Scalable Operator Allocation for Multi-Robot Assistance: A Restless Bandit Approach

Abhinav Dahiya, Nima Akbarzadeh, Aditya Mahajan, Stephen L. Smith

***Abstract*—In this paper, we consider the problem of allocating human operators in a system with multiple semi-autonomous robots. Each robot is required to perform an independent sequence of tasks, subject to a chance of failing and getting stuck in a fault state at every task. If and when required, a human operator can assist or teleoperate a robot. Conventional dynamic programming-based techniques used to solve such problems face scalability issues due to exponential growth of state and action spaces with the number of robots and operators. In this paper we derive conditions under which the operator allocation problem satisfies a technical condition called indexability, thereby enabling the use of the Whittle index heuristic. The conditions are easy to check, and we show that they hold for a wide range of problems of interest. Our key insight is to leverage the structure of the value function of individual robots, resulting in conditions that can be verified separately for each state of each robot. We apply these conditions to two types of transitions commonly seen in remote robot supervision systems. Through numerical simulations, we demonstrate the efficacy of Whittle index policy as a near-optimal and scalable approach that outperforms existing scalable methods.**

***Index Terms*—Human-robot collaboration, Restless bandits, Markov decision processes, decision support systems**

I. Introduction

Advances in robot autonomy have led to a decrease in the necessity of strict human supervision of robots. This has enabled the development of human-robot collaborative systems where the task is primarily executed by a number of semi-autonomous robots requiring intermittent assistance from a human teammate, either in event of a *fault* [1], [2] or to further increase performance of the multi-agent teams in warehouse operation [3], search-and-rescue [4] or in a social setting [5]. However, identifying which robot to assist in an uncertain environment is a challenging task for human operators [3], [6]. Moreover, as the number of robots increases, it becomes challenging for the operators to maintain awareness of every robot, which cripples system's performance [7], [8]. Therefore, human operators can benefit from having a decision support system (DSS) that advises which robots require attention and when [3], [9].

This research is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and in part by the Innovation for Defence Excellence and Security (IDEaS) Program of the Canadian Department of National Defence through grant CFPMN2-037, and Fonds de Recherche du Quebec-Nature et technologies (FRQNT).

Abhinav Dahiya and Stephen L. Smith are with Department of Electrical and Computer Engineering, University of Waterloo, Waterloo (abhinav.dahiya@uwaterloo.ca, stephen.smith@uwaterloo.ca)

Nima Akbarzadeh and Aditya Mahajan are with Department of Electrical and Computer Engineering, McGill University, Montreal. (nima.akbarzadeh@mail.mcgill.ca, aditya.mahajan@mcgill.ca)

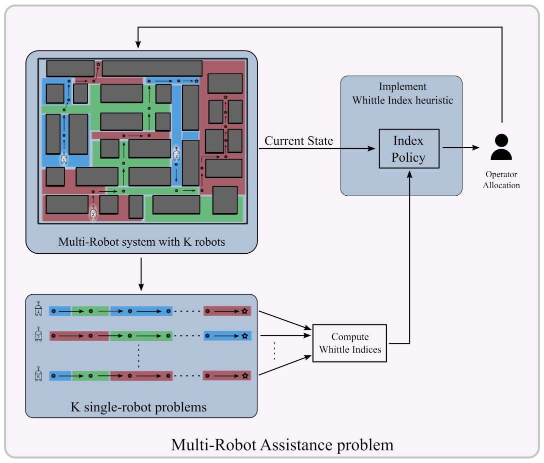
Multi-Robot system with K robots
Current State
Implement Whittle Index heuristic
Index Policy
Operator Allocation
K single-robot problems
Compute Whittle Indices
Multi-Robot Assistance problem

Fig. 1. Overview of the multi-robot assistance problem for robots navigating in an environment. A number of mobile robots are tasked to navigate through a series of waypoints. Operators are allocated to the robots when required. This is done by decomposing the complete K-robot problem into K single robot problems and computing the Whittle index heuristic. Given the current state of the system, this heuristic can be used to efficiently compute the operator allocation.

In this paper, we present such a DSS for a multi-robot system comprising a fleet of semi-autonomous robots with multiple human operators available for assistance if and when required. Figure 1 presents an overview of the problem setup, showing K robots navigating in a city block-like environment, moving from start to goal locations. While navigating, a robot passes through a series of waypoints, each characterized by a different probability of success in progressing to the next waypoint. There is also a possibility that the robot may fail at a task and get stuck in a fault (error) state from which assistance from a human operator is required to continue. There are M identical human operators available ($M \leq K$), each of whom can assist/teleoperate at most one robot at a time. While being assisted by an operator, robots have different probabilities of success and failure, and to get out of a fault state.

It is possible to solve the above problem by modelling it as a Markov Decision Process (MDP) [10]. However, such formulations suffer from the curse of dimensionality, making conventional MDP-based solution techniques scale poorly with problem size [11]. Moreover, the policy needs to be re-computed every time a robot or an operator enters or leaves the system. To tackle the scalability issue, researchers have proposed solutions based on simple myopic policies, sampling-based planner or an approximation algorithm [3], [12], [13]. However, these solutions either do not apply to stochastic settings, or they do not scale well with increasing number of robots and/or operators.

In this paper, we show how an index-based policy can provide a scalable and better performing solution than the existing approaches for the given multi-robot multi-operator allocation problem with stochastic transitions. Specifically, our work makes the following contributions:

1) We show that the operator allocation problem with multiple independent robots can be formulated as an instance of the Restless Multi-Armed Bandit (RMAB) problem. We leverage this formulation to decompose the problem into several single-robot problems and computing the Whittle index heuristic (see Fig. 1). The resulting policy scales linearly with the number of robots and is independent of the number of operators.

2) We derive simple conditions to verify indexability of the model. These conditions can be checked independently for each state of each robot, thus providing a method that scales linearly with the size of the problem. This method can be applied to systems with any number of states and does not require the optimal policy to be of a threshold type.

3) We then implement our approach in two practical scenarios and present numerical experiments. The results show that the proposed method provides near-optimal solutions and outperforms existing efficient solution approaches, namely the reactive policy, 1- and 2-step myopic policies, and the benefit-maximizing policy.

The appendix contains details about the relevant proofs. A more detailed version of the proofs can be found in the arXiv preprint of the paper [14].

A. Background and Related Work

The problem of allocating operators in a multi-robot team bears similarities with the disciplines of multi-robot supervision, task scheduling and queuing theory. In this section, we briefly review the related research, followed by an introduction of Restless-Multi Armed Bandits.

In the literature, several studies discuss the problem of enabling human operators to assist multiple robots such as a team of navigating robots, a fleet of multiple UAVs, or a team performing search and rescue operations [15], [16]. To understand and improve human supervision, researchers have used frameworks such as sliding autonomy to incorporate various human-robot team capabilities (like coordination and situational awareness) [17], [18]. Some studies also present interaction interfaces to facilitate and improve such supervision [19], [20].

The most closely related work to our problem is presented in [3], where the authors discuss single-operator multi-robot supervision systems. An advising agent guides the operator on which robot they should assist. The problem is solved using an l-step look-ahead (myopic) approach, which provides an efficient and practical solution, but suffers from scalability issues with increasing number of operators and the look-ahead steps. Researchers have also discussed deterministic versions of the problem, where exact outcomes of robots' actions and times for fault occurrences are known, and the allocation policy is determined using a sampling-based planner [12]. In [13], the authors present an approximation algorithm for a similar scheduling problem. These approaches however are not applicable in a stochastic setting.

The problem of assisting a number of independent robots, has also been studied under a learning framework. The approach presented in [2] learns the decision-making model of human operator from recorded data and tries to replicate that behaviour, optimizing based on the operator's internal utility function. In contrast, the problem presented in this paper is designed to optimize a global performance metric assuming the knowledge of success and failure rates of robots with and without an operator allocated to them. Such knowledge can be estimated using recorded data similar to the work presented in [3]. For the scope of this paper, we will assume this knowledge takes the form of known transition probabilities.

In the queuing discipline, several studies have investigated the effects of different queuing techniques [21] or threshold-based strategies [22] to prioritize operator's attention to the robots. However, the model that we study is different from a queuing model as it is possible for the robots to complete their tasks without the help of operators, and for the operators to be allocated to robots not stuck in a fault state.

The multi-target–multi-agent problems form another class of problems similar to the operator allocation problem. These problems deal with allocation of multiple agents to a number of targets aiming to detect or follow the targets under certain constraints [23], [24]. However, our problem setup is different because the behaviour of the targets (robots) changes with the allocation of agents (operators) and it is not possible to allocate multiple agents to a single target at once. Moreover, our problem presents a collaborative task, where both the robots and operators are working to achieve a common goal.

Restless Multi-Armed Bandit: Restless Multi-Armed Bandits (RMAB), first introduced in [25], is a generalization of Multi-Armed Bandits (MAB) [26] which has been previously used in problems like assisting human partners [27] and distributing resources among human teammates [28]. RMAB is a class of scheduling problems where limited resources have to be allocated among several alternative choices. Each choice, referred to as an *arm*, is a discrete-time controlled Markov process which evolves depending on the resource allocated to it. RMAB framework has been applied to problems in stochastic scheduling, patrol planning, sensor management and resource allocation in general [29].

Finding the optimal policy for RMAB suffers from the curse of dimensionality as the state space grows exponentially with the number of arms. In general, obtaining the optimal policy in an RMAB is PSPACE-hard [30]. However, the *Whittle index policy* offers a simpler and scalable alternative to the optimal policy. Even though the Whittle index policy does not guarantee an optimal solution, it minimizes expected cost for a relaxed problem under time-averaged constraint [25]. This approach is shown to work quite well for several scheduling and resource allocation problems [31]–[33]. A few studies have also implemented index-based methods to solve a sensor scheduling problem [34] or to serve a number of users transmitting a queue of data packets through a channel [32]. Therefore, it is a reasonable approach to solve an RMAB given that the problem satisfies a technical condition known as indexability (more details in Section III). Unfortunately, it is difficult to verify this condition in general and there is no universal framework that applies to all problems. Existing methods proposed for verifying indexability have been investigated for specific systems such as two state restless bandits [35], [36] or restless bandits with optimal threshold-based policy [35], [37], [38].

