards along the way); stay in location 9 until we see an object with value $W_9(t) > \theta$; collect this and return home using any 10-step path. By renewal-reward theory, the time average reward is

$$\overline{r} = \frac{\frac{1}{2}(\theta + 20)}{19 + \frac{40}{20-\theta}} = 0.66791$$

where the numerical value is obtained by maximizing over $\theta \in [0, 20)$, achieved at $\theta^* = 12.690$.

With this reward distribution, it is desirable to visit the hard-to-get location 9 to obtain higher-valued rewards. It is not clear whether or not Heuristic 2 is optimal. However, our simulated results for the algorithm of this paper, implemented online over 10^6 time slots, yields a time average reward in a *virtual system* of $\overline{r} = 0.6672$, and in the *actual system* of $\overline{r} = 0.6604$ (the concept of *virtual* and *actual* systems is made apparent when the algorithm is presented). This suggests that Heuristic 2 is either optimal or near optimal. It also suggests that our online algorithm learns to visit home so it can refresh its $Hold(t)$ state, learns to ignore low-valued objects, learns the shortest paths to and from location 9, learns near-optimal thresholding rules, all without knowing the distribution of the rewards. The reported values 0.6672 and 0.6604 for our algorithm are time averages over $t \in \{0, \ldots, 10^6\}$, so these averages include the relatively small rewards earned early on when the robot is just starting to learn efficient behavior.

C. Prior work

Our paper characterizes optimality of the stochastic problem in terms of a deterministic and nonconvex problem (8)-(12). This deterministic problem is reminiscent of linear programming representations of optimality for simpler Markov decision problems that do not have the opportunistic scheduling aspect and that have finite state and action sets (see, for example, [3][4][5]). In principle, our nonconvex problem (8)-(12) can be transformed into a convex problem by a nonlinear change of variables similar to methods for linear fractional programming in [6][7]. The work [7] uses the nonlinear transformation for offline computation of an optimal Markov decision policy. However, the approaches in [6][7] do not help for our context because: (i) The deterministic problem (8)-(12) uses abstract and unknown sets $\overline{\Gamma}_i$ that make the resulting convex problem complicated; (ii) We seek an online solution with desirable time averages, and time averages are not preserved under nonlinear transformations. Classical descriptions of optimality for dynamic programming and Markov decision problems with general Borel spaces are in [8][9][10].

Our paper uses a classical Lyapunov drift technique pioneered by Tassiulas and Ephremides for stabilizing queueing networks [11][12]. Specifically, we use an extended *drift-plus-penalty* method that incorporates a penalty function to jointly minimize time average cost subject to stability of certain virtual queues [1]. Such methods have been extensively used for opportunistic scheduling in data networks with unknown arrival and channel probabilities [13][14][15]. Other opportunistic scheduling approaches are stochastic Frank-Wolfe methods [16][17][18], fluid model techniques [19], and related dual and primal-dual approaches [20][21][22][23].

The drift-plus-penalty method was used to treat Markov decision problems (MDPs) with an opportunistic scheduling aspect in [24][25]. The work [24] is the most similar to the current paper. That work uses drift-plus-penalty theory on a virtual system. However, the algorithm that runs on the virtual system requires knowledge of certain *max-weight functionals* that depend on unknown probability distributions. For this, it estimates the functionals by sampling over a window of past $W(t)$ samples (see also [26]). This slows down learning time and makes a precise convergence analysis difficult. In contrast, the current paper develops a new layered stochastic optimization technique that operates online, on a single timescale, and does not require averaging over a window of past samples.

Related problems of robot navigation with problem structures different from ours are in [27][28][29][30]. Online MDPs are treated in [31][32][33][34]. The work [31] treats a known transition probability model but adversarial costs that are revealed *after* a decision is made. An $O(\sqrt{T})$ regret algorithm is developed using online convex programming and a quasi-stationary assumption on the Markov chain. The model is extended in [32] to allow time varying constraint costs and coupled multi-chains, again with $O(\sqrt{T})$ regret, see also a recent treatment in [33]. MDPs where transition probabilities are allowed to vary slowly over time are considered in [35]. The above works have a different structure from the current paper and do not have an opportunistic learning aspect. Also, the works [31][32][33] take a 2-timescale approach and transform the decision variables to a vector in a "policy class," which requires each decision to solve a linear program associated with a stationary distribution for a certain "time-t" MDP. In contrast, the current paper operates on one timescale and uses an easier "max-weight" type decision on every slot.

A shortest path problem with random context variables, similar to our opportunistic variables $W(t)$, is treated in [34] (see also [36] for context-based shortest paths with unknown but unchanging contexts). The work [34] provides a Dijkstra-based algorithm when distributions are known, and develops Thompson sampling and reinforcement learning methods when the distribution is unknown. The works [36][34] treat problems with structure different from ours and do not treat time average cost constraints.

D. Our contributions

Similar to [24], our paper focuses on a virtual system. However, our virtual system has a simpler and more direct structure. We develop an algorithm where the virtual system observes the current $W(t)$ and chooses a *contingency action* $A_i(t)$ for each $i \in \mathcal{S}$. Equivalence between virtual and actual systems is enforced by imposing time averaged global balance constraints. Our algorithm uses a single timescale and a novel layered structure and, unlike [24], does not require estimation over a window of past samples. This enables explicit performance guarantees for time average expected cost. Specifically, for any desired $\epsilon > 0$, we show the virtual system achieves within $O(\epsilon)$ of optimality with convergence time $1/\epsilon^2$. The complexity and convergence time are independent of the dimension of $W(t)$. This virtual algorithm also provides a simple online algorithm for the actual system. We show that for all time t, the actual and virtual systems have the same *conditional* transition probability and *conditional* expected cost given the current state. Simulations show the two systems also match in their unconditional costs. The resulting algorithm is simple and operates online with no training.

