The Impact of Diversity on Optimal Control Policies for Heterogenous Robot Swarms

Amanda Prorok, M. Ani Hsieh, and Vijay Kumar

Abstract—**We present a framework that allows us to quantify the impact of diversity on the performance of a heterogeneous swarm of robots. We model the system of heterogeneous robots as a community of species, where each species (robot type) is defined by the traits (capabilities) that it owns. Specifically, we consider the problem of distributing a large group of heterogeneous robots among a set of tasks that require specialized capabilities in order to be completed. In order to solve this distribution problem, we develop centralized as well as decentralized methods to efficiently control the heterogeneous swarm of robots. The methods are based on a continuous model of the system at a macroscopic level. We solve the system to obtain an optimal set of transition rates for each species, so that the desired trait distribution is reached as quickly as possible. Since the optimization relies on an analytical derivation of the gradient, it is very efficient with respect to state-of-the-art methods. Building on this result, we formulate a control policy that performs the optimization in real-time to continuously adapt the transition rates. Importantly, our framework includes a diversity metric that enables an evaluation of the impact of swarm heterogeneity on performance. The metric defines the notion of *minspecies*, i.e., the minimum set of species that are required to achieve a given goal. We show that two distinct goal functions lead to two specializations of minspecies, which we term as *eigenspecies* and *coverspecies*. Quantitative results show the relation between diversity and performance.**

Index Terms—**Swarm robotics, stochastic systems, task allocation, heterogeneous multi-robot systems.**

I. Introduction

Technological advances in embedded systems, such as component miniaturization and improved efficiency of sensors and actuators, are enabling the deployment of very large-scale robot systems, i.e., robot swarms. However, as we aspire to solve increasingly complex problems, it becomes ever more difficult to embed all necessary capabilities into one single robot type. Our premise is that one single type of robot cannot cater to all aspects of the task at hand, because at the individual level, it is governed by design rules that limit the scope of its capabilities. For example, a larger robot may be able to carry more powerful sensors, but may be less agile than its smaller counterpart. Or, we could consider the limited payload of aerial robots: if a given task requires rich sensory feedback, multiple heterogeneous aerial robots can complement each other by carrying distinct sensors. Instances of information gathering lend themselves naturally to this problem formulation, with applications to search, surveillance, environmental monitoring, and situational awareness [7, 9, 26].

As we allocate distinct capabilities among robot team members, we imply a certain degree of specialization among individuals [1, 2, 11]. During this process, heterogeneity becomes a design feature. The question is, then, how to best design such systems so that the resulting performance is optimized. However, since there has been very little work on quantitative measures of diversity in multi-robot systems, we still lack the analytical tools to understand the implications.

In this work, we contribute towards a general understanding of heterogeneity by proposing a measure that quantifies the diversity of a swarm of robots that is tasked to complete a goal. Our hypothesis is that the diversity measure must be tied to the underlying goal function for it to be meaningful. In particular, we show how for two different goals, we need two different diversity measures. Towards this end, we consider a concrete application with the objective of distributing a swarm of heterogeneous robots as efficiently as possible among tasks that require specialized competences. Our methodology enables the formulation of control policies that take the heterogeneity of the swarm into account explicitly, and that are capable of adapting to changes online.

A. Example of the Redistribution Problem

Figure 1 shows a system comprising ten tasks that can be serviced by means of four distinct traits. The initial trait distribution is shown at t_0, and subsequent desired trait distributions are shown at t_1, t_2, and t_3. Figure 2 shows how the distribution of the traits evolves over time. This sequence is an example of how a heterogeneous robot system can be controlled to complete a global goal composed of several subtasks that require a specific set of capabilities in specific amounts. Also, the example shows how the solution to the redistribution problem can incorporate temporal constraints or precedence constraints: at an operator-level, we can define arbitrary rules that govern transitions from one desired trait distribution to another as a function of the system's performance. As an example application, we could consider a spatially distributed information gathering scenario: if enough data has been gathered at one site, we can reconfigure the system of robots so that it distributes to sites that have not yet been sufficiently accounted for. Which robots are deployed to which sites will depend on their capabilities and how these capabilities meet the needs that are anticipated at the sites. We note that the communication infrastructure required for such an approach is asymmetric: a centralized operator gathers abstract state information about the robot swarm, and relays control inputs back to the robots. This approach ensures algorithmic invariance as we scale the system [16].

*Amanda Prorok and Vijay Kumar are with the University of Pennsylvania, Philadelphia, PA, USA: {prorok | kumar}@seas.upenn.edu. M. Ani Hsieh is with Drexel University, Philadelphia, PA, USA: mhsieh1@drexel.edu.