B. Organization

The contents of this paper are organized as follows: The multi-robot assistance problem is presented in Section II. We discuss the general Restless Bandit Problem and define property of indexability in Section III. In Section V, we present two practical classes of transition functions and establish conditions under which problem indexability is ensured. In Section VI, we cover the calculation of Whittle index heuristic and present an efficient policy for the problem. Next, we present simulations of the problem in Section VII to examine validity and performance of the presented policy. The paper ends with a brief discussion and conclusion.

II. Multi-Robot Assistance Problem

Consider a decision support system (DSS), consisting of a team of M human operators supervising a fleet of K semi-autonomous robots. Each robot $k \in \mathcal{K} := \{1, \dots, K\}$ is required to complete a sequence of N^k tasks to reach its goal. We will use a fleet of robots delivering packages in a city as a running example but similar interpretations hold for other applications mentioned in previous sections (e.g., robots reaching a sequence of configurations [12]). In this case, the robot's trajectory would correspond to a series of waypoints that a robot needs to navigate to reach its destination (goal location). At each waypoint, a robot can either operate autonomously or be teleoperated by one of the human operators. We assume that all human operators are identical in the way they operate the robots and that a human operator can operate at most one robot at a time. We now provide a mathematical model for different components of the system[1].

A. Model of the robots

It is assumed that when operating autonomously, each robot uses a pre-specified control algorithm to complete its task. For the delivery robot example, this could be, for instance, a SLAM-based local path planner that the robot uses for navigating between the waypoints. We will not model the details of this control algorithm but simply assume that this control is imperfect and occasionally causes the robot to enter a fault state while doing a task (e.g., delivery robot getting stuck in a pothole or losing its localization). We model this behaviour by assuming that while completing each task, the robot may be in one of the two internal states: a *normal* state (denoted by $s = 0$) or a *fault* state (denoted by $s = 1$). When a robot is being teleoperated, it may still be possible for it to enter into a fault state.

[1]**Remark on notation:** Throughout this paper, we use calligraphic font to denote sets and roman font to denote variables. Uppercase letters are used to represent random variables and the corresponding lowercase letters represent their realizations. Bold letters are used for variables pertaining to multi-robot system while light letters represent corresponding single-robot variables.

The operating state of robot $k \in \mathcal{K}$ at time t, denoted by $x_t^k = (n_t^k, s_t^k)$, is tuple of its current task and internal state. The state space for robot k is given by

$$\mathcal{X}^k := \bigcup_{n=1}^{N^k} \{(n,0),(n,1)\} \cup \{(G,0)\},$$

where the terminal state $(G, 0)$ indicates that all tasks have been completed. The state space for all robots is denoted by $\boldsymbol{\mathcal{X}} = \mathcal{X}^1 \times \cdots \times \mathcal{X}^K$.

The state of a robot evolves differently depending on whether it is operating autonomously (denoted by mode $a^k = 0$) or teleoperated (denoted by $a^k = 1$). Given robot $k \in \mathcal{K}$ in state $(n, s) \in \mathcal{X}^k$ operating in mode $a \in \{0, 1\}$, let p_{ns}^{ka} denote the probability of successfully completing the current task at the current time step and let q_{ns}^{ka} denote the probability of toggling the current internal state (i.e. going from normal to fault state and vice-versa). A diagram describing these transitions is shown in Fig. 2. Note that the terminal state $(G, 0)$ is an absorbing state, so $p_{G0}^{ka} = 0$ and $q_{G0}^{ka} = 0$.

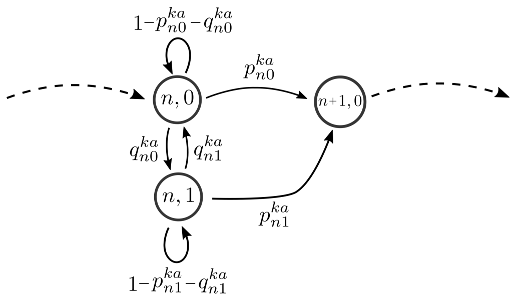

Fig. 2. State-transition diagram for robot k working on task n, where in $(n, 0)$ the robot is in the normal state $s = 0$, and in $(n, 1)$ the robot is in the fault state $s = 1$. Transitions can occur between $(n, 0), (n, 1)$ and $(n + 1, 0)$, and the probabilities change with operating mode a.

There is a per-step cost $C^k : \mathcal{X}^k \times \{0, 1\} \to \mathbb{R}_{\geq 0}$, where $C^k((n^k, s^k), a^k)$ denotes the cost of operating robot $k \in \mathcal{K}$ in mode a^k when the robot is in state (n^k, s^k). Note that the per-step cost is zero in the terminal state, i.e, $C^k((G, 0), a) = 0$.

B. Model of the decision support system (DSS)

There is a decision support system that helps to allocate operators to the robots. At each time the decision support system observes the operating state $\boldsymbol{X}_t := (X_k^1, \dots, X_t^K)$ of all robots and picks at most M robots to teleoperate. We capture this by the allocation $\boldsymbol{A}_t = (A_t^1, \dots, A_t^K) \in \mathcal{A}$, where

$$\mathcal{A} = \left\{ \boldsymbol{a} := (a^1, \dots, a^K) \in \{0,1\}^K : \sum_{k=1}^{K} a^k \leq M \right\}. \quad (1)$$

The allocation is selected according to a time-homogeneous Markov policy $\pi : \boldsymbol{\mathcal{X}} \to \mathcal{A}$. The expected total cost incurred by any policy π is given by

$$J(\pi) = \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \sum_{k=1}^{K} C^k(X_t^k, A_t^k) \;\middle|\; \boldsymbol{X}_0 = \boldsymbol{x}_0\right], \tag{2}$$

where $\gamma \in (0,1)$ is the discount factor and $\boldsymbol{x}_0 = (x_0^1, \dots, x_0^K)$ is the initial state with $x_0^k = (1,0)$ for every $k \in \mathcal{K}$.

C. Problem objective

We impose the following assumptions on the model:

(A1) Given an allocation $\boldsymbol{a} = (a^1, \dots, a^K)$ by the DSS, the operating states of the robots evolve independently of each other.

(A2) For every robot $k \in \mathcal{K}$, the probability of getting out the faulty internal state when teleoperated is strictly greater than 0, i.e., $p_{n1}^{k1} + q_{n1}^{k1} > 0$.

(A3) Under autonomous operation, a robot stays in the fault state, i.e., $p_{n1}^{k0} = q_{n1}^{k0} = 0$.

The design objective is to solve the following optimization problem:

Problem 1. Given the set $\mathcal{K}$ of robots, the system dynamics and the per-step costs, the number M of human operators, and the discount factor $\gamma \in (0,1)$, choose a policy $\pi : \boldsymbol{\mathcal{X}} \to \mathcal{A}$ to minimize the total discounted cost $J(\pi)$ given by (2).

Optimal solution for Problem 1 can be found by modelling it as a Markov decision process and solving using dynamic programming [10]. However, the sizes of state and action spaces of the resulting model grows exponential with the number of robots and operators. Thus, solving Problem 1 using dynamic programming becomes intractable for larger systems. To address this, we model Problem 1 as a restless multi-armed bandit (RMAB) problem and use the notion of indexability to find an efficient and scalable policy. We start by providing an overview of RMAB in the next section.

III. Overview of Restless Multi-armed bandits

In this section we provide an overview of Restless Multi-Armed Bandits (RMAB), indexability and the Whittle index policy.

A. Restless Bandit Process

A restless bandit (RB) process is a controlled Markov process $(\tilde{\mathcal{Z}}, \{0,1\}, \tilde{T}, \tilde{C}, \tilde{z}_0)$ where $\tilde{\mathcal{Z}}$ is the state space, $\{0,1\}$ is the action space, $\tilde{T}\colon \tilde{\mathcal{Z}} \times \tilde{\mathcal{Z}} \times \{0,1\} \to \mathbb{R}_{[0,1]}$ is the transition probability function, $\tilde{C} : \tilde{\mathcal{Z}} \times \{0,1\} \to \mathbb{R}$ is the per-step cost function, and $\tilde{z}_0$ is the initial state. By convention, action 0 is called the *passive* action and action 1 is called the *active* action.