II. System model

Fix integers $n > 0, k \geq 0$. The basic states are $\mathcal{S} = \{1, \ldots, n\}$. Let c_{max} bound the magnitude of all costs:

$$|C_l(t)| \leq c_{max} \quad \forall l \in \{0, \ldots, k\}, t \in \{0, 1, 2, \ldots\}$$

For each $i \in \mathcal{S}$, let $(\mathcal{A}_i, \mathcal{G}_i)$ be a Borel space that is used for the actions $A_i(t)$ when $S(t) = i$. For example, $\mathcal{A}_i$ can be any nonempty Borel subset of $\mathbb{R}^{k_i}$ for some positive integer k_i, with $\mathcal{G}_i$ being its standard Borel sigma algebra. Let $(\mathcal{W}, \mathcal{G}_W)$ be any measurable space that is used for the random elements $W(t)$. Define measurable *cost* and *transition probability functions* of the form

$$c_{i,l} : \mathcal{W} \times \mathcal{A}_i \to [-c_{max}, c_{max}]$$
$$p_{i,j} : \mathcal{W} \times \mathcal{A}_i \to [0, 1]$$

for $i, j \in \mathcal{S}$ and $l \in \{0, \ldots, k\}$. The $p_{i,j}$ functions satisfy

$$\sum_{j=1}^{n} p_{i,j}(w, a) = 1 \quad \forall i \in \mathcal{S}, w \in \mathcal{W}, a \in \mathcal{A}_i$$

A. Probability space

The probability space is $(\Omega, \mathcal{F}, P)$. Let $\{W(t)\}_{t=0}^{\infty}$, $\{U(t)\}_{t=0}^{\infty}$, $\{\tilde{U}(t)\}_{t=-1}^{\infty}$ be mutually independent, where

- $\{W(t)\}_{t=0}^{\infty}$ are i.i.d. random elements in $\mathcal{W}$.
- $\{U(t)\}_{t=0}^{\infty}$ are i.i.d. Unif$[0,1]$ (generated in software at the controller to enable randomized actions).
- $\{\tilde{U}(t)\}_{t=-1}^{\infty}$ are i.i.d. Unif$[0,1]$ (generated by "nature" to determine state transitions from $S(t)$ to $S(t+1)$).

Let $\{S(t)\}_{t=0}^{\infty}$ be the sequence of basic states. Fix an initial state $s_0 \in \mathcal{S}$ and assume $S(0) = s_0$ surely. Define $H(0) = 0$ and for $t \in \{1, 2, 3, \ldots\}$ define the *history* $H(t)$ by

$$H(t) = (S(0), ..., S(t-1), W(0), ..., W(t-1), U(0), \ldots, U(t-1))$$

Define $D_H(t)$ as the set of possible values of $H(t)$. Let $\mathcal{G}_H(t)$ be the product sigma algebra on $D_H(t)$.

B. Control policies

A *causal and measurable policy* is a sequence of measurable functions $\{h_t\}_{t=0}^{\infty}$ of the type:

$$h_t : \mathcal{W} \times [0,1] \times D_H(t) \to \mathcal{A}$$

where $\mathcal{A} = \times_{i=1}^{n} \mathcal{A}_i$ and $h_t = (h_{t,1}, \ldots, h_{t,n})$. These specify $A(t) = (A_1(t), \ldots, A_n(t))$ for each $t \in \{0, 1, 2, \ldots\}$ by

$$A_i(t) = h_{t,i}(W(t), U(t), H(t)) \quad \forall i \in \mathcal{S} \tag{6}$$

One can view $U(t)$ as a random number generated by software at the controller at the start of each slot t. A causal and measurable policy is said to be *memoryless* if it does not depend on $H(t)$, so $A(t) = x_t(W(t), U(t))$ for some measurable functions $x_t : \mathcal{W} \times [0,1] \to \mathcal{A}$ for $t \in \{0, 1, 2, \ldots\}$.

C. State transitions

Define $\mathcal{P}$ as the probability simplex:

$$\mathcal{P} = \{(q_1, \ldots, q_n) : \textstyle\sum_{i=1}^{n} q_i = 1, q_i \geq 0 \quad \forall i \in \{1, \ldots, n\}\}$$

For $i \in \{1, \ldots, n\}$ define $P_i(t) \in \mathcal{P}$ by

$$P_i(t) = [p_{i,1}(W(t), A(t)), \ldots, p_{i,n}(W(t), A(t))]$$

Given $(S(t), W(t), A(t), \tilde{U}(t))$, the next state $S(t+1)$ is (measurably) chosen according to distribution $P_{S(t)}(t)$ using $\tilde{U}(t)$ as an independent random seed.

III. Optimality

A. The sets Γ_i

Fix $i \in \mathcal{S}$. Let $\mathcal{X}_i$ be the collection of all measurable functions $x_i : \mathcal{W} \times [0,1] \to \mathcal{A}_i$. Define

$$\mathcal{I} = \{(l, j) : l \in \{0, 1, ..., k\}, j \in \mathcal{S}\}$$

Define $g_i : \mathcal{W} \times \mathcal{A} \to \mathbb{R}^{2|\mathcal{I}|}$ by

$$g_i(w, a) = (c_{i,l}(w, a); p_{i,j}(w, a))_{(l,j) \in \mathcal{I}} \quad \forall w \in \mathcal{W}, a \in \mathcal{A}_i$$

For notational simplicity, define $W = W(0)$, $U = U(0)$. Then W and U are independent and $U \sim$ Unif$[0,1]$. For each $x_i \in \mathcal{X}_i$, the random vector $g_i(W, x_i(W, U))$ is bounded and has finite expectation. Its expectation is a vector of conditional expected costs and transition probabilities for a slot t, given that $S(t) = i$ and a memoryless control action $A_i(t) = x_i(W(t), U(t))$ is taken. Define $\Gamma_i \subseteq \mathbb{R}^{2|\mathcal{I}|}$ by

$$\Gamma_i = \{\mathbb{E}[g_i(W, x_i(W, U))] : x_i \in \mathcal{X}_i\} \tag{7}$$

Define $\overline{\Gamma}_i$ as the closure of set Γ_i. It can be shown that $\overline{\Gamma}_i$ is compact and convex [2].