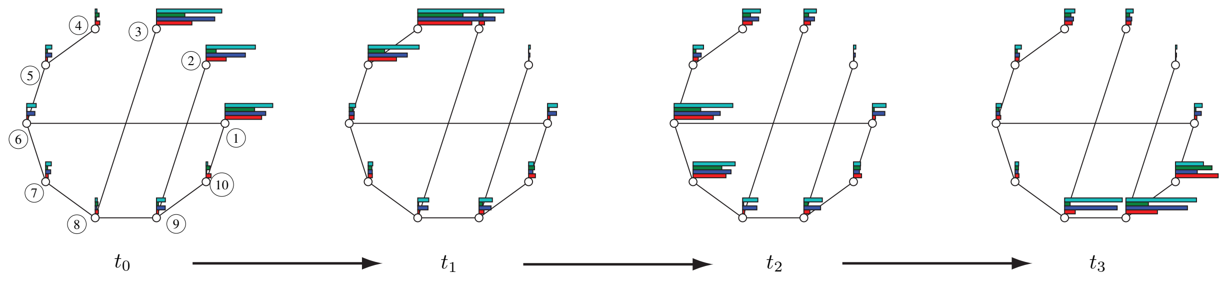

Fig. 1. Four configurations of a system with 10 tasks (nodes) and 4 traits. The trait abundance per task is represented by a bar plot. The edges of this strongly connected graph represent the possibility of switching between a pair of tasks. The system's initial distribution is shown at t_0, with subsequent desired target distributions at $t_{[1,2,3]}$.

B. Background

Given a set of tasks, and knowledge about the task requirements, our problem considers which robots should be allocated to which tasks. This problem is an instance of the *MT-MR-TA: Multi-Task Robots, Multi-Robot Tasks* problem [8], and can be reformulated as a set-covering problem that stems from combinatorial optimization. This problem is strongly NP-hard [13]. A greedy algorithm to solve this problem was proposed in [4], and later adapted for use in distributed multi-agent systems [24]. In the latter approach, the robots must compute all possible "coalitions" (groupings to solve a specific task), and agree on the best ones. Hence, this algorithm is best applied when the space of possible coalitions is naturally limited. Market-based approaches have also been considered to solve task allocation problems [6]. However, such approaches rely on bidding mechanisms that make extensive use of communication, and hence, scale poorly as the number of robots and tasks increases (not to mention that they do not address the problem of controller synthesis). In particular, for systems that are required to adapt to changing task requirements online, we need to consider algorithms that are efficient and that run on low-cost, resource-constrained mobile platforms in real time. Hence, we consider a strategy that is scalable in the number of robots and their capabilities, and is robust to changes in the robot population [3, 10]. An important property of this strategy is its inherently decentralized architecture, with robots switching between tasks (behaviors) stochastically. The key difference between our work and previous work is that we formulate our desired state as a distribution of traits among tasks, instead of specifying the desired state as a direct measure of the robot distribution. In other words, our framework allows a user to specify how much of a given capability is needed for a given task, irrespective of which robot type satisfies that need. As a consequence, we do not employ optimization methods that utilize final robot distributions in their formulations (which is the case in previous works [3] and [15]). Instead, we explicitly optimize the distribution of traits, and implicitly solve the combinatorial problem of distributing the right number of robots of a given type to the right tasks.

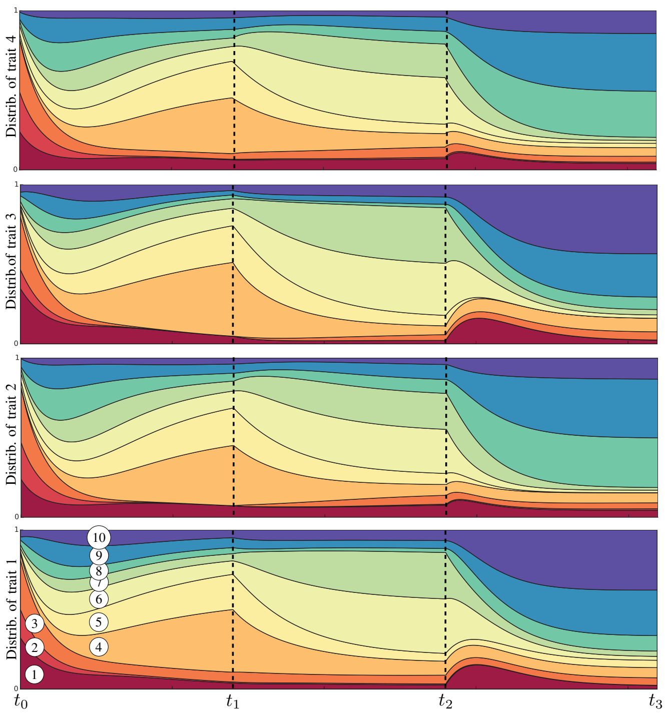

Fig. 2. Evolution over time of the trait distribution as specified by the distributions shown in Fig. 1. Each subplot represents one trait, indicating the distribution of that trait over the set of tasks (for each subplot, task 1 is shown at the bottom and task 5 at the top). The system's initial distribution is shown at t_0, with subsequent desired target distributions reached at $t_{[1,2,3]}$.