B. Restless Multi-armed Bandit Problem

A Restless Multi-armed Bandit (RMAB) is a collection of K independently evolving RBs $(\tilde{\mathcal{Z}}^k, \{0,1\}, \tilde{T}^k, \tilde{C}^k, \tilde{z}_0^k)$, $k \in \mathcal{K} := \{1, \dots, K\}$. Each process is conventionally called an *arm*. A decision-maker selects at most M arms ($M < K$) at each time instance. Let $\tilde{Z}_t^k$ and $\tilde{A}_t^k$ denote the state of arm k and the action chosen for arm k at time t. Let $\{\tilde{\boldsymbol{Z}}_t\}_{t\ge 0}$ and $\{\tilde{\boldsymbol{A}}_t\}_{t\ge 0}$ where

$$\tilde{\boldsymbol{Z}}_t := (\tilde{Z}_t^1, \dots, \tilde{Z}_t^K) \text{ and } \tilde{\boldsymbol{A}}_t := (\tilde{A}_t^1, \dots, \tilde{A}_t^K),$$

denote the states and actions of all arms. As the dynamics of each arm are independent, we have

$$\tilde{T}(\tilde{\boldsymbol{Z}}_{t+1} | \tilde{\boldsymbol{Z}}_t, \tilde{\boldsymbol{A}}_t) = \prod_{k\in\mathcal{K}} \tilde{T}^k(\tilde{Z}_{t+1}^k | \tilde{Z}_t^k, \tilde{A}_t^k).$$

The instantaneous cost of the system is the sum of costs incurred by each arm. The performance of any time homogeneous Markov policy $\tilde{\boldsymbol{\pi}} : \prod_{k=1}^K \mathcal{Z}^k \to \{\boldsymbol{a} \in \{0,1\}^K : ||\boldsymbol{a}||_1 \le M\}$ is measured by

$$\tilde{J}(\tilde{\boldsymbol{\pi}}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \sum_{k=1}^{K} \tilde{C}^k(\tilde{Z}_t^k, \tilde{\boldsymbol{\pi}}(\tilde{Z}_t^k)) \middle| \tilde{z}_0^1, \dots, \tilde{z}_0^K\right], \tag{3}$$

where $\gamma \in (0,1)$ denotes the discount factor. Finally, the RMAB optimization problem is as follows:

Problem 2. Given a discount factor $\gamma \in (0,1)$, a collection of arms $\{(\tilde{\mathcal{Z}}^k, \{0,1\}, \tilde{T}^k, \tilde{C}^k, \tilde{z}_0^k)\}_{k\in\mathcal{K}}$, and the number M of arms to be chosen at each time, choose a policy $\tilde{\boldsymbol{\pi}} : \prod_{k=1}^K \mathcal{Z}^k \to \{\boldsymbol{a} \in \{0,1\}^K : ||\boldsymbol{a}||_1 \le M\}$ that minimizes $\tilde{J}(\tilde{\boldsymbol{\pi}})$.

Even though the arms operate independently, the actions applied to them are not independent. They are coupled through the operator allocation constraints. Therefore, we cannot decompose the dynamic programming into multiple smaller MDPs. As discussed earlier, the Whittle index policy is one of the commonly used heuristic to solve a RMAB problem [25] and it addresses the scalability issues of dynamic programming-based solutions. This policy is computationally efficient and it readily generalizes to the setting where K or M changes over time. Next, we present the required definitions.

C. Indexability and the Whittle index policy

In this section, we restrict our discussion to a single arm and therefore omit the superscript k for the ease of notation. Consider an arm $(\tilde{\mathcal{Z}}, \{0,1\}, \tilde{T}, \tilde{C}_\lambda, \tilde{z}_0)$ where, for some penalty $\lambda \in \mathbb{R}$, modify the per-step cost as

$$\tilde{C}_\lambda(z,a) := \tilde{C}(z,a) + \lambda a, \quad \forall\, z \in \tilde{\mathcal{Z}}, a \in \{0,1\}. \tag{4}$$

Then the performance of any given time-homogeneous Markov policy $\tilde{\pi} : \tilde{\mathcal{Z}} \to \{0,1\}$ is given by

$$\tilde{J}_\lambda(\tilde{\pi}) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \tilde{C}_\lambda(\tilde{Z}_t, \tilde{\pi}(\tilde{Z}_t)) \middle| \tilde{Z}_0 \sim \tilde{z}_0\right]. \tag{5}$$

Now consider the following auxiliary problem:

Problem 3. Given an arm $(\tilde{\mathcal{Z}}, \{0,1\}, \tilde{T}, \tilde{C}, \tilde{z}_0)$, the discount factor $\gamma \in (0,1)$ and the penalty $\lambda \in \mathbb{R}$, choose a Markov policy $\tilde{\pi} : \tilde{\mathcal{Z}} \to \{0,1\}$ to minimize $\tilde{J}_\lambda^{(\tilde{\pi})}(\tilde{z}_0)$ given by (5).

Problem 3 is a Markov decision process. Let us denote the optimal policy of Problem 3 by $\tilde{\pi}_\lambda$. It is assumed that the optimal policy picks passive action at any state where both the active and passive actions result in same expected cost. Next, define passive sets and indexability.

Definition 1 (Passive set)**.** Given $\lambda \in \mathbb{R}$, the passive set $\tilde{\mathcal{P}}(\lambda)$ is the set of states where passive action is prescribed by $\tilde{\pi}_\lambda$, i.e.,

$$\tilde{\mathcal{P}}(\lambda) := \{z \in \mathcal{Z} : \tilde{\pi}_\lambda(z) = 0\}.$$

Definition 2 (Indexability)**.** An arm is indexable if $\tilde{\mathcal{P}}(\lambda)$ is non-decreasing in λ, i.e., for any $\lambda_1, \lambda_2 \in \mathbb{R}$,

$$\lambda_1 \le \lambda_2 \implies \tilde{\mathcal{P}}(\lambda_1) \subseteq \tilde{\mathcal{P}}(\lambda_2).$$

A RMAB problem is indexable if all n arms are indexable.

Definition 3 (Whittle index)**.** For an indexable arm, the Whittle index of the state z of an arm is the smallest value of λ for which state z is part of $\tilde{\mathcal{P}}(\lambda)$, i.e.,

$$\tilde{w}(z) = \inf \left\{ \lambda \in \mathbb{R} : z \in \tilde{\mathcal{P}}(\lambda) \right\}. \quad (6)$$

Equivalently, the Whittle index $\tilde{w}(z)$ is the smallest value of λ for which $\tilde{\pi}_\lambda$ is indifferent between the active action and passive action when the arm is in state z.

The Whittle index policy is as follows: *At each time, compute the Whittle indices of the current state of all arms and select the arms in states with M highest Whittle indices (provided they are positive).*

IV. Indexability of the Assistance Problem

Problem 1 can be formulated as an instance of RMAB, where each robot corresponds to an arm. Under such a formulation, the state $\tilde{Z}_t^k$ of arm k corresponds to operating state $x_t^k = (n_t^k, s_t^k)$ of robot k. The transition function $\tilde{T}^k$ corresponds to the robot state evolution shown in Fig. 2 and the cost function $\tilde{C}^k$ corresponds to the associated per-step cost C^k. In addition, allocating an operator to robot corresponds to choosing the active action for that arm while autonomous operation corresponds to choosing the passive action. This motivates using the Whittle index policy to solve Problem 1. However, before we can implement this approach, we must check for indexability of the problem. As discussed earlier, there is no universal framework to verify indexability of a problem. Moreover, the optimal policy for the given problem does not show any threshold-based behaviour. Therefore, we determine sufficient conditions for indexability from first principles by using properties of the value function of each individual arm.

Since indexability has to be checked for each arm separately, for this analysis, we drop the superscript k from all variables.

Let $V_\lambda : \mathcal{X} \to \mathbb{R}$ be the unique fixed point of the following equation

$$V_\lambda(x) = \min_{a \in \{0,1\}} Q_\lambda(x, a),$$

where

$$Q_\lambda(x, a) = C(x, a) + \lambda a + \gamma \sum_{x' \in \mathcal{X}} T(x'|x, a) V_\lambda(x'), \quad (7)$$

represents the Q-value of taking action a in state x. Here the transition function $T(x'|x, a)$ denotes the probability of transition from state x to state x' under action a and is represented by Fig. 2. Let $\pi_\lambda : \mathcal{X} \to \{0, 1\}$ be the corresponding optimal policy

$$\pi_\lambda(x) = \arg \min_{a \in \{0,1\}} Q_\lambda(x, a).$$

To ensure uniqueness of the $\arg\min$, we follow the convention that when $Q_\lambda(x, 0) = Q_\lambda(x, 1)$, the passive action $a = 0$ is chosen. Let $\mathcal{P}(\lambda)$ be the passive set given penalty λ and $w(x)$ be the Whittle index of state x for the problem of operator allocation in a single-robot system. Furthermore, define the *benefit function* as

$$B_\lambda(x) = Q_\lambda(x, 1) - Q_\lambda(x, 0). \quad (8)$$

Then, a sufficient condition for indexability is as follows:

Lemma 1. A sufficient condition for Problem 1 to be indexable is that the benefit function $B_\lambda(x)$ for each robot is monotonically increasing in λ for all states $x \in \mathcal{X}$.

Proof. The result follows from the observation that using (8) and Def. 1, we can re-write the passive set as

$$\mathcal{P}(\lambda) = \{x \in \mathcal{X} : B_\lambda(x) \ge 0\}. \quad (9)$$

Thus, monotonicity of the benefit function $B_\lambda(x)$ implies that the condition for indexability given in Def. 2 is satisfied. $\square$

We verify the monotonicity of $B_\lambda(x)$ by finding bounds on the value function and establish the following:

Theorem 1. Let $r_{ns}^a \triangleq 1 - p_{ns}^a - q_{ns}^a$ denote the probability of repeating a task n under mode a with internal state s. Define $\alpha_1(n)$ and $\beta_0(n)$ as follows:

$$\alpha_1(n) = 1 + \frac{\gamma q_{n0}^1}{1 - \gamma r_{n1}^1} + \frac{\gamma q_{n0}^0 \left(\gamma r_{n0}^1 + \dfrac{\gamma^2 q_{n0}^1 q_{n1}^1}{1 - \gamma r_{n1}^1} - 1 \right)}{1 - \gamma r_{n1}^1 - \gamma r_{n0}^0 + \gamma^2 r_{n1}^1 r_{n0}^0 - \gamma^2 q_{n0}^0 q_{n1}^1},$$

and

$$\beta_0(n) = \frac{\gamma (p_{n0}^1 - p_{n0}^0) + \gamma^2 (p_{n0}^0 r_{n0}^1 - p_{n0}^1 r_{n0}^0)}{1 - \gamma r_{n0}^0}.$$

Then, the single-robot problem is indexable if for all $n \in \{1, 2, \dots, N\}$:

$$\alpha_1(n) \ge 0 \quad \text{and} \quad \frac{\beta_0(n)}{1 - \gamma} \ge -1. \quad (10)$$

Proof. See Appendix. $\square$

The multi-robot problem is indexable if the conditions given in Theorem 1 hold true for all robots. In the next section, we present specific instances of the general model described in Section II which are indexable and discuss their relevance in practical assistance problems for (semi)autonomous delivery robots.