B. Optimal cost

Consider a deterministic optimization with decision variables $\pi_i, c_{i,l}, p_{i,j}$ for $i, j \in \mathcal{S}$ and $l \in \{0, \ldots, k\}$:

$$\text{Minimize:} \quad \sum_{i \in \mathcal{S}} \pi_i c_{i,0} \tag{8}$$

$$\text{Subject to:} \quad \pi_j = \sum_{i \in \mathcal{S}} \pi_i p_{i,j} \quad \forall j \in \mathcal{S} \tag{9}$$

$$\sum_{i \in \mathcal{S}} \pi_i c_{i,l} \leq 0 \quad \forall l \in \{1, \ldots, k\} \tag{10}$$

$$(\pi_1, \ldots, \pi_n) \in \mathcal{P} \tag{11}$$

$$((c_{i,l}); (p_{i,j})) \in \overline{\Gamma}_i \quad \forall i \in \mathcal{S} \tag{12}$$

This problem (8)-(12) is nonconvex due to the multiplication of variables in (8), (9), (10). The problem is said to be *feasible* if it is possible to satisfy the constraints (9)-(12). Assuming feasibility, define c_0^* as the infimum objective value in (8) subject to the constraints (9)-(12). The next two results are proven in [2].

Theorem 1: (Converse) Suppose the stochastic problem (3)-(5) is feasible when $s_0 \in \mathcal{S}$. Then the deterministic problem (8)-(12) is also feasible. Further, if $S(0) = s_0$ surely and if $\{A(t)\}_{t=0}^{\infty}$ are causal and measurable control actions of the form (6) that satisfy the constraints (4)-(5) then

$$\liminf_{T \to \infty} \frac{1}{T} \textstyle\sum_{t=0}^{T-1} \mathbb{E}[C_0(t)] \geq c_0^* \tag{13}$$

Theorem 2: (Achievability) Suppose Γ_i is a closed set for each $i \in \mathcal{S}$. If the deterministic problem (8)-(12) is feasible, there is an initial state $s_0 \in \mathcal{S}$ for which the stochastic problem (3)-(5) is feasible. Further, there exists a measurable function $x : \mathcal{W} \times [0,1] \to \mathcal{A}$ such that when $S(0) = s_0$ surely, using the memoryless actions $A^*(t) = x(W(t), U(t))$ for $t \in \{0, 1, 2, \ldots\}$ yields costs that satisfy

$$\lim_{T \to \infty} \frac{1}{T} \textstyle\sum_{t=0}^{T-1} \mathbb{E}[C_0^*(t)] = c_0^*$$

$$\lim_{T \to \infty} \frac{1}{T} \textstyle\sum_{t=0}^{T-1} \mathbb{E}[C_l^*(t)] \leq 0 \quad \forall l \in \{1, \ldots, k\}$$

The assumption that sets Γ_i are closed is crucial for existence of a single memoryless action function $c : \mathcal{W} \times [0,1] \to \mathcal{A}$. A counter-intuitive example by Blackwell in [37] shows that memoryless actions are insufficient for systems defined on Borel sets with nonBorel projections, see also [38].

Theorem 2 requires a proper choice of $s_0 \in \mathcal{S}$. Specifically, the memoryless action induces a homogeneous discrete time Markov chain with transition probabilities $P = (P_{ij})$ over $i, j \in \mathcal{S}$, and s_0 should be chosen as any state in a *desirable communicating class*. The transition probability matrix P can also have *undesirable* communicating classes over which the global balance equations are satisfied but optimal cost is not achieved. Our simulations of the robot example in [2] show

these undesirable communicating classes can arise in certain cases when it is optimal to restrict the robot to a small subset of locations (so an "irreducible" type property fails). The *virtual system* is unaffected by this issue. However, the *actual system* can get trapped in an undesirable communicating class. Fortunately, this situation is easy to detect and a simple online fix is discussed in [2] that maintains a close match between virtual and actual system in all simulations.

C. Lagrange multipliers

While problem (8)-(12) is nonconvex, it can be shown that the set of all vectors of the form

$$(\pi; (\pi_i c_{i,l})_{i\in\mathcal{S}, l\in\{0,\ldots,k\}}; (\pi_i p_{ij})_{i,j\in\mathcal{S}}) \tag{14}$$

for some values $\pi = (\pi_1, \ldots, \pi_n), c_{i,l}, p_{i,j}$ that satisfy

$$\pi \in \mathcal{P} \tag{15}$$

$$((c_{i,l}); (p_{i,j})) \in \overline{\Gamma}_i \quad \forall i \in \mathcal{S} \tag{16}$$

is a convex set. Therefore, the following additional *Lagrange multiplier assumption* is mild.

Assumption 1: (Lagrange multipliers) Problem (8)-(12) is feasible with optimal objective c_0^*. There are real numbers $\mu_l \geq 0$ for $l \in \{0, \ldots, k\}$ and $\lambda_j \in \mathbb{R}$ for $j \in \mathcal{S}$ such that

$$\sum_{i\in\mathcal{S}} \pi_i c_{i,0} + \sum_{j\in\mathcal{S}} \lambda_j \left(\pi_j - \sum_{i\in\mathcal{S}} \pi_i p_{i,j}\right) + \sum_{l=1}^{k} \mu_l \left(\sum_{i\in\mathcal{S}} \pi_i c_{i,l}\right) \geq c_0^*$$

for all $(\pi; (c_{i,l}); (p_{i,j}))$ that satisfy (15)-(16).