C. Contributions

In [23], we first presented a method that distributes a swarm of heterogeneous robots among a set of tasks that require specialized capabilities in order to be completed. Since our method is based on the derivation of an analytical gradient, it is very efficient with respect to state-of-the-art methods. The work in [21] builds on this result, proposing a real-time optimization method that enables an online adaptation of transition rates. Finally, in [22], we first propose a diversity metric, and show how the performance of the system relates to this measure. Apart from consolidating these previous publications, and assembling all major contributions into one coherent article, the present article provides the following contributions:

- In our previous work, we considered a single goal, which required an *exact* match of the final trait distribution to the desired trait distribution. Here, we also consider the formulation of a goal that allows for a *minimum* match (and does not penalize superfluous traits). By considering this additional goal function and extending the scope of our work, this paper presents a unified framework that allows us to simultaneously solve the allocation problem as well as the controller synthesis problem for heterogeneous swarms.
- We generalize the definition of our diversity metric to include multiple possible underlying goal functions. We show that our previous definition of *eigenspecies* is a subclass of the general class of *minspecies*. Furthermore, we show that our second goal function leads to an additional subclass of minspecies, and we refer to this new subclass as the *coverspecies*. We analyze the performance of our system as a function of dedicated diversity measures, one that is based on eigenspecies, and the other that is based on coverspecies.
- We provide a decentralized implementation of our online performance optimization algorithm for robot swarms that compute the controls locally. Simulation results show that our algorithm exhibits a graceful performance degradation in the case of increasing communication constraints.
- We reformulate our optimization problem as a constrained optimization problem that guarantees a minimum acceptable level of performance.
- We provide updated and new experimental results that support all novel contributions.

II. Problem Formulation

Heterogeneity and diversity are core concepts of this work. To develop our formalism, we borrow terminology from biodiversity literature [19]. We define our robot system as a *community* of robots. Each robot belongs to a *species*, defining the unique set of *traits* that encodes the robots' capabilities. In this work, we will consider binary instantiations of traits (corresponding to the presence or absence of a given trait in a species). Concretely, traits are a concatenation of one-hot encodings of different bucketized robot characteristics - such as geometric constraints (e.g., form factor), dynamics/kinematics (e.g., maximum speed), sensors (e.g., camera or laser range finder), computing power (e.g., storage capacity), or communication (e.g., typical range). In this work, we assume that the tasks have been encoded through such binary characteristics that represent the skill sets critical to task completion.

A. Notation

We consider a community of S robot species, with a total number of robots N, and $N^{(s)}$ robots per species such that $\sum_{s=1}^{S} N^{(s)} = N$. The community is defined by a set of U traits, and each robot species owns a subset of these traits. A species is defined by a binary vector $\mathbf{q}^{(s)} = [q_1^{(s)}, q_2^{(s)}, ..., q_U^{(s)}]$. We can then define a $S \times U$ matrix $\mathbf{Q}$, with rows $\mathbf{q}^{(s)}$:

$$\mathbf{Q}_{su} = \begin{cases} 0 \text{ , if species } s \text{ does not have trait } u \\ 1 \text{ , if species } s \text{ has trait } u \end{cases}$$

We model the interconnection topology of the M tasks via a directed graph, $\mathfrak{G} = (\mathcal{E}, \mathcal{V})$ where the set of vertices, $\mathcal{V}$, represents tasks $\{1, \ldots, M\}$ and the set of edges, $\mathcal{E}$, represents the ordered pairs (i, j), such that $(i, j) \in \mathcal{V} \times \mathcal{V}$, and i and j are adjacent. Edges denote the possibility to switch between two adjacent tasks. We assume the graph $\mathfrak{G}$ is a strongly connected graph, i.e., a path exists between any pair of vertices (in contrast to a *fully* connected graph, where an edge exists between any pair of vertices), and we assume the robots have knowledge of this graph. We assign every edge in $\mathcal{E}$ a transition rate, $k_{ij}^{(s)} > 0$, where $k_{ij}^{(s)}$ defines the transition probability per unit time for one robot of species s at site i to switch to site j. Here $k_{ij}^{(s)}$ is a stochastic transition rule. We impose a limitation on the maximum rate of each edge with $k_{ij}^{(s)} < k_{ij,\max}^{(s)}$. These values can be determined by applying system identification methods on the actual system. For example, in a system where nodes represent physically distributed sites, the transition rate represents the rate with which a specific path is chosen. This value can depend on observed factors, such as typical road congestion or the condition of the terrain. The distribution of the robots belonging to a species s at time t is described by a vector $\mathbf{x}^{(s)}(t) = [x_1^{(s)}(t), ..., x_M^{(s)}(t)]^\top$, and is summarized in a $M \times S$ matrix $\mathbf{X}(t)$, which we refer to as abstract state information. Then, if $\mathbf{q}^{(s)}$ are the rows of $\mathbf{Q}$, we have the $M \times U$ matrix $\mathbf{Y}$ that describes the distribution of traits on sites. For time t this relationship is given by

$$\mathbf{Y}(t) = \mathbf{X}(t) \cdot \mathbf{Q} \tag{1}$$

B. System

The initial state of the system is described by $\mathbf{X}(0)$, and hence, the initial distribution of traits at the sites is described by $\mathbf{Y}(0)$. The time evolution of the number of robots of species s at site i is given by a linear control law