V. Special Cases: Robot transitions in the city

This section presents two specific classes of transition functions which represent two types of faults commonly occurring in systems with remote navigating robots.

A. Transition Type-1 : Faults with continuation

Consider the following transition behaviour along a robot's waypoints. At each time step, the robot moves to its next waypoint with a probability representing, for example, the crowd in the area. There is also a probability of getting into a fault state such as encountering an unidentifiable obstacle. A human operator can teleoperate the robot to its next waypoint both from a normal or fault state. Such transitions represent faults where the robot is functioning properly but is unsure about how to proceed due to uncertainty in its surroundings. Thus the probability of success when being teleoperated is the same regardless of whether the robot is in its normal state or stopped in the fault state, i.e., $p^1_{n0} = p^1_{n1}$ and $q^1_{n0} = q^1_{n1} = 0$. The corresponding transition dynamics are shown in Figure 3. Note that in this case $r^1_{n0} = r^1_{n1} = 1 - p^1_{n0}$.

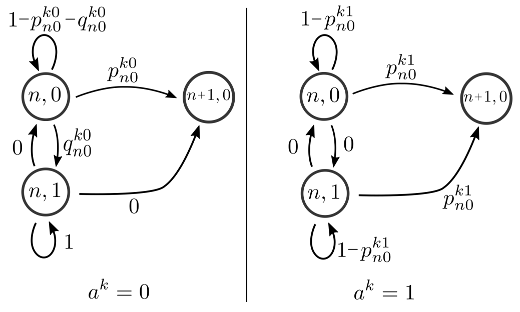

Fig. 3. State-transition probabilities under the autonomous operation and teleoperation for type-1 transitions.

In this case, the coefficients $\alpha_1(n)$ and $\beta_0(n)$ can be simplified to the following expressions:

$$\alpha_1(n) = 1 - \frac{\gamma q^0_{n0}}{1-\gamma r^0_{n0}},$$
$$\beta_0(n) = \frac{\gamma(1-\gamma)(r^0_{n0} - r^1_{n0}) + \gamma q^0_{n0}(1-\gamma\, r^1_{n0})}{1-\gamma\, r^0_{n0}}. \quad (11)$$

Note that

$$\alpha_1(n) = \frac{1-\gamma+\gamma p^0_{n0}}{1-\gamma r^0_{n0}} \geq 0,$$
$$\frac{\beta_0(n)}{1-\gamma} + 1 \geq \frac{\gamma(r^0_{n0} - r^1_{n1})}{1-\gamma r^0_{n0}} + 1 = \frac{1-\gamma r^1_{n1}}{1-\gamma r^0_{n0}} \geq 0.$$

Thus, $\alpha_1(n)$ and $\beta_0(n)$ satisfy the sufficient condition of Theorem 1 for all allowed values of transition probabilities and the discount factor γ. Therefore, any robot following the Type-1 transitions is indexable.

B. Transition Type-2 : Faults with reset

Consider another type of transition where the robot can get into a fault state and needs error fixing while staying at its next waypoint. This includes scenarios such as losing localization or getting stuck in a minor obstacle. The human operator can try to assist the robot out of that situation by fixing the fault, resetting it back to its current waypoint (assuming the system is equipped with means to do so). Such transitions will mean that the probabilities $q^1_{n0} = p^1_{n1} = 0$ and the corresponding transition dynamics are shown in Fig.4:

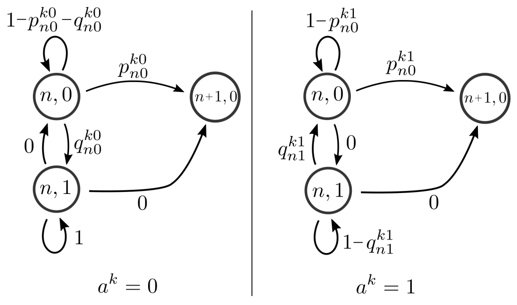

Fig. 4. State-transition probabilities under the autonomous and assist/teleoperate actions for Type-2 transition dynamics.

Substituting the values of transition probabilities from Fig. 4 to the expressions of $\alpha_1(n)$ and $\beta_0(n)$, the coefficients can be simplified to the following:

$$\alpha_1(n) = 1 - \frac{\gamma q^0_{n0}(1-\gamma r^1_{n0})}{(1-\gamma r^0_{n0})(1-\gamma(1-q^1_{n1})) - \gamma^2 q^0_{n0} q^1_{n1}},$$
$$\beta_0(n) = \frac{\gamma(1-\gamma)(r^0_{n0} - r^1_{n0}) + \gamma q^0_{n0}(1-\gamma\, r^1_{n0})}{1-\gamma\, r^0_{n0}}. \quad (12)$$

Note that $\beta_0(n)$ here is the same as (11) and, therefore, satisfies (10). For $\alpha_1(n)$ to satisfy (10), the condition $\alpha_1(n) \geq 0$ results in the following condition on q^1_{n1}:

$$q^1_{n1} \geq 1 - \frac{1}{\gamma} + \frac{\gamma q^0_{n0} p^1_{n0}}{1-\gamma r^0_{n0} - \gamma q^0_{n0}}. \quad (13)$$

As $q^1_{n1} \leq 1$, (13) also yields the following condition on q^0_{n0}:

$$q^0_{n0} \leq \frac{1-\gamma r^0_{n0}}{\gamma(1+\gamma p^1_{n0})}. \quad (14)$$

Therefore, any robot state following the Type-2 transitions will satisfy the condition of indexability in Theorem 1 if (13) and (14) are satisfied, i.e., the probability q^0_{n0} that the robot transitions from a normal state to fault state during autonomous operation is not too high and the probability q^1_{n1} that the operator brings the robot from a fault state to a normal state is not too small.

As an example, consider a robot following Type-2 transitions with $p^0_{n0} = q^0_{n0} = p^1_{n0} = 0.3$ and $\gamma = 0.95$. In this setting, any $q^0_{n0} \in [0,1]$ satisfies (14) and any $q^1_{n1} \in [0.1462, 1]$ satisfies (13). Thus, the model is indexable if there is at least a 14.62% chance that teleoperation successfully resets the robot from the fault state to a normal state.

VI. Computation of Whittle Index

As discussed in Section I-A, once the indexability of the problem instance has been verified, we can compute Whittle indices for all robots and determine the Whittle index policy.

General approaches of computing Whittle indices are either based on adaptive greedy algorithm [33], [39] or binary search [37]. In this section, we briefly provide details on adaptive greedy algorithm and describe how the Whittle index policy works. Readers are encouraged to refer to [33] for a detailed explanation and validation of the algorithm. The algorithm is presented in Alg. 1 for computing Whittle indices for a single robot.

Algorithm 1: Adaptive Greedy Algorithm for Whittle Index Computation

```
1: Input: Robot $(\mathcal{X}, \mathcal{A}, T, C, \gamma, x_0)$.
2: Initialize $\mathcal{P} = \emptyset$.
3: while $\mathcal{P} \neq \mathcal{X}$ do
4:     Compute $\mu_y^*$, $\forall y \in \mathcal{X} \backslash \mathcal{P}$ using Eq. (15).
5:     $\lambda^* \leftarrow \min_{y \in \mathcal{X} \backslash \mathcal{P}} \ \mu_y^*$
6:     $\mathcal{Y} \leftarrow \arg\min_{y \in \mathcal{X} \backslash \mathcal{P}} \ \mu_y^*$
7:     $w(y) \leftarrow \lambda^*$, $\forall y \in \mathcal{Y}$
8:     $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{Y}$
9: end while
```

The algorithm operates as follows: For any subset $\mathcal{Z} \subseteq \mathcal{X}$, define the policy vector[2] $\pi^{(\mathcal{Z})} : \mathcal{X} \to \{0, 1\}$ as

$$\pi^{(\mathcal{Z})}(x) = \begin{cases} 0, & \text{if } x \in \mathcal{Z} \\ 1, & \text{if } x \in \mathcal{X} \backslash \mathcal{Z}. \end{cases}$$

Also define, $C_\pi = \big[C(x, \pi(x))\big]_{x \in \mathcal{X}}$, the cost vector for all states under a policy π, and $T_\pi = \big[T(x'|x, \pi(x))\big]_{x, x' \in \mathcal{X}}$, the transition matrix under policy π.

Then, in each iteration of the while loop, compute μ_y^* as follows:

$$\mu_y(x) = -\frac{D_{\pi^{(\mathcal{P})}}(x) - D_{\pi^{(\mathcal{P} \cup \{y\})}}(x)}{N_{\pi^{(\mathcal{P})}}(x) - N_{\pi^{(\mathcal{P} \cup \{y\})}}(x)}, \ \forall x \in \mathcal{X}$$
$$\mu_y^* = \min_{x \in \Lambda_y} \mu_y(x), \tag{15}$$

where

$$D_\pi(x) = \big[(I - \gamma T_\pi)^{-1} C_\pi\big](x) \ ,$$
$$N_\pi(x) = \big[(I - \gamma T_\pi)^{-1} \pi\big](x) \ ,$$
$$\Lambda_y = \{s \in \mathcal{X} : N_{\pi^{(\mathcal{P})}}(x) - N_{\pi^{(\mathcal{P} \cup \{y\})}}(x) \neq 0\}.$$

The minimum value of μ_y^* calculated in line 5 in Alg. 1 corresponds to the Whittle indices of the minimizing states (Line 6). These states are then taken out of consideration in the next iteration of the while loop by including them in the passive set $\mathcal{P}$. When the Alg. 1 exits the while loop, the Whittle indices for all states of that robot are calculated. This procedure is then repeated for all the robots in the system.