In [2] it is shown that Assumption 1 implies that for any random element $W : \Omega \to \mathcal{W}$ with the same distribution as $W(0)$, any random element $\pi : \Omega \to \mathcal{P}$ that is independent of W, and any random element $A : \Omega \to \mathcal{A}$ with arbitrary dependence on (π, W) we have

$$\sum_{i\in\mathcal{S}} \mathbb{E}\left[\pi_i c_{i,0}(W, A_i)\right] + \sum_{j\in\mathcal{S}} \lambda_j \mathbb{E}\left[\pi_j - \sum_{i\in\mathcal{S}} \pi_i p_{i,j}(W, A_i)\right] + \sum_{l=1}^{k} \mu_l \sum_{i\in\mathcal{S}} \mathbb{E}\left[\pi_i c_{i,l}(W, A_i)\right] \geq c_0^* \tag{17}$$

IV. Algorithm development

A. K-L divergence

Define $\mathcal{P}_o \subseteq \mathcal{P}$ by

$$\mathcal{P}_o = \{(\pi_1, \ldots, \pi_n) : \textstyle\sum_{i=1}^n \pi_i = 1, \pi_i > 0 \quad \forall i\}$$

Define $D : \mathcal{P} \times \mathcal{P}_o \to \mathbb{R}$ as the Kullback-Leibler (K-L) divergence function:

$$D(p; q) = \sum_{i=1}^{n} p_i \log(p_i/q_i) \quad \forall p \in \mathcal{P}, q \in \mathcal{P}_o$$

where $\log(\cdot)$ denotes the natural logarithm. It is well known that $D(\cdot)$ is a nonnegative function that satisfies

$$D(p; (1/n, \ldots, 1/n)) \leq \log(n) \quad \forall p \in \mathcal{P} \tag{18}$$

$$D(p; q) \geq \frac{1}{2}||p - q||_1^2 \quad \forall p \in \mathcal{P}, q \in \mathcal{P}_o \tag{19}$$

where $||x||_1 = \sum_{i=1}^n |c_i|$. Inequality (19) is the *Pinsker inequality*. The following *pushback lemma* is standard [39][40][41]:

Lemma 1: (Pushback) Fix $q \in \mathcal{P}_o$, $\alpha \geq 0$. Let $A \subseteq \mathcal{P}$ be a convex set and $f : A \to \mathbb{R}$ a convex function. Consider:

$$\text{Minimize:} \quad f(p) + \alpha D(p; q) \tag{20}$$

$$\text{Subject to:} \quad p \in A \tag{21}$$

If p^{opt} solves (20)-(21), and if $p^{opt} \in \mathcal{P}_o$, then for all $p \in A$:

$$f(p^{opt}) + \alpha D(p^{opt}; q) \leq f(p) + \alpha D(p; q) - \alpha D(p, p^{opt})$$

B. Time averaged problem

The algorithm observes $W(t)$ and makes contingency actions $A_i(t) \in \mathcal{A}_i$ for all $i \in \mathcal{S}$ and $t \in \{0, 1, 2, \ldots\}$. It introduces a decision vector $\pi(t) = (\pi_1(t), \ldots, \pi_n(t)) \in \mathcal{P}$ that intuitively represents the state probability on a virtual system where we can choose any state in $\mathcal{S}$ on each new slot. The goal is to make these decisions over time to solve:

$$\text{Minimize:} \quad \overline{\sum_{i\in\mathcal{S}} \pi_i(t) c_{i,0}(W(t), A_i(t))} \tag{22}$$

Subject to:

$$\overline{\pi_j(t) - \sum_{i\in\mathcal{S}} \pi_i(t) p_{i,j}(W(t), A_i(t))} = 0 \quad \forall j \in \mathcal{S} \tag{23}$$

$$\overline{\sum_{i\in\mathcal{S}} \pi_i(t) c_{i,l}(W(t), A_i(t))} \leq 0 \quad \forall l \in \{1, \ldots, k\} \tag{24}$$

$$\pi(t) \in \mathcal{P} \quad \forall t \in \{0, 1, 2, ...\} \tag{25}$$

$$A_i(t) \in \mathcal{A}_i \quad \forall i \in \mathcal{S}, t \in \{0, 1, 2, ...\} \tag{26}$$

$$\pi(t) \text{ and } W(t) \text{ are independent for each } t \tag{27}$$

where the overbar notation in (22), (23), (24) denotes a limiting time average expectation (assuming existence for simplicity). The above problem is a time averaged version of the deterministic problem (8)-(12). Specifically, the constraints (23)-(25) directly relate to the deterministic constraints (9)-(11). The constraints (23) shall be called the *global balance constraints*. These play a crucial role in the learning process. The constraints (26), (27) together correspond to the deterministic constraint (12). Constraint (27) is a nonstandard and subtle constraint that prevents $\pi(t)$ from being influenced by the realization of $W(t)$. Without (27), the decisions for $\pi(t)$ would be biased towards those states for which there is a favorable realization of $W(t)$, which might not correspond to a solution that can be achieved on the actual system. Our method of enforcing (27) ensures an algorithm complexity that depends only on the size of the finite set $\mathcal{S}$ (which is n), rather than the size of the (possibly infinite) set $\mathcal{S} \times \mathcal{W}$.