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = \sum_{\forall j | (i,j) \in \mathcal{E}} k_{ji} x_i(t) - \sum_{\forall j | (i,j) \in \mathcal{E}} k_{ij} x_i(t), \tag{2}$$

which exhibits an explicit feedback structure. Then, for all species s, our base model is given by

$$\frac{\mathrm{d}\mathbf{x}^{(s)}}{\mathrm{d}t} = \mathbf{K}^{(s)} \mathbf{x}^{(s)} \quad \forall s \in 1, \ldots, S \tag{3}$$

where $\mathbf{K}^{(s)} \in \mathbb{R}^{M \times M}$ is a rate matrix with the properties

$$\begin{aligned} {\mathbf{K}^{(s)}}^\top \mathbf{1} &= \mathbf{0} & (4) \\ \mathbf{K}_{ij}^{(s)} &\geq 0 \quad \forall (i,j) \in \mathcal{E} & (5) \end{aligned}$$

These two properties result in the following definition:

$$\mathbf{K}_{ij}^{(s)} = \begin{cases} k_{ji}^{(s)}, & \text{if } i \neq j, (i,j) \in \mathcal{E} \\ 0, & \text{if } i \neq j, (i,j) \notin \mathcal{E} \\ -\sum_{i=1,(j,i)\in\mathcal{E}}^{M} k_{ij}^{(s)}, & \text{if } i = j \end{cases}$$

Since the total number of robots and the number of robots per species is conserved, the system in Eq. 3 is subject to the constraints

$$\mathbf{X}^\top \cdot \mathbf{1} \quad = \quad [N^{(1)}, N^{(2)}, \ldots, N^{(S)}]^\top \tag{6}$$

$$\text{with } \mathbf{X} \quad \succeq \quad \mathbf{0}, \tag{7}$$

where $\succeq$ is an element-wise greater-than-or-equal-to operator.

C. Problem Statement

Our aim is to redeploy the robots of each species, distributed according to $\mathbf{X}(0)$ initially, so that a desired trait distribution $\mathbf{Y}^\star$ is reached. As described in the introduction, we will consider two goals. A goal consists of a set of admissible trait distributions, and is described by a function $\mathcal{G} : \mathbb{N}^{+\ M \times U} \rightarrow \Omega$, where Ω is the set of sets of matrices of size $M \times U$. The goal function $\mathcal{G}$ takes as input a target trait distribution $\mathbf{Y}^\star$ and returns a set of admissible trait distributions $\mathcal{G}(\mathbf{Y}^\star)$. We study the following two goal functions in detail.

- $\mathcal{G}_1(\mathbf{Y}^\star) = \{\mathbf{Y} \mid \mathbf{Y}^\star = \mathbf{Y}\}$: This goal is achieved by a trait distribution that is exactly equal to the target trait distribution. Thus, the robots must organize themselves among tasks such that the exact number of traits is met for each task.
- $\mathcal{G}_2(\mathbf{Y}^\star) = \{\mathbf{Y} \mid \mathbf{Y}^\star \preceq \mathbf{Y}\}$: This goal is achieved by trait distributions that are equal or greater than the target trait distribution. Thus, robots can organize themselves such that there is an excess of traits for any task.

Finally, the problem consists of finding an optimal rate matrix $\mathbf{K}^{(s)\star}$ for each species s so that the goal is reached as fast as possible:

$$\mathbf{K}^{(s)\star}, \tau^\star \quad = \quad \underset{\mathbf{K}^{(s)}, \tau}{\operatorname{argmin}}\, \tau \tag{8}$$

$$\text{such that} \qquad \mathbf{X}(\tau^\star) \cdot \mathbf{Q} \in \mathcal{G}(\mathbf{Y}^\star) \tag{9}$$

The solution leads to a robot configuration $\mathbf{X}(\tau^\star)$ that satisfies Eq. 9, subject to Eq. 6 and Eq. 7. In other words, by computing optimal rates, we are centrally synthesizing the feedback policy based on the abstract state information $\mathbf{X}(0)$. We will initially assume that this information can be gathered centrally, and that the control input $\mathbf{K}^{(s)\star}$ can be broadcast to the swarm. Later, in Section V, we see how to infer the abstract state information using local estimators, enabling the robots to synthesize the feedback policy in a decentralized manner.

III. Diversity Metric

Since the desired state of our system is solely described through $\mathbf{Y}^\star$, the corresponding final robot distribution $\mathbf{X}(\tau^\star) = \mathbf{X}^\star$ that achieves the goal $\mathcal{G}(\mathbf{Y}^\star)$ is not known a priori. In particular, there may be several $\mathbf{X}^\star$ that satisfy Eq. 9 – this is true for both goals $\mathcal{G}_1$ and $\mathcal{G}_2$. Hence, we pose the question: *Can we infer properties of the species-trait matrix* $\mathbf{Q}$ *that quantify how easy it is to find a solution* $\mathbf{X}^\star$ *that reaches* $\mathcal{G}(\mathbf{Y}^\star)$*?* In the following, we show how $\mathbf{Q}$ embodies the *diversity* of the robot community, and how we can quantitatively evaluate the diversity to make conclusions about the system's performance.