Once the Whittle indices for all states of all robots are obtained, the Whittle index policy can be implemented as given in Alg. 2. In line 2 of the algorithm, the function $\texttt{arg\_top\_M}(\{w^k(x^k)\})$ returns indices of top M positive elements in a set, where ties are broken randomly. As determined in [33], the computational complexity of this method is $\mathcal{O}(K|\mathcal{X}|^3)$. In contrast, the computational complexity of finding the optimal policy for Problem 1 is $\mathcal{O}(\binom{K}{M}|\mathcal{X}|^{2K})$ using value iteration, where $|\mathcal{X}|$ is the size of state space of individual robot.

[2]In the following expressions π is used as a vector of size $|\mathcal{X}|$, constructed as a mapping from each state to corresponding action $a \in \{0, 1\}$.

Algorithm 2: Whittle Index Policy π^I

```
1: Input: Set of Whittle indices $w^k(x^k)$ for all
   $k \in \{1, \ldots, K\}$ and $x^k \in \mathcal{S}^k$, No. of Operators $M$
2: $\mathcal{M} \leftarrow \texttt{arg\_top\_M}(\{w^k(x^k)\})$
3: $a^k \leftarrow 0$ for all $k \notin \mathcal{M}$
4: $a^k \leftarrow 1$ for all $k \in \mathcal{M}$        // Allocate operators
5: return $(a^1, \ldots, a^K)$
```

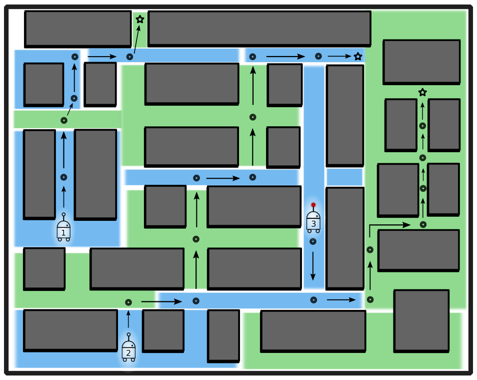

Fig. 5. An instance of multi-robot assistance problem for robots navigating in a city block-like environment. Transition zones are marked by different color shadings, representing type-1 and type-2 transitions, as described in Section V. Three robots are shown navigating to their corresponding goal locations, via a sequence of waypoints shown as black circles. These waypoints may lie in different transition zones resulting in varied performance for the robots.

VII. Simulations and Results

In this section, we present performance results for a simulated multi-robot assistance problem under the following policies (described later): 1) Optimal policy, 2) Index policy, 3) Benefit maximizing Policy, 4) Myopic Policy, and 5) Reactive Policy. The problem and the solution frameworks for all policies were implemented using POMDPs.jl library in Julia [40].

A. Simulation Setup

For the simulations, a city map is generated as shown in Fig. 5 where the map is randomly divided into different zones corresponding to one of the two transition types presented in Section V.

The exact values of transition probabilities at different locations in the map are sampled randomly from a uniform distribution, according to Table I. The bounds on transition probabilities $\bar{q}_{n1}^{k1}$ and $\bar{q}_{n0}^{k0}$ for transition type-2 are determined by (13) and (14) respectively.

For the teleoperation problem, we use the following cost structure:

$$C^k\left((n, s), a\right) = \begin{cases} 0 & \text{if } n = G \\ \rho_n^k & \text{if } a = 0, s = 0 \\ \phi_n^k & \text{if } a = 0, s = 1 \\ \rho_n^k + \rho_T^k & \text{if } a = 1, s = 0 \\ \phi_n^k + \rho_T & \text{if } a = 1, s = 1, \end{cases} \tag{16}$$

with $\rho_n^k, \phi_n^k, \rho_T^k \in \mathbb{R}_{\geq 0}$ for any $n \in \{1, \ldots, N^k\}$. This cost function captures the time that a robot takes to reach its

TABLE I
PROBABILITIES VALUES USED FOR SIMULATIONS FOR THE TWO TYPES OF TRANSITIONS.

Probability	Type-1	Type-2
r_{n0}^{k0}	$[0.2, 0.5]$	$[0.2, 0.5]$
q_{n0}^{k0}	$[0.2, 0.5]$	$[0.1, \min\{\overline{q}_{n0}^{k0}, 1 - r_{n0}^{k0}\}]$
r_{n0}^{k1}	$[0.1, 0.4]$	$[0.1, 0.4]$
q_{n0}^{k1}	0.0	0.0
r_{n1}^{k1}	r_{n0}^{k1}	$1 - q_{n1}^{k1}$
q_{n1}^{k1}	0.0	$[\max\{\overline{q}_{n1}^{k1}, 0.1\}, 0.9]$

goal, i.e., zero cost on reaching goal, non-negative costs for intermediate states, and an additional cost while being assisted.

Values of the different costs and the discount factor used are sampled from ranges specified in Table II.

TABLE II
PARAMETER VALUES USED IN THE SIMULATED ASSISTANCE TASK.

Parameter	ρ_T	ρ	ρ_e	γ
Value	0.75	2.0	4.0	0.99

For simplicity, for a given parameter, same range is used for every state of all robots. Therefore, we have removed subscript k from the notation.

Separate tests were performed to test the validity, performance and scalability of the Index policy. At the beginning of each simulation, a number of robots are placed on the map (ranging from 1 to 25) with randomly generated start and goal locations, and 7 waypoints uniformly placed between the two. In practice, these waypoints are generated by a separate robot path planner for each individual robot, and are considered as an input for the operator allocation problem.

B. Baseline Policies

We consider the following baseline policies to assess the performance of the Index policy.

Optimal policy: The Optimal policy $\pi^* : \boldsymbol{\mathcal{X}} \to \mathcal{A}$, as defined by (2), is found by encoding the complete problem with all robots as an MDP and solving it using the *Sparse Value Iteration Solver* from the POMDP.jl library.

Reactive policy: The reactive policy allocates an operator to any robot stuck in a fault state. If there are more such robots than operators, a random subset of those robots is selected.

Myopic policy: Myopic/Greedy Policies are commonly used to obtain fast (but sub-optimal) solutions to intractable problems. In this paper, we implement an l-step myopic policy presented in [3] for $l \in \{1, 2\}$. Define $V^{\mathbf{0}}(\boldsymbol{x}_{t+1})$ as the expected cost incurred by the system from current time step to infinity under passive actions. The l-step myopic policy $\pi^{G\text{-}l} : \boldsymbol{\mathcal{X}} \to \mathcal{A}$ is then defined as

$$\pi^{G\text{-}l}(\boldsymbol{x}_t) = \arg\min_{\boldsymbol{a} \in \mathcal{A}} \; g(\boldsymbol{x}_t, \boldsymbol{a}, l), \tag{17}$$

where the l-step look-ahead cost $g(\boldsymbol{x}_t, \boldsymbol{a}, l)$ is given by

$$g(\boldsymbol{x}_t, \boldsymbol{a}, l) = \begin{cases} C\left(\boldsymbol{x}_t, \boldsymbol{a}\right) + \\ \sum\limits_{\boldsymbol{x}_{t+1}} \gamma\, T(\boldsymbol{x}_{t+1} | \boldsymbol{x}_t, \boldsymbol{a})\, g(\boldsymbol{x}_t, \boldsymbol{a}, l-1), & \text{if } l = 0, \\ V^{\mathbf{0}}(\boldsymbol{x}_{t+1}), & \text{otherwise,} \end{cases}$$

where

$$C\left(\boldsymbol{x}, \boldsymbol{a}\right) = \sum_{k=1}^{K} C^k\left(x^k, a^k\right),$$

is the cost incurred in the current time step.

Benefit maximizing policy: For comparison, we also propose a heuristic policy which we call *benefit maximizing policy* that tries to exploit the independence of the robots' transitions. This policy is inspired by the *advantage function* used in reinforcement learning (for example, see [41]). The policy considers the *benefit* or *advantage* of taking the active action over the passive action for each robot and picks the robots with highest benefit at each time step, i.e.,

$$\pi^B(\boldsymbol{x}_t) = \arg\min_{\boldsymbol{a} \in \mathcal{A}} \sum_{k=1}^{K} a^k \; B_0(x_t^k), \tag{18}$$

where $B_0(x)$ corresponds to $B_\lambda(s)$ defined in (8) with $\lambda = 0$.

C. Comparison with the Optimal policy

First, the Index policy is compared against the Optimal policy to validate its applicability for our problem. Due to its poor scalability, the Optimal policy cannot be computed for larger problem instances, therefore this test is limited to smaller problem size (up to 4 robots and 2 operators). The relative performance (ratio of the cost incurred under Index policy to that under Optimal policy) is shown in Fig. 6. For comparison, 100 problem instances were tested under both policies and were simulated through Monte Carlo rollouts. Each problem instance is run until all robots reach their respective goal locations. This is repeated for 10^6 iterations and average cost is recorded.

It is observed that the Index policy performs quite close to the optimal policy for all test cases. As the ratio of number of robots to number of operators increases, the Index policy starts to degrade in comparison to the optimal. However, the relative cost for most cases still remains within 5% of the optimal.

D. Comparison with other baseline policies

Next, we compare the performance (measured as average cost incurred per robot before reaching its goal) of the Index policy with the three baseline policies on larger problem instances. For the comparison, a set of 100 problem instances is created, each with a set of 7 waypoints with randomly sampled transition probabilities according to Table I. Each instance of the problem is then simulated separately under the different policies using Monte Carlo rollouts until all robots reach their goal states, repeated for 500 iterations. Each simulation iteration (rollout) is timed out at 10 seconds for each policy. If an iteration takes longer than this time, the simulations are interrupted and the policy's result for that test condition is not reported.