C. Virtual queues

For $t \in \{0, 1, 2, \ldots\}$ define the $n \times 1$ vector $G_0(t)$ by

$$G_0(t) = (c_{1,0}(W(t), A_1(t)), \ldots, c_{n,0}(W(t), A_n(t)))$$

Define the $n \times n$ matrix $Y(t) = (Y_{i,j}(t))$ and the $n \times k$ matrix $G(t) = (G_{i,l}(t))$ by

$$Y_{i,j}(t) = 1_{\{i=j\}} - p_{i,j}(W(t), A_i(t)) \quad \forall i, j \in \mathcal{S} \tag{28}$$

$$G_{i,l}(t) = c_{i,l}(W(t), A_i(t)) \quad \forall i \in \mathcal{S}, l \in \{1, \ldots, k\} \tag{29}$$

where $1_{\{i=j\}}$ is 1 if $i = j$, and 0 else. For initialization, define $G_0(-1) = -(c_{max}, \ldots, c_{max})$, $Y(-1) = 0$, $G(-1) = 0$.

Using the virtual queue technique of [1] to enforce the constraints (23), (24), define processes $Q_j(t)$ and $Z_l(t)$ with initial conditions $Q_j(0) = 0$, $Z_l(0) = 0$ and update equation for $t \in \{0, 1, 2, \ldots\}$ given for $j \in \mathcal{S}$ and $l \in \{1, ..., k\}$ by

$$Q_j(t+1) = Q_j(t) + \pi(t)^\top Y(t-1)y_j \quad (30)$$

$$Z_l(t+1) = \max\left[Z_l(t) + \pi(t)^\top G(t-1)g_l, 0\right] \quad (31)$$

where $\pi^\top(t) = (\pi_1(t), \ldots, \pi_n(t))$ is a row vector; $y_j \in \mathbb{R}^n$ is the unit vector that is 1 in entry j and 0 in all other entries; $g_l \in \mathbb{R}^k$ is the unit vector that is 1 in entry j and 0 in all other entries. In particular, $Y(t-1)y_j$ selects the jth column of matrix $Y(t-1)$, while $G(t-1)g_l$ selects the lth column of matrix $G(t-1)$. Technical report [2] shows that maintaining small virtual queues enforces constraints (23)-(24).

D. Lyapunov drift

For $t \in \{0, 1, 2, \ldots\}$ define

$$Q(t) = (Q_1(t), \ldots, Q_n(t)) \ , \ Z(t) = (Z_1(t), \ldots, Z_k(t))$$

and $J(t) = (Q(t); Z(t))$. Define

$$L(t) = \tfrac{1}{2}||J(t)||^2 = \tfrac{1}{2}\sum_{j=1}^n Q_j(t)^2 + \tfrac{1}{2}\sum_{l=1}^k Z_l(t)^2$$

Define $\Delta(t) = L(t+1) - L(t)$.

Lemma 2: For all $t \in \{0, 1, 2, \ldots\}$ and all choices of the decision variables we have

$$\Delta(t) \leq b_1 + \pi(t)^\top Y(t-1)Q(t) + \pi(t)^\top G(t-1)Z(t)$$

where $b_1 = \frac{n + kc_{max}^2}{2}$.

Proof: The result follows by squaring (30) and (31) and using $\max[z, 0]^2 \leq z^2$ for $z \in \mathbb{R}$, which is a standard Lyapunov optimization technique (see, for example, [1]). □

Fix $V > 0, \alpha > 0$ as parameters to be sized later. From Lemma 2 we have

$$\begin{aligned}&\Delta(t) + V\pi(t)^\top G_0(t-1) + \alpha D(\pi(t); \pi(t-1)) \\ &\leq b_1 + \pi(t)^\top \left[VG_0(t-1) + Y(t-1)Q(t) + G(t-1)Z(t)\right] \\ &\quad + \alpha D(\pi(t); \pi(t-1)) \end{aligned} \quad (32)$$

The following algorithm uses a layered structure to make the right-hand-side of the above expression small.

E. Online algorithm

Initialize:

$$\begin{aligned}&Q_j(-1) = Y_{i,j}(-1) = 0 \quad \forall i, j \in \mathcal{S} \\ &Z_l(-1) = G_{i,l}(-1) = 0 \quad \forall l \in \{1, \ldots, k\}, i \in \mathcal{S} \\ &G_0(-1) = -(c_{max}, \ldots, c_{max}) \\ &\pi(-1) = (1/n, 1/n, ..., 1/n) \in \mathcal{P}\end{aligned}$$

On each slot $t \in \{0, 1, 2, \ldots\}$ do:

- Layer 1: Ignore $W(t)$. Choose $\pi(t) \in \mathcal{P}$ to minimize

$$\begin{aligned}&\pi(t)^\top \left[VG_0(t-1) + Y(t-1)Q(t) + G(t-1)Z(t)\right] \\ &\quad + \alpha D(\pi(t); \pi(t-1))\end{aligned} \quad (33)$$

  This has the following solution $\pi(t) \in \mathcal{P}_o$:

$$\pi_i(t) = \frac{\pi_i(t-1)\exp(-M_i(t)/\alpha)}{\sum_{j=1}^n \pi_j(t-1)\exp(-M_j(t)/\alpha)} \quad \forall i \in \mathcal{S}$$

  where $M_i(t)$ is defined for $i \in \mathcal{S}$ by

$$\begin{aligned}&M_i(t) \\ &= y_i^\top \left(VG_0(t-1) + Y(t-1)Q(t) + G(t-1)Z(t)\right)\end{aligned}$$

  where $y_i \in \mathbb{R}^n$ is the unit vector that is 1 in entry i and 0 in all other entries.
- Layer 2: Observe $W(t)$. For each $i \in \mathcal{S}$ choose $A_i(t) \in \mathcal{A}_i$ to minimize:[1]

$$\begin{aligned}&Vc_{i,0}(W(t), A_i(t)) + \textstyle\sum_{l=1}^k Z_l(t)c_{i,l}(W(t), A_i(t)) \\ &- \textstyle\sum_{j=1}^n Q_j(t)p_{i,j}(W(t), A_i(t))\end{aligned} \quad (34)$$

- Virtual queue update by (30) and (31).
- Actual system implementation: Observe the actual system state $S(t) \in \mathcal{S}$. Apply action $A_{S(t)}$.