A. Definitions

Given an unlimited number of robots per species, it may be possible to reach any given goal $\mathcal{G}(\mathbf{Y}^\star)$ with a subset of the original robot species (independent of the target trait distribution $\mathbf{Y}^\star$). We call the species belonging to an inclusion-wise minimal subset the *minspecies*, and we refer to the size of this subset as the *minspecies cardinality* of the system. More formally, we introduce the following terminology:

*Definition 1 (**Minspecies**)***:** In a robot community described by a species-trait matrix $\mathbf{Q}$, a *minspecies* set is a subset of rows of $\mathbf{Q}$ with minimal cardinality, such that the system can still reach the goal $\mathcal{G}(\mathbf{Y}^\star)$. We represent minspecies by a matrix $\hat{\mathbf{Q}}$ containing a subset of the original rows of $\mathbf{Q}$ such that for any $\mathbf{Y}^\star$ there exists at least one robot distribution $\hat{\mathbf{X}}$ for which $\hat{\mathbf{X}}\hat{\mathbf{Q}} \in \mathcal{G}(\mathbf{Y}^\star)$.

*Definition 2 (**Minspecies cardinality**)***:** The *minspecies cardinality* of a robot community is given by the cardinality of the minspecies set. It is a function $\mathcal{D}_\mathcal{G} : \{0,1\}^{S \times U} \rightarrow \mathbb{N}^+$ that takes a species-trait matrix $\mathbf{Q}$ as input, and returns the minimum number of rows of $\mathbf{Q}$ that are needed to reach $\mathcal{G}(\mathbf{Y}^\star)$ for any $\mathbf{Y}^\star$.

B. Implementation

In this section, we develop the minspecies cardinality of our two goals $\mathcal{G}_1$ and $\mathcal{G}_2$. In particular, we demonstrate that for both goals, the minspecies cardinality is a meaningful quantitative measure of the constraint in Eq. 9.

Proposition 1: The minspecies cardinality with respect to goal $\mathcal{G}_1$ is

$$\mathcal{D}_{\mathcal{G}_1}(\mathbf{Q}) = \operatorname{rank}(\mathbf{Q}) \tag{10}$$

This implementation of the minspecies cardinality is directly related to the concept of algebraic independence, and hence, we use the specialized term *eigenspecies* (as previously introduced in [22]).

Proof: The admissible trait distribution set contains a single target trait distribution, $\mathbf{Y}^\star$, and thus, Eq. 9 is equivalent to $\mathbf{Y}^\star = \mathbf{X}^\star\mathbf{Q}$. The matrix $\mathbf{Q}^\top$ can be rank-factorized into the product of two matrices $\mathbf{A}$ and $\hat{\mathbf{Q}}$ such that $\mathbf{Q}^\top = \hat{\mathbf{Q}}^\top\mathbf{A}^\top$ with $\hat{\mathbf{Q}}$ containing a subset of the rows of $\mathbf{Q}$ [25]. Since $\mathbf{Y}^\star = \mathbf{X}^\star\mathbf{Q} = \mathbf{X}^\star\mathbf{A}\hat{\mathbf{Q}}$, there exists a robot distribution $\hat{\mathbf{X}} = \mathbf{X}^\star\mathbf{A}$ for which $\hat{\mathbf{X}}\hat{\mathbf{Q}} = \mathbf{Y}^\star$. Hence, as $\hat{\mathbf{Q}}$ has minimal size (due to the rank-factorization), $\hat{\mathbf{Q}}$ is a minspecies matrix. ■

Indeed, the rank of $\mathbf{Q}$ quantifies the number of non-collinear species in $\mathbf{Q}$ that span the solution space of the equation $\mathbf{X}^\star\mathbf{Q} = \mathbf{Y}^\star$ (with $\mathbf{X}^\star$ unknown):

- If $\operatorname{rank}(\mathbf{Q}) < S$, the system is underdetermined, and an infinite number of solutions $\mathbf{X}^\star$ will satisfy Eq. 9. In other words, at least one species in the system can be replaced by a combination of the other species. As the rank decreases, the *redundancy* of the community increases.
- If $\operatorname{rank}(\mathbf{Q}) = S$, there is only one solution $\mathbf{X}^\star$ that satisfies Eq. 9. In other words, no species in the system

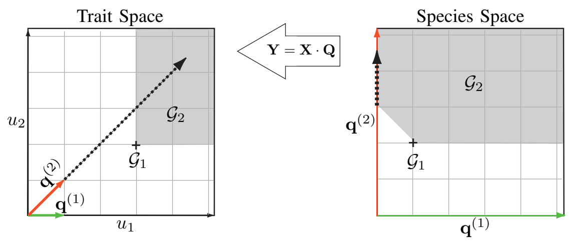

Fig. 3. This basic example illustrates how a goal is achieved by two species, $\mathbf{q}^{(1)}$ and $\mathbf{q}^{(2)}$. Species $\mathbf{q}^{(1)}$ only owns trait u_1, and species $\mathbf{q}^{(2)}$ owns both traits u_1 and u_2. The left panel shows how the two species span the trait space. Goal $\mathcal{G}_1$ can only be reached by a combination of both species, whereas goal $\mathcal{G}_2$ can be reached by species $\mathbf{q}^{(2)}$ alone. The same insight can be made by observing the right panel, which shows the goals in species space. We remind the reader that the species-trait matrix $\mathbf{Q}$ is used to map robot species into trait space.

can be replaced by a combination of the other species, and all species are fully *complementary*.