Figure 7 shows performance comparison of the four policies. The Index policy performs best out of the four policies, followed by the benefit maximizing policy (π^B) and the myopic policy (π^G). The Reactive policy performed the worst as expected. As a side note, the average cost incurred per

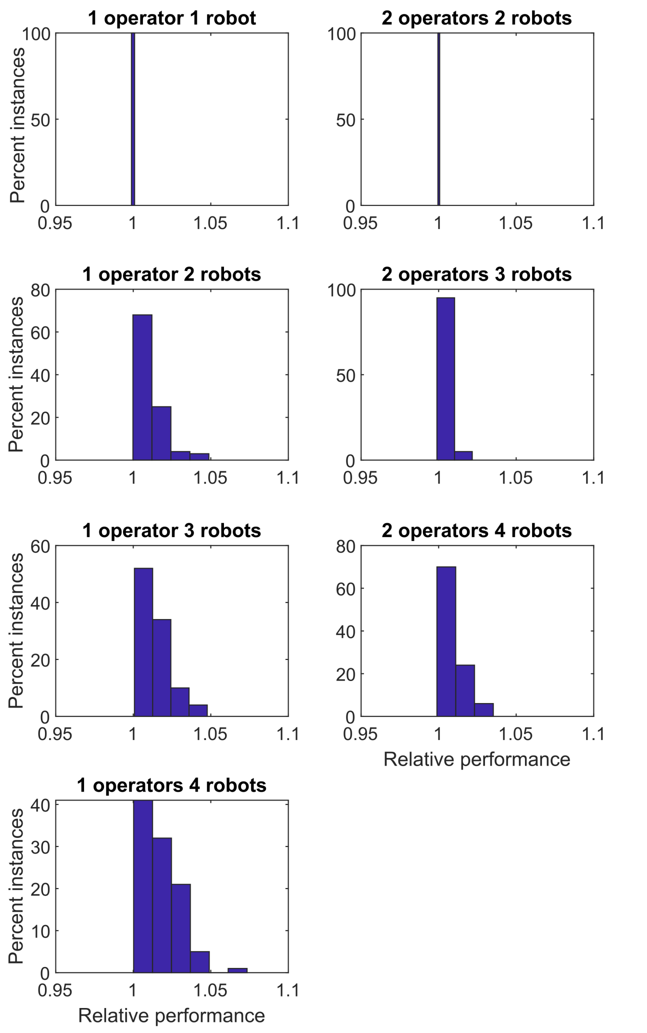

Fig. 6. Relative performance of Index policy compared to the optimal policy. The plots show distribution of $\mathbf{100}$ indexable problem instances based on their performance under the two policies. Relative performance is calculated as the ratio of the average cost incurred under Index policy to that under Optimal policy.

robot under any policy is strongly correlated with the ratio of number of robots to the number of available operators. This observation supports the intuition that as human operators are required to distribute their assistance among more robots, their effectiveness decreases.

E. Scalability

Table III shows the time that each policy takes to compute operator the allocation under different problem sizes. For these simulations, each robot is set to have 7 waypoints. As observed from the table, the computation times of the two Myopic policies scale exponentially with both the number of robots and the number of operators, with the time for 2-step Myopic policy growing at a much higher rate.

The computation times for the index and benefit maximizing policies scale linearly with the number of robots and are independent of the number of operators. Also, note that the Whittle index computation for one robot is independent from the rest. Therefore robots can be added/removed without re-computation of already computed indices. Furthermore, if the number of operators changes to $M > 1$, the policy simply allocates operators to the robots with the M highest Whittle indices. As a result, the policy is efficiently scalable with the number of robots and operators. For reference, the simulations were run on a Desktop PC with a 4 core, 4.20 GHz processor and 32 GB of RAM.

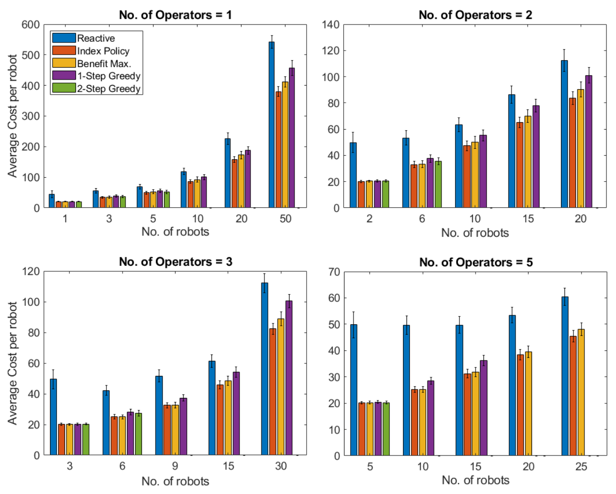

Fig. 7. Performance comparison of the four policies for different number of operators available for allocation. Error bars in the plots show one standard-deviation above and below the average. Note that in larger problem instances, the simulations for myopic policies timed out and could not be completed in the specified time limit (10 sec per rollout).

TABLE III
COMPUTATION TIMES OF DIFFERENT POLICIES (SECONDS)

Operators/ Robots	Index Policy	1-step Myopic	2-step Myopic	Benefit Maximizing
2/6	$1.2e{-}6$	$3.8e{-}3$	$2.4e{-}1$	$3.4e{-}6$
3/6	$1.2e{-}6$	$6.2e{-}3$	$4.8e{-}1$	$3.4e{-}6$
4/6	$1.2e{-}6$	$8.1e{-}3$	$6.6e{-}1$	$3.4e{-}6$
1/9	$1.7e{-}6$	$9.6e{-}2$	$4.8e{+}0$	$4.9e{-}6$
2/9	$1.7e{-}6$	$2.6e{-}1$	$2.2e{+}1$	$4.9e{-}6$
3/9	$1.7e{-}6$	$6.4e{-}1$	$6.2e{+}1$	$4.9e{-}6$

VIII. CONCLUSIONS AND DISCUSSION

In this paper, we provide an analysis of operator allocation problem for a multi-robot assistance task and demonstrate the effectiveness of Restless Bandit framework to obtain a scalable policy. This policy is based on Whittle index heuristic and performs close to the optimal and significantly better than other efficient solution approaches. We also provide an analysis of indexability of such problems and give a simplified condition to quickly verify if a problem instance is indexable. These results can also be used to specify required transition behavior in the form of bounds on transition probabilities.

There are, however, a few limitations of the proposed approach. When a problem instance is not indexable, the Whittle indices are not defined and the methods of computing these indices may not give meaningful values. Therefore, the proposed approach is not applicable in such cases. Also, note that the conditions for indexability identified in Theorem 1 are sufficient but not necessary. However, the assistance problem presented here is indexable in most instances. This was verified by randomly generating problem instances without any bounds and numerically verifying monotonicity of the passive set $\mathcal{P}(\lambda)$. Out of the 1000 random instances, $999, 992$ and 940 instances were found to be indexable for discount factors $\gamma = 0.9, 0.95, 0.99$ respectively. However, the sufficient conditions were satisfied for $700, 607$ and 452 instances for $\gamma = 0.9, 0.95, 0.99$. This suggests that model is, in general, indexable and the Whittle index heuristic is applicable. It also suggests that the system may benefit from improved sufficient conditions for indexability. It is worth noting that this analysis can provide us with class of transitions for which there is no need to check the conditions. For instance, in case of transition type-1, the sufficient conditions are satisfied for all values of transitions probabilities and thus the requirements for indexability are always met.

There are several research directions for future work that can lead to a richer modelling of such operator allocation problems. Incorporating the notion of uncertain transition probabilities instead of assuming the knowledge of exact values will result in a more robust model. The probabilities can be estimated using recorded data [2], [3] or, by modelling performance parameters such as operators' expertise [42]. Shifting the system definition from discrete to continuous space (or just adding the time dimension to the actions) can represent a more practical scenario. Overall, this paper presents a starting point to a wide variety of human-robot collaborative systems with multiple agents and provides a promising framework to solve large instances of such problems.

References

[1] Y. Wang, Z. Shi, C. Wang, and F. Zhang, "Human-robot mutual trust in (semi) autonomous underwater robots," in *Cooperative Robots and Sensor Networks 2014*. Springer, 2014, pp. 115–137.

[2] G. Swamy, S. Reddy, S. Levine, and A. D. Dragan, "Scaled autonomy: Enabling human operators to control robot fleets," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5942–5948.

[3] A. Rosenfeld, N. Agmon, O. Maksimov, and S. Kraus, "Intelligent agent supporting human–multi-robot team collaboration," *Artificial Intelligence*, vol. 252, pp. 211–231, 2017.

[4] A. Khasawneh, H. Rogers, J. Bertrand, K. C. Madathil, and A. Gramopadhye, "Human adaptation to latency in teleoperated multi-robot human-agent search and rescue teams," *Automation in Construction*, vol. 99, pp. 265–277, 2019.

[5] K. Zheng, D. F. Glas, T. Kanda, H. Ishiguro, and N. Hagita, "Supervisory control of multiple social robots for navigation," in *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2013, pp. 17–24.

[6] J. Y. Chen and M. J. Barnes, "Supervisory control of multiple robots: Effects of imperfect automation and individual differences," *Human Factors*, vol. 54, no. 2, pp. 157–174, 2012.

[7] D. R. Olsen Jr and S. B. Wood, "Fan-out: measuring human control of multiple robots," in *SIGCHI conference on Human factors in computing systems*, 2004, pp. 231–238.

[8] S. Y. Chien, M. Lewis, S. Mehrotra, and K. Sycara, "Imperfect automation in scheduling operator attention on control of multi-robots," in *Human Factors and Ergonomics Society Annual Meeting*, vol. 57, no. 1. SAGE Publications Sage CA: Los Angeles, CA, 2013, pp. 1169–1173.

[9] J. Y. Chen and M. J. Barnes, "Human–agent teaming for multirobot control: A review of human factors issues," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 1, pp. 13–29, 2014.

[10] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[11] M. L. Littman, T. L. Dean, and L. P. Kaelbling, "On the complexity of solving markov decision problems," in *Eleventh Annual Conference on Uncertainty in Artificial Intelligence*, 1995, pp. 394–402.