F. Layered analysis

Throughout, assume the sets Γ_i are closed for all $i \in \mathcal{S}$ (so $\Gamma_i = \overline{\Gamma}_i$) and the problem (8)-(12) is feasible with optimal solution $(\pi^*; (c_{i,l}^*); (p_{i,j}^*))$ where $\pi^* \in \mathcal{P}$ and

$$((c_{i,l}^*); (p_{i,j}^*))_{(l,j)\in\mathcal{I}} \in \Gamma_i \quad \forall i \in \mathcal{S} \quad (35)$$

For each $i \in \mathcal{S}$, by definition of Γ_i, there is a function $x_i \in \mathcal{X}_i$ such that defining $A_i^*(t) = x_i(W(t), U(t))$ yields

$$\mathbb{E}\left[c_{i,l}(W(t), A_i^*(t))\right] = c_{i,l}^* \quad (36)$$

$$\mathbb{E}\left[p_{i,j}(W(t), A_i^*(t))\right] = p_{i,j}^* \quad (37)$$

for all $t \in \{0, 1, 2, \ldots\}$, all $l \in \{0, \ldots, k\}$, $j \in \mathcal{S}$.

Lemma 3: For all $t \in \{0, 1, 2, \ldots\}$ we have

$$\begin{aligned}&\mathbb{E}\left[\Delta(t) + V\pi(t)^\top G_0(t-1) + \alpha D(\pi(t); \pi(t-1))\right] \\ &\leq b + Vc_0^* + \alpha\mathbb{E}\left[D(\pi^*; \pi(t-1)) - D(\pi^*; \pi(t))\right]\end{aligned} \quad (38)$$

where $b = (3/2)(n + kc_{max}^2)$.

Proof: Fix $t \in \{0, 1, 2, \ldots\}$. For each $i \in \mathcal{S}$, the layer 2 decision chooses $A_i(t) \in \mathcal{A}_i$ to minimize (34) and so

$$\begin{aligned}&Vc_{i,0}(W(t), A_i(t)) + \sum_{l=1}^k Z_l(t)c_{i,l}(W(t), A_i(t)) \\ &- \sum_{j=1}^n Q_j(t)p_{i,j}(W(t), A_i(t)) \\ &\leq Vc_{i,0}(W(t), A_i^*(t)) + \sum_{l=1}^k Z_l(t)c_{i,l}(W(t), A_i^*(t)) \\ &- \sum_{j=1}^n Q_j(t)p_{i,j}(W(t), A_i^*(t))\end{aligned}$$

where $A_i^*(t)$ is any other (possibly randomized) decision in $\mathcal{A}_i$. Multiplying the above by π_i^*, summing over $i \in$

[1]For simplicity we assume a minimizer $A_i(t) \in \mathcal{A}_i$ exists. This holds when $\mathcal{A}_i$ is a finite set. More generally we can use a δ-approximation to the infimum with insignificant impact on our results.

$\{1,\ldots,n\}$, and adding $\sum_{j=1}^{n} Q_j(t)\pi_j^*$ to both sides gives (using definitions of $Y(t)$ and $G(t)$ in (28) and (29)):

$$\begin{aligned}&\pi^{*\top}\left[VG_0(t)+G(t)Z(t)+Y(t)Q(t)\right]\\&\leq V\sum_{i=1}^{n}\pi_i^* c_{i,0}(W(t),A_i^*(t))\\&+\sum_{l=1}^{k}Z_l(t)\sum_{i=1}^{n}\pi_i^* c_{i,l}(W(t),A_i^*(t))\\&+\sum_{j=1}^{n}Q_j(t)\left[\pi_j^*-\sum_{i=1}^{n}\pi_i^* p_{i,j}(W(t),A_i^*(t))\right]\end{aligned}\tag{39}$$

For $i\in\mathcal{S}$, let $A_i^*(t)=x_i(W(t),U(t))$ be the action that is independent of history $H(t)$ (hence independent of the virtual queues) that yields (36)-(37). Taking expectations of (39) gives

$$\begin{aligned}&\pi^{*\top}\mathbb{E}\left[VG_0(t)+G(t)Z(t)+Y(t)Q(t)\right]\\&\leq V\sum_{i=1}^{n}\pi_i^* c_{i,0}^*+\sum_{l=1}^{k}Z_l(t)\sum_{i=1}^{n}\pi_i^* c_{i,l}^*\\&+\sum_{j=1}^{n}Q_j(t)\left[\pi_j^*-\sum_{i=1}^{n}\pi_i^* p_{i,j}^*\right]\end{aligned}$$

By definition of a solution to (8)-(12) we have $\sum_{i=1}^{n}\pi_i^* c_{i,l}^*\leq 0$, $\pi_j^*=\sum_{i=1}^{n}\pi_i^* p_{i,j}^*$, and $c_0^*=\sum_{i=1}^{n}\pi_i^* c_{i,0}^*$ and so

$$\pi^{*\top}\mathbb{E}\left[VG_0(t)+G(t)Z(t)+Y(t)Q(t)\right]\leq Vc_0^*\tag{40}$$

By definition of $G_0(-1)=-(c_{max},\ldots,c_{max})$, $Q(-1)=0$, $Z(-1)=0$, inequality (40) also holds for time $t=-1$.