As an example, consider matrix

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \mathbf{A} \cdot \hat{\mathbf{Q}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

The rank of $\mathbf{Q}$ is 2, hence, $\mathcal{D}_{\mathcal{G}_1}(\mathbf{Q}) = 2$, which is the number of species in $\hat{\mathbf{Q}}$.

Proposition 2: The minspecies cardinality with respect to goal $\mathcal{G}_2$ is

$$\mathcal{D}_{\mathcal{G}_2}(\mathbf{Q}) = \min \sum_{s=1}^{S} a^{(s)} \tag{11}$$

$$\text{such that} \quad \sum_{s=1}^{S} a^{(s)} \mathbf{q}^{(s)} \succ \mathbf{0} \text{ and } a^{(s)} \in \{0, 1\}$$

This implementation of the minspecies cardinality is directly related to the concept of cover sets, and hence, we use the specialized term *coverspecies*.

Proof: The admissible trait distribution set for $\mathcal{G}_2$ contains all trait distributions that contain at least the specified amount of traits per task. Eq. 9 under goal $\mathcal{G}_2$ becomes $\mathbf{Y}^\star \preceq \mathbf{X}^\star \mathbf{Q}$. The matrix $\mathbf{Q}$ can be factorized into a product of two matrices $\mathbf{A}$ and $\hat{\mathbf{Q}}$ such that $\mathbf{Q} \preceq \mathbf{A}\hat{\mathbf{Q}}$. Since $\mathbf{Y}^\star \preceq \mathbf{X}^\star \mathbf{Q} \preceq \mathbf{X}^\star \mathbf{A} \hat{\mathbf{Q}}$ (since $\mathbf{X}^\star \succeq \mathbf{0}$), there exists a distribution $\hat{\mathbf{X}} = \mathbf{X}^\star \mathbf{A}$ for which the goal is reached. If $\hat{\mathbf{Q}}$ has the minimum number of rows possible, $\hat{\mathbf{Q}}$ is a minspecies matrix, and thus, finding $\hat{\mathbf{Q}}$ amounts to finding the minimum cover-set required to cover all traits with selected species of $\mathbf{Q}$. ■

We consider the same example as above:

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \preceq \mathbf{A} \cdot \hat{\mathbf{Q}} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

The minimum cover-set has size 1, hence, $\mathcal{D}_{\mathcal{G}_2}(\mathbf{Q}) = 1$, which is the number of species in $\hat{\mathbf{Q}}$.

Figure 3 illustrates a basic example of how goals $\mathcal{G}_1$ and $\mathcal{G}_2$ are achieved by two species, defined as $\mathbf{q}^{(1)} = [1, 0]$ and $\mathbf{q}^{(2)} = [1, 1]$. The left panel shows how the goals occupy the trait space, and the right panel shows how they occupy the species space. In particular, the panels illustrate how $\mathcal{G}_2$ can be reached by species $\mathbf{q}^{(2)}$ alone (see the dashed arrow). Species $\mathbf{q}^{(1)}$ and $\mathbf{q}^{(2)}$ both belong to the eigenspecies of $\mathcal{G}_1$, while $\mathbf{q}^{(2)}$ is the only coverspecies of $\mathcal{G}_2$.

IV. Methodology

In this section, we describe our methodology for obtaining an optimal transition matrix $\mathbf{K}^{(s)\star}$ for each species so that the desired trait distribution is reached as fast as possible. Two general approaches have been considered so far [3]: convex optimization and stochastic optimization. The convex optimization approach requires knowledge of the desired final robot distribution. However, our problem formulation specifies a desired trait distribution $\mathbf{Y}^\star$ without explicit definition of the final robot distribution $\mathbf{X}^\star$. Fully stochastic schemes such as Metropolis optimization have been shown to produce similar results, but they are not computationally efficient, and are ill-suited to real-time applications. In the following, we present a differentiable constrained optimization problem that can be efficiently solved through gradient descent techniques. Our method explicitly minimizes the convergence time of $\mathbf{K}^{(s)}$, unlike the convex optimization methods presented in [3], which approximate $\mathbf{K}^{(s)}$ with a symmetric equivalent (forcing bidirectionally equal transition rates between sites). Additionally, it is able to find optimal transition rates with only knowledge of $\mathbf{Y}^\star$ and $\mathbf{X}(0)$ (i.e., without knowledge of $\mathbf{X}^\star$).