[12] S. A. Zanlongo, P. Dirksmeier, P. Long, T. Padir, and L. Bobadilla, "Scheduling and path-planning for operator oversight of multiple robots," *Robotics*, vol. 10, no. 2, p. 57, 2021.

[13] S. K. K. Hari, A. Nayak, and S. Rathinam, "An approximation algorithm for a task allocation, sequencing and scheduling problem involving a human-robot team," *Robotics and Automation Letters*, vol. 5, no. 2, pp. 2146–2153, 2020.

[14] A. Dahiya, N. Akbarzadeh, A. Mahajan, and S. L. Smith, "Scalable operator allocation for multi-robot assistance: A restless bandit approach," *arXiv preprint arXiv:2111.06437*, 2021.

[15] J. R. Peters, V. Srivastava, G. S. Taylor, A. Surana, M. P. Eckstein, and F. Bullo, "Human supervisory control of robotic teams: Integrating cognitive modeling with engineering design," *IEEE Control Systems Magazine*, vol. 35, no. 6, pp. 57–80, 2015.

[16] J. W. Crandall, M. L. Cummings, M. Della Penna, and P. M. De Jong, "Computing the effects of operator attention allocation in human control of multiple robots," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 41, no. 3, pp. 385–397, 2010.

[17] S. Musić and S. Hirche, "Control sharing in human-robot team interaction," *Annual Reviews in Control*, vol. 44, pp. 342–354, 2017.

[18] M. B. Dias, B. Kannan, B. Browning, E. Jones, B. Argall, M. F. Dias, M. Zinck, M. Veloso, and A. Stentz, "Sliding autonomy for peer-to-peer human-robot teams," in *International conference on intelligent autonomous systems*, 2008, pp. 332–341.

[19] D. Szafir, B. Mutlu, and T. Fong, "Designing planning and control interfaces to support user collaboration with flying robots," *The International Journal of Robotics Research*, vol. 36, no. 5-7, pp. 514–542, 2017.

[20] E. A. Kirchner, S. K. Kim, M. Tabie, H. Wöhrle, M. Maurus, and F. Kirchner, "An intelligent man-machine interface—multi-robot control adapted for task engagement based on single-trial detectability of p300," *Frontiers in human neuroscience*, vol. 10, p. 291, 2016.

[21] S. Y. Chien, Y. L. Lin, P. J. Lee, S. Han, M. Lewis, and K. Sycara, "Attention allocation for human multi-robot control: Cognitive analysis based on behavior data and hidden states," *International Journal of Human-Computer Studies*, vol. 117, pp. 30–44, 2018.

[22] M. M. Raeissi, N. Brooks, and A. Farinelli, "A balking queue approach for modeling human-multi-robot interaction for water monitoring," in *International Conference on Principles and Practice of Multi-Agent Systems*. Springer, 2017, pp. 212–223.

[23] J. Berger, N. Lo, and M. Noel, "A new multi-target, multi-agent search-and-rescue path planning approach," *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, vol. 8, no. 6, pp. 935–944, 2014.

[24] H. Pei, S. Chen, and Q. Lai, "Multi-target consensus circle pursuit for multi-agent systems via a distributed multi-flocking method," *International Journal of Systems Science*, vol. 47, no. 16, pp. 3741–3748, 2016.

[25] P. Whittle, "Restless bandits: Activity allocation in a changing world," *Journal of applied probability*, vol. 25, no. A, pp. 287–298, 1988.

[26] J. C. Gittins, "Bandit processes and dynamic allocation indices," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 41, no. 2, pp. 148–164, 1979.

[27] L. Chan, D. Hadfield-Menell, S. Srinivasa, and A. Dragan, "The assistive multi-armed bandit," in *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2019, pp. 354–363.

[28] H. Claure, Y. Chen, J. Modi, M. Jung, and S. Nikolaidis, "Multi-armed bandits with fairness constraints for distributing resources to human teammates," in *2020 ACM/IEEE International Conference on Human-Robot Interaction*, 2020, pp. 299–308.

[29] A. Mahajan and D. Teneketzis, "Multi-armed bandit problems," in *Foundations and applications of sensor management*. Springer, 2008, pp. 121–151.

[30] 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.

[31] G. Xiong, R. Singh, and J. Li, "Learning augmented index policy for optimal service placement at the network edge," *arXiv preprint arXiv:2101.03641*, 2021.

[32] V. S. Borkar, G. S. Kasbekar, S. Pattathil, and P. Y. Shetty, "Opportunistic scheduling as restless bandits," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 4, pp. 1952–1961, 2017.

[33] N. Akbarzadeh and A. Mahajan, "Conditions for indexability of restless bandits and an algorithm to compute Whittle index," 2021.

[34] S. Wu, K. Ding, P. Cheng, and L. Shi, "Optimal scheduling of multiple sensors over lossy and bandwidth limited channels," *IEEE Transactions on Control of Network Systems*, vol. 7, no. 3, pp. 1188–1200, 2020.

[35] K. Avrachenkov, U. Ayesta, J. Doncel, and P. Jacko, "Congestion control of TCP flows in internet routers by means of index policy," *Computer Networks*, vol. 57, no. 17, pp. 3463–3478, 2013.

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

[37] N. Akbarzadeh and A. Mahajan, "Dynamic spectrum access under partial observations: A restless bandit approach," in *Canadian Workshop on Information Theory*. IEEE, 2019, pp. 1–6.

[38] N. Akbarzadeh and A. Mahajan, "Restless bandits: Indexability and computation of whittle index," *arXiv preprint arXiv:2008.06111*, 2020.

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

[40] M. Egorov, Z. N. Sunberg, E. Balaban, T. A. Wheeler, J. K. Gupta, and M. J. Kochenderfer, "POMDPs.jl: A framework for sequential decision making under uncertainty," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 831–835, 2017.

[41] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4*, 2016.

[42] P. Carreno-Medrano, A. Dahiya, S. L. Smith, and D. Kulić, "Incremental estimation of users' expertise level," in *International Conference on Robot & Human Interactive Communication (ROMAN)*. IEEE, 2019.

APPENDIX

For any fixed value of λ, the value function $V_\lambda(x)$ can also be written as

$$V_\lambda(x) = \min_{\pi\in\Pi} \mathbb{E}\Big[\sum_{t=0}^{T}\big[C(X_t, A_t) + \lambda A_t\big] | X_0 = x\Big],$$

where Π denotes the set of all Markov policies from $\mathcal{X}$ to $\{0,1\}$. Since the state space $\mathcal{X}$ is finite, so is Π. Thus, $V_\lambda(x)$ is the minimum of a finite number of functions, each of which is linear in λ. Therefore, $V_\lambda(x)$ is continuous and piecewise linear, with a finite number of corner points. This means $V_\lambda(x)$, and therefore $Q_\lambda(x,a)$ and $B_\lambda(x)$, are non differential w.r.t. λ at a finite number of points. Therefore, $B_\lambda(x)$ is monotonically increasing if $\partial B_\lambda(x)/\partial\lambda$, wherever it exists, is non-negative. Let $\Lambda^*(x)$ denote the finite set of values where $B_\lambda(x)$ is non-differentiable. Let $\Lambda^* = \cup_{x\in\mathcal{X}}\Lambda^*(x)$.

The main idea for the proof of Theorem 1 is to show that if (10) is satisfied then $\partial B_\lambda(x)/\partial\lambda$ is non-negative for $\lambda \notin \Lambda^*$. Then, Lemma 1 implies the indexability of the problem.

Now fix an $n \in \{1,\dots,N\}$. Define $z = (n,0)$, $z' = (n+1,0)$ and $e = (n,1)$. Then using Eq. (7) and Fig. 2, we have

$$Q_\lambda(z,a) = C(z,a) + \lambda a + \gamma q^a_{n0} V_\lambda(e) + \gamma p^a_{n0} V_\lambda(z') + \gamma r^a_{n0} V_\lambda(z). \quad (19)$$

$$Q_\lambda(e,a) = C(e,a) + \lambda a + \gamma q^a_{n1} V_\lambda(z) + \gamma p^a_{n1} V_\lambda(z') + \gamma r^a_{n1} V_\lambda(e). \quad (20)$$

Then we have the following results[3]:

Lemma 2. For all $\lambda \notin \Lambda^*$,

$$0 \le \frac{\partial V_\lambda(x)}{\partial\lambda} \le \frac{1}{1-\gamma}, \quad \forall x \in \mathcal{X}.$$

[3]More details are provided in arXiv preprint [14].