The layer 1 decision chooses $\pi(t)$ in the convex set $\mathcal{P}$ to minimize $\alpha D(\pi(t);\pi(t-1))$ plus a linear function of $\pi(t)$, so by the pushback lemma (Lemma 1):

$$\begin{aligned}&\pi(t)^\top\left[VG_0(t-1)+Y(t-1)Q(t)+G(t-1)Z(t)\right]\\&+\alpha D(\pi(t);\pi(t-1))\\&\leq\pi^{*\top}\left[VG_0(t-1)+G(t-1)Z(t)+Y(t-1)Q(t)\right]\\&+\alpha D(\pi^*;\pi(t-1))-\alpha D(\pi^*;\pi(t))\end{aligned}$$

Define $b_2=kc_{max}^2+n$. Since $|Z_l(t)-Z_l(t-1)|\leq c_{max}$, $|G_{i,l}(t-1)|\leq c_{max}$, $|Q_j(t)-Q_j(t-1)|\leq 1$, and $|Y_{i,j}(t-1)|\leq 1$ we obtain from the previous inequality

$$\begin{aligned}&\mathbb{E}\left[\pi(t)^\top\left[VG_0(t-1)+Y(t-1)Q(t)+G(t-1)Z(t)\right]\right]\\&+\alpha\mathbb{E}\left[D(\pi(t);\pi(t-1))\right]\\&\leq b_2+\alpha\mathbb{E}\left[D(\pi^*;\pi(t-1))-D(\pi^*;\pi(t))\right]\\&+\pi^{*\top}\mathbb{E}\left\{VG_0(t-1)+G(t-1)Z(t-1)\right.\\&\qquad\left.+Y(t-1)Q(t-1)\right\}\\&\overset{(a)}{\leq} b_2+Vc_0^*+\alpha\mathbb{E}\left[D(\pi^*;\pi(t-1))-D(\pi^*;\pi(t))\right]\end{aligned}$$

where (a) uses the fact that (40) holds for all $t\in\{-1,0,1,2,\ldots\}$. Adding b_1 to both sides, defining $b=b_1+b_2$, and substituting into inequality (32) proves the result. □

Now fix $\epsilon>0$ and define $\alpha=1/\epsilon^2$, $V=\rho/\epsilon$ for some positive constant ρ (such as $\rho=1$). Details in [2] use the above lemma (mainly by summing it over t) to prove that for all $T\geq 1/\epsilon^2$, the virtual system reaches an $O(\epsilon)$-approximate solution, meaning that for all $l\in\{1,...,k\}$:

$$\begin{aligned}&\tfrac{1}{T}\textstyle\sum_{t=0}^{T-1}\mathbb{E}\left[\pi(t)^\top G_0(t)\right]\leq c_0^*+O(\epsilon)\\&\tfrac{1}{T}\textstyle\sum_{t=0}^{T-1}\mathbb{E}\left[\pi(t)^\top G(t)g_l\right]\leq O(\epsilon)\end{aligned}$$

and all global balance equations are within $O(\epsilon)$ of being satisfied. There, it is also shown that for all slots t, the conditional transition probabilities and costs, given $S(t)=i$, match across the virtual and actual systems. For unconditional probability and cost, simulations in [2] for the robot example show an odd behavior related to *undesirable communicating classes* that arise in some rare cases where irreducibility fails. This situation is easy to detect and fix online: If the robot spends a significant fraction of time in an actual state that the virtual system says should have near-zero probability, the robot takes a shortest path to home. This happens infrequently and, with this simple fix, the unconditional probabilities and rewards are close between virtual and actual systems across a wide range of parameter settings (see Fig. 2). Simulations with time average cost constraints are also in [2].

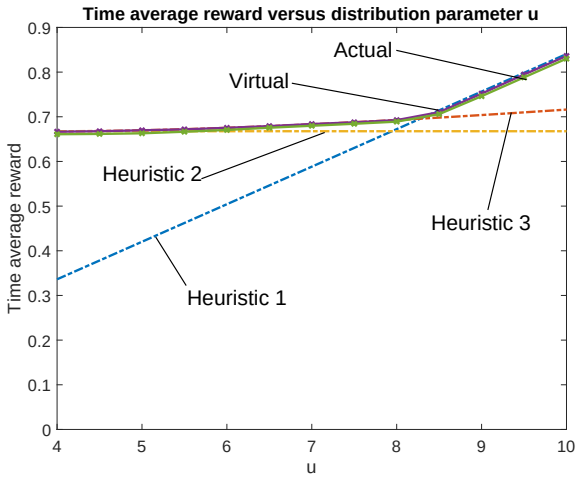

Fig. 2: Robot simulations for $V=5$, $\alpha=1000$, $T=10^6$. Distribution similar to Section I-B with the exception $R_{16}(t)\sim\text{Unif}[0,u]$ where u is a parameter in the x-axis.

V. Conclusion

This work extends the max-weight and drift-plus-penalty methods of stochastic network optimization to opportunistic Markov decision systems. The basic state variable $S(t)$ can take one of n values, but the full state $(S(t),W(t))$ is augmented by a sequence of i.i.d. random vectors $\{W(t)\}$ that can be observed at the start of each slot t but have an unknown distribution. The dimension of $W(t)$ and the cardinality of its set of possible values does not impact convergence or complexity. An online algorithm that operates on a virtual system is shown to reach an $O(\epsilon)$-approximate solution with convergence time $O(1/\epsilon^2)$. The algorithm also acts online for the actual system. Simulations show a close match between state probabilities and costs in the virtual and actual systems.

REFERENCES

[1] M. J. Neely. *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.

[2] Michael J. Neely. Opportunistic learning for Markov decision systems with application to smart robots. *arXiv:2408.05322*, 2024.

[3] S. Ross. *Introduction to Probability Models*. Academic Press, 8th edition, Dec. 2002.

[4] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2005.

[5] H. Mine and S. Osaki. *Markovian Decision Processes*. American Elsevier, New York, 1970.