A. Optimization Problem

We combine the solution of the linear ordinary differential equation, Eq. 3, with Eq. 1 to obtain the solution:

$$\mathbf{Y}(\mathbf{K}^{(1\ldots S)}, t; \mathbf{X_0}) = \sum_{s=1}^{S} e^{\mathbf{K}^{(s)} t} \mathbf{x}_0^{(s)} \cdot \mathbf{q}^{(s)} \tag{12}$$

To find the optimal transition rates $\mathbf{K}^{(s)\star}$ for $\mathcal{G}_1$ (details for $\mathcal{G}_2$ are in the appendix), we consider the trait distribution error

$$\mathbf{E_1}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0}) = \left\| \mathbf{Y}^\star - \mathbf{Y}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0}) \right\|_F^2 \tag{13}$$

where τ is the time at which the desired state is reached. The notation $\mathbf{x}_0^{(s)}$ is shorthand for $\mathbf{x}^{(s)}(0)$. The operator $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. An objective function based on $\mathbf{E_1}$ alone will return transition rates that may lead to the desired trait distribution quickly, but there is no guarantee that this state is also is a steady-state. Additionally, if we compute the transition rates at the outset of the experiment (without refining them online), we may wish to ensure that the state reached at the optimal time $\tau^\star$ remains near-constant. Hence, we also consider the difference in robot distribution

$$\mathbf{E_2}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0}) = \sum_{s=1}^{S} \left\| e^{\mathbf{K}^{(s)} \tau} \mathbf{x}_0^{(s)} - e^{\mathbf{K}^{(s)} (\tau + \nu)} \mathbf{x}_0^{(s)} \right\|_2^2$$

at time τ (when the desired trait distribution should be reached) and time $\tau + \nu$. Enforcing a small value for $\mathbf{E_2}$

allows us to guarantee that the robot distribution remains near-constant for arbitrarily long time intervals ν. This is possible because our model in Eq. 3 is stable [10], and the difference between the current robot distribution and the one at steady-state can only decrease monotonically over time. In other words, the trait distribution corresponding to the steady-state of $\mathbf{K}^{(s)\star}$ gets arbitrarily close to the distribution reached at τ as ν increases. We can now formulate our constrained optimization problem as:

$$\begin{aligned} &\text{minimize} && \tau \\ &\text{such that} && \mathbf{E_1}(\mathbf{K}^{(1\ldots S)},\tau;\mathbf{X_0}) \leq \epsilon_1 \\ &&& \mathbf{E_2}(\mathbf{K}^{(1\ldots S)},\tau;\mathbf{X_0}) \leq \epsilon_2 \\ &&& k_{ij}^{(s)} \leq k_{ij,\max}^{(s)} \\ &&& \tau > 0, \end{aligned} \tag{14}$$

which states that an optimal time $\tau^\star$ is found when the final trait distribution error is smaller than the admissible squared trait error ϵ_1, and when the difference in robot distributions at times $\tau^\star$ and $\tau^\star + \nu$ is smaller than the admissible squared deviation ϵ_2, subject to maximum transition rates $k_{ij,\max}^{(s)}$. The smaller we choose ϵ_1, the closer the trait distribution at time $\tau^\star$ will be to the desired trait distribution. The smaller we choose ϵ_2, the closer the robot distribution at time $\tau^\star$ is to the steady-state distribution of $\mathbf{K}^{(s)\star}$ (and the closer the trait distribution is to the steady-state trait distribution). While the first constraint will decrease $\tau^\star$, the second constraint will tend to increase it.

B. Analytical Gradients

There is no closed-form solution to the optimization problem in Eq. 14, hence we resort to numerical techniques. Nevertheless, we can maximize the efficiency of our computations by finding the closed-form expression of the gradient, for each of our constraints. In the following, for better readability, we will omit the explicit notation of the parameters of $\mathbf{E_1}(\mathbf{K}^{(1\ldots S)},\tau;\mathbf{X_0})$ and $\mathbf{E_2}(\mathbf{K}^{(1\ldots S)},\tau;\mathbf{X_0})$, and write $\mathbf{E_1}$ and $\mathbf{E_2}$, respectively. Let us first consider the derivative of $\mathbf{E_1}$. By applying the chain rule, the derivative with respect to the transition matrix $\mathbf{K}^{(s)}$ is

$$\frac{\partial \mathbf{E_1}}{\partial \mathbf{K}^{(s)}} = \frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} \cdot \frac{\partial e^{\mathbf{K}^{(s)}\tau}}{\partial \mathbf{K}^{(s)}\tau} \cdot \frac{\partial \mathbf{K}^{(s)}\tau}{\partial \mathbf{K}^{(s)}} \tag{15}$$

We first compute the derivative of the cost with respect to the expression $e^{\mathbf{K}^{(s)}\tau}$

$$\frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} = -2\left[\mathbf{Y}^\star - \mathbf{Y}(\mathbf{K}^{(s)},\tau;\mathbf{X_0})\right] \cdot \left[\mathbf{x}_0^{(s)} \cdot \mathbf{q}^{(s)}\right]^\top \tag{16}$$