Proof. Under an optimal policy π^*, we have:

$$V_\lambda(x) = \mathbb{E}\Big[\sum_{t=0}^{\infty}\gamma^t C_\lambda(X_t, \pi^*(X_t))\Big| X_0 = x\Big].$$

Therefore, we get

$$\frac{\partial V_\lambda(x)}{\partial\lambda} = \frac{\partial}{\partial\lambda}\mathbb{E}\Big[\sum_{t=0}^{\infty}\gamma^t C_\lambda(X_t, \pi^*(X_t))\Big| X_0 = x\Big] = \mathbb{E}\Big[\sum_{t=0}^{\infty}\gamma^t \frac{\partial}{\partial\lambda} C_\lambda(X_t, \pi^*(X_t))\Big| X_0 = x\Big].$$

Since $\frac{\partial}{\partial\lambda}C_\lambda(x,\pi^*(x)) \in [0,1]$ for all $x \in \mathcal{X}$, we can write

$$0 \le \frac{\partial V_\lambda(x)}{\partial\lambda} \le \sum_{t=0}^{\infty}\gamma^t \implies 0 \le \frac{\partial V_\lambda(x)}{\partial\lambda} \le \frac{1}{1-\gamma}.$$

□

Define

$$\alpha_0(n) = 1,$$

$$\beta_0(n) = \frac{\gamma(p^1_{n0} - p^0_{n0}) + \gamma^2(p^0_{n0} r^1_{n0} - p^1_{n0} r^0_{n0})}{1 - \gamma r^0_{n0}},$$

$$\alpha_1(n) = 1 + \frac{\gamma q^1_{n0}}{1-\gamma r^1_{n1}} + \frac{\gamma q^0_{n0}\left(\gamma r^1_{n0} + \dfrac{\gamma^2 q^1_{n0} q^1_{n1}}{1-\gamma r^1_{n1}} - 1\right)}{1 - \gamma r^1_{n1} - \gamma r^0_{n0} + \gamma^2 r^1_{n1} r^0_{n0} - \gamma^2 q^0_{n0} q^1_{n1}},$$

$$\beta_1(n) = \gamma p^1_{n0} + \frac{\gamma^2 q^1_{n0} p^1_{n0}}{1-\gamma r^1_{n1}} + \frac{\left(\gamma p^0_{n0} - \gamma^2 p^0_{n0} r^1_{n1} + \gamma^2 q^0_{n0} p^1_{n0}\right)\left(\gamma r^1_{n0} + \dfrac{\gamma^2 q^1_{n0} q^1_{n1}}{1-\gamma r^1_{n1}} - 1\right)}{1 - \gamma r^1_{n1} - \gamma r^0_{n0} + \gamma^2 r^1_{n1} r^0_{n0} - \gamma^2 q^0_{n0} q^1_{n1}}.$$

Also define $b_{00}(n) = b_{10}(n) = 1$ and

$$b_{01}(n) = \frac{1-\gamma r^0_{n0}}{1-\gamma r^1_{n0}},$$

$$b_{11}(n) = \frac{1 - \gamma r^1_{n1} - \gamma r^0_{n0} + \gamma^2 r^1_{n1} r^0_{n0} - \gamma^2 q^1_{n1} q^0_{n0}}{1 - \gamma r^1_{n1} - \gamma r^1_{n0} + \gamma^2 r^1_{n1} r^1_{n0} - \gamma^2 q^1_{n1} q^1_{n0}}.$$

Lemma 3. Let $\pi_\lambda(z) = i$ and $\pi_\lambda(e) = j$, then for $\lambda \notin \Lambda^*$,

$$\frac{\partial B_\lambda(z)}{\partial\lambda} = b_{ij}(n)\left[\alpha_j(n) + \beta_j(n)\frac{\partial V_\lambda(z')}{\partial\lambda}\right].$$

Proof. Observe that when $i = 0$, from (19), we get that

$$Q_\lambda(z,0) = \frac{C(z,0) + \gamma q^0_{n0} V_\lambda(e) + \gamma p^0_{n0} V_\lambda(z')}{1-\gamma r^0_{n0}}, \quad (21)$$

$$Q_\lambda(z,1) = C(z,1) + \lambda + \gamma q^1_{n0} V_\lambda(e) + \gamma p^1_{n0} V_\lambda(z') + \gamma r^1_{n0} Q_\lambda(z,0). \quad (22)$$

Similarly, when $i = 1$, from (19), we get that

$$Q_\lambda(z,0) = C(z,0) + \gamma q^0_{n0} V_\lambda(e) + \gamma p^0_{n0} V_\lambda(z') + \gamma r^0_{n0} Q_\lambda(z,1), \quad (23)$$

$$Q_\lambda(z,1) = \frac{C(z,1) + \lambda + \gamma q^1_{n0} V_\lambda(e) + \gamma p^1_{n0} V_\lambda(z')}{1-\gamma r^1_{n0}}. \quad (24)$$

Furthermore, when $j = 1$, from (20), we get

$$V_\lambda(e) = \frac{C(e,1) + \lambda + \gamma q^1_{n1} V_\lambda(z) + \gamma p^1_{n1} V_\lambda(z')}{1 - \gamma r^1_{n1}}, \quad (25)$$

and when $j = 0$, we get

$$V_\lambda(e) = \frac{C(e,0)}{1-\gamma}. \quad (26)$$

The result then follows from considering the four cases $(i,j) \in \{(0,0),(0,1),(1,0),(1,1)\}$ separately and simplifying

$$\frac{\partial B_\lambda(z)}{\partial \lambda} = \frac{\partial Q_\lambda(z,1)}{\partial \lambda} - \frac{\partial Q_\lambda(z,0)}{\partial \lambda}. \quad (27)$$

□

Lemma 4. For all $\lambda \notin \Lambda^*$, $\partial B_\lambda(e)/\partial\lambda \geq 0$.

Proof. We consider two cases:

1) Case I: $\pi_\lambda(e) = 0$:
From (20), we have

$$Q_\lambda(e,0) = V_\lambda(e) = \frac{C(e,0)}{1-\gamma},$$

which is independent of λ. Therefore, we get

$$\frac{\partial B_\lambda(e)}{\partial\lambda} = \frac{\partial Q_\lambda(e,1)}{\partial\lambda} = 1 + \gamma\, q^1_{n1}\frac{\partial V_\lambda(z)}{\partial\lambda} + \gamma\, p^1_{n1}\frac{\partial V_\lambda(z')}{\partial\lambda}.$$

From Lemma 2, we know that $\partial V_\lambda(z)/\partial\lambda \geq 0$ and $\partial V_\lambda(z')/\partial\lambda \geq 0$. This gives us $\partial B_\lambda(e)/\partial\lambda \geq 0$.

2) Case II: $\pi_\lambda(e) = 1$:
As a result, we have

$$Q_\lambda(e,0) = C(e,0) + \gamma\, V_\lambda(e).$$

Therefore, using (27), we get

$$\frac{\partial B_\lambda(e)}{\partial\lambda} = (1-\gamma)\frac{\partial V_\lambda(e)}{\partial\lambda}.$$

From Lemma 2, we get that $\partial B_\lambda/\partial\lambda \geq 0$. □

A. Proof of Theorem 1

Lemma 4 shows that the benefit function $B_\lambda(x)$ is always monotonically increasing for all fault states. Since $\gamma \in (0,1)$ and $q^a_{ns} \leq 1 - r^a_{ns}$ for all $a, s \in \{0,1\}$, we can verify that $b_{01}(n)$ and $b_{11}(n)$ are strictly positive (by replacing q^a_{ns} with $1 - r^a_{ns}$ and using the fact that $\gamma - \gamma r^a_{ns} < 1 - \gamma r^a_{ns}$).

Lemma 2 gives us bounds on the derivative of the value function w.r.t. λ. These bounds are then used with results of Lemma 3 to show that the function $B_\lambda(x)$ is monotonically increasing for all $x \in \mathcal{X}$ if for all $n \in \{1,2,\ldots,N\}$ and $j \in \{0,1\}$ [4]:

$$\alpha_j(n) \geq 0 \text{ and } \alpha_j(n) + \beta_j(n)\frac{1}{1-\gamma} \geq 0.$$

We observe that $\alpha_0(n) = 1 \geq 0$. Also, the terms in $\alpha_1(n) + \beta_1(n)/(1-\gamma)$ can be simplified to $1/b_{11}(n)$ by rearranging and cancelling the common terms. Since $b_{11}(n) > 0$, we get $\alpha_1(n) + \beta_1(n)/(1-\gamma) > 0$.

Therefore, the single-robot problem is indexable if:

$$\alpha_1(n) \geq 0 \quad \text{and} \quad \frac{\beta_0(n)}{1-\gamma} \geq -1,\ \forall n \in \{1,\ldots,N\}.$$

□

[4]To obtain this result, we make use of Lemma 2, which establishes bounds on the derivative of the value function. Since transition probabilities at each state are independent, monotonicity of the benefit function B_λ is guaranteed if $\alpha_j(n) + \beta_j(n)\partial V(n+1)/\partial\lambda$ is non-negative for both lower and upper bounds of $\partial V(n+1)/\partial\lambda$.

PLACE PHOTO HERE

Abhinav Dahiya is a PhD student in the Autonomous Systems Lab. in the department of Electrical and Computer Engineering, University of Waterloo, Canada. He received his Bachelor's degree in electrical engineering from Indian Institute of Technology Roorkee, India in 2016. His research interests include control and characterization of multi-agent systems involving human-robot interaction.

PLACE PHOTO HERE

Nima Akbarzadeh (S'17) is a PhD student in the Electrical and Computer Engineering, McGill University, Canada. He received the B.Sc. degree in Electrical and Computer Engineering from Shiraz University, Iran, in 2014, the M.Sc. in Electrical and Electronics Engineering from Bilkent University, Turkey, in 2017. He is a recipient of 2020 FRQNT PhD Scholarship. His research interests include stochastic control, reinforcement learning and multi-armed bandits.

PLACE PHOTO HERE

Aditya Mahajan (S'06-M'09-SM'14) is Associate Professor in the department of Electrical and Computer Engineering, McGill University, Montreal, Canada. He is Associate Editor for IEEE Transactions on Automatic Control and Springer Mathematics of Control, Signal, and Systems. He was an Associate Editor of the IEEE Control Systems Society Conference Editorial Board from 2014 to 2017. He is the recipient of the 2015 George Axelby Outstanding Paper Award, 2014 CDC Best Student Paper Award (as supervisor), and the 2016 NecSys Best Student Paper Award (as supervisor). His principal research interests include learning and control of centralized and decentralized stochastic systems.

PLACE PHOTO HERE

Stephen L. Smith (S'05–M'09–SM'15) is an Associate Professor in the department of Electrical and Computer Engineering, University of Waterloo, Canada where he holds a Canada Research Chair in Autonomous Systems. He is an Associate Editor for the IEEE Transactions on Control of Network Systems, and is the Co-General Chair for the 2021 IEEE International Conference on Robot and Human Interactive Communication (RO-MAN). His main research interests lie in control and optimization for autonomous systems, with an emphasis on robotic motion planning and coordination.