[6] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[7] B. Fox. Markov renewal programming by linear fractional programming. *Siam J. Appl. Math*, vol. 14, no. 6, Nov. 1966.

[8] D. Blackwell. Discounted dynamic programming. *Annals of Mathematical Statistics*, 1964.

[9] A. Maitra. Discounted dynamic programming on compact metric spaces. *Indian Journal of Statistics, Series A*, 30(2):211–216, 1968.

[10] M. Schäl. On dynamic programming: Compactness of the space of policies. *Stochastic Processes and their Applications*, 3:345–364, 1975.

[11] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936-1948, Dec. 1992.

[12] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, vol. 39, no. 2, pp. 466-478, March 1993.

[13] L. Tassiulas and A. Ephremides. Throughput properties of a queueing network with distributed dynamic routing and flow control. *Advances in Applied Probability*, vol. 28, pp. 285-307, 1996.

[14] M. J. Neely. Energy optimal control for time varying wireless networks. *IEEE Transactions on Information Theory*, vol. 52, no. 7, pp. 2915-2934, July 2006.

[15] M. J. Neely, E. Modiano, and C. Li. Fairness and optimal stochastic control for heterogeneous networks. *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 396-409, April 2008.

[16] H. Kushner and P. Whiting. Asymptotic properties of proportional-fair sharing algorithms. *Proc. 40th Annual Allerton Conf. on Communication, Control, and Computing, Monticello, IL*, Oct. 2002.

[17] R. Agrawal and V. Subramanian. Optimality of certain channel aware scheduling policies. *Proc. 40th Annual Allerton Conf. on Communication, Control, and Computing, Monticello, IL*, Oct. 2002.

[18] M. J. Neely. Convergence and adaptation for utility optimal opportunistic scheduling. *IEEE/ACM Transactions on Networking*, 27(3):904–917, June 2019.

[19] A. Stolyar. Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm. *Queueing Systems*, vol. 50, no. 4, pp. 401-457, 2005.

[20] X. Liu, E. K. P. Chong, and N. B. Shroff. A framework for opportunistic scheduling in wireless networks. *Computer Networks*, vol. 41, no. 4, pp. 451-474, March 2003.

[21] A. Eryilmaz and R. Srikant. Joint congestion control, routing, and MAC for stability and fairness in wireless networks. *IEEE Journal on Selected Areas in Communications, Special Issue on Nonlinear Optimization of Communication Systems*, vol. 14, pp. 1514-1524, Aug. 2006.

[22] A. Eryilmaz and R. Srikant. Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control. *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1333-1344, Dec. 2007.

[23] A. Stolyar. Greedy primal-dual algorithm for dynamic resource allocation in complex networks. *Queueing Systems*, vol. 54, no. 3, pp. 203-220, 2006.

[24] M. J. Neely. Asynchronous control for coupled Markov decision systems. *Proc. Information Theory Workshop (ITW)*, 2012.

[25] M. J. Neely. Online fractional programming for Markov decision systems. *Proc. Allerton Conf. on Communication, Control, and Computing*, Sept. 2011.

[26] M. J. Neely, S. T. Rager, and T. F. La Porta. Max weight learning algorithms for scheduling in unknown environments. *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1179-1191, May 2012.

[27] R. Dimitrova, I. Gavran, R. Majumdar, V. S. Prabhu, S. Soudjani, and E. Zadeh. The Robot Routing Problem for Collecting Aggregate Stochastic Rewards. In *28th International Conference on Concurrency Theory (CONCUR 2017)*, volume 85 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:17, Dagstuhl, Germany, 2017.

[28] Satoshi Hoshino and Shingo Ugajin. Adaptive patrolling by mobile robot for changing visitor trends. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, page 104–110. IEEE Press, 2016.

[29] R. Stranders, E. Munoz de Cote, A. Rogers, and N.R. Jennings. Near-optimal continuous patrolling with teams of mobile information gathering agents. *Artificial Intelligence*, 195:63–105, 2013.

[30] A. Dutta, , O. P. Kreidl, and J. M. O'Kane. Opportunistic multi-robot environmental sampling via decentralized Markov decision processes. In Fumitoshi Matsuno, Shun-ichi Azuma, and Masahito Yamamoto, editors, *Distributed Autonomous Robotic Systems*, pages 163–175, Cham, 2022. Springer International Publishing.

[31] E. Eyal, S. M. Kakade, and Y. Mansour. Online Markov decision processes. *Mathematics of Operations Research*, 34(3), 2009.

[32] X. Wei, H. Yu, and M. J. Neely. Online learning in weakly coupled Markov decision processes: A convergence time study. *ACM Meas. Anal. Comput. Syst.*, March 2018.

[33] P. Zhao, L. Li, and Z. Zhou. Dynamic regret of online Markov decision processes. *Proc. 39th Int. Conf. on Machine Learning*, 2022.

[34] Xinghan Wang, Gregory Fields, and Tara Javidi. Contextual shortest path with unknown context distributions. In *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pages 471–476, 2021.

[35] I. Szita, B. Takács, and A. Lörincz. ϵ-mdps: Learning in varying environments. *Journal of Machine Learning Research*, 3:145–174, 2002.

[36] Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual Markov decision processes, 2015.

[37] D. Blackwell. Memoryless strategies in finite-stage dynamic programming. *Annals of Mathematical Statistics*, 1963.

[38] L. E. Dubins and L. J. Savage. *How to Gamble if you Must: Inequalities for Stochastic Processes*. Courier Corp., 2014.

[39] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. *submitted to SIAM J. Optimization*, 2008.

[40] X. Wei, H. Yu, and M. J. Neely. Online primal-dual mirror descent under stochastic constraints. *Proc. ACM Meas. Anal. Comput. Syst*, 4(2), 2020.

[41] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.