The derivation of the 2nd element of Eq. 15 requires the derivative of the matrix exponential. Computing the derivative of the matrix exponential is not trivial. We adapt the closed-form solution given in [12] to our problem, and write the gradient of our constraint as

$$\frac{\partial \mathbf{E_1}}{\partial \mathbf{K}^{(s)}} = \mathbf{V^{-1}}^\top \left[\mathbf{V}^\top \frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} \mathbf{V^{-1}}^\top \odot \mathbf{W}(\tau)\right] \mathbf{V}^\top \tau \tag{17}$$

where $\odot$ is the Hadamard product, $\mathbf{K}^{(s)} = \mathbf{V}\mathbf{D}\mathbf{V}^{-1}$ is the eigendecomposition of $\mathbf{K}^{(s)}$. $\mathbf{V}$ is the $M \times M$ matrix whose jth column is a right eigenvector corresponding to eigenvalue d_i, and $\mathbf{D} = \operatorname{diag}(d_1, \ldots, d_M)$. The matrix $\mathbf{W}(t)$ is composed as follows [1]

$$\mathbf{W}(t) = \begin{cases} (e^{d_i t} - e^{d_j t})/(d_i t - d_j t) & i \neq j \\ e^{d_i t} & i = j \end{cases}$$

We also need the derivative with respect to parameter τ. This derivative is computed analogously to the derivative with respect to $\mathbf{K}^{(s)}$ (confer Eq. 17). We have

$$\frac{\partial \mathbf{E_1}}{\partial \tau} = \sum_{s=1}^{S} \mathbf{1}^\top \mathbf{V^{-1}}^\top \mathbf{A}_1 \mathbf{V}^\top \mathbf{K}^{(s)} \mathbf{1} \tag{18}$$

and

$$\mathbf{A}_1 = \mathbf{V}^\top \frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} \mathbf{V^{-1}}^\top \odot \mathbf{W}(\tau) \tag{19}$$

Now let us consider the derivative of the second constraint in Eq. 14. Again, we apply the chain rule to obtain

$$\begin{aligned} \frac{\partial \mathbf{E_2}}{\partial \mathbf{K}^{(s)}} &= \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}\tau}} \frac{\partial e^{\mathbf{K}^{(s)}\tau}}{\partial \mathbf{K}^{(s)}\tau} \frac{\partial \mathbf{K}^{(s)}\tau}{\partial \mathbf{K}^{(s)}} \\ &- \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}} \frac{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}}{\partial \mathbf{K}^{(s)}(\tau+\nu)} \frac{\partial \mathbf{K}^{(s)}(\tau+\nu)}{\partial \mathbf{K}^{(s)}} \end{aligned} \tag{20}$$

The outer derivative is

$$\begin{aligned} \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}\tau}} &= \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}} \\ &= 2\left[e^{\mathbf{K}^{(s)}\tau}\mathbf{x}_0^{(s)} - e^{\mathbf{K}^{(s)}(\tau+\nu)}\mathbf{x}_0^{(s)}\right] \cdot \mathbf{x}_0^{(s)\top} \end{aligned} \tag{21}$$

We apply the same development as in Eq. 17 to obtain the equation

$$\frac{\partial \mathbf{E_2}}{\partial \mathbf{K}^{(s)}} = \mathbf{V^{-1}}^\top \left[\mathbf{A}_2\tau - \mathbf{A}_3(\tau+\nu)\right] \mathbf{V}^\top \tag{22}$$

with

$$\mathbf{A}_2 = \mathbf{V}^\top \cdot \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}\tau}} \cdot \mathbf{V^{-1}}^\top \odot \mathbf{W}(\tau) \tag{23}$$

and

$$\mathbf{A}_3 = \mathbf{V}^\top \cdot \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}} \cdot \mathbf{V^{-1}}^\top \odot \mathbf{W}(\tau+\nu) \tag{24}$$

The derivative with respect to time τ is analogous:

$$\frac{\partial \mathbf{E_2}}{\partial \tau} = \sum_{s=1}^{S} \mathbf{1}^\top \mathbf{V^{-1}}^\top \left[\mathbf{A}_2 - \mathbf{A}_3\right] \mathbf{V}^\top \mathbf{K}^{(s)} \mathbf{1} \tag{25}$$

For all the above, the derivative with respect to the off-diagonal elements ij of the matrix $\mathbf{K}^{(s)}$, with $(i,j) \in \mathcal{E}$, is

$$\frac{\partial \mathbf{E_z}}{\partial k_{ij}^{(s)}} = \left\{\frac{\partial \mathbf{E_z}}{\partial \mathbf{K}^{(s)}}\right\}_{ij} - \left\{\frac{\partial \mathbf{E_z}}{\partial \mathbf{K}^{(s)}}\right\}_{jj} \tag{26}$$

where $\{\cdot\}_{ij}$ denotes the element on row i and column j.

[1]Here, we assume that that $\mathbf{K}^{(s)}$ has M distinct eigenvalues. If this is not the case, an analogous decomposition of $\mathbf{K}^{(s)}$ to Jordan canonical form is possible, as elaborated in [12]. We note that for most models of interest, however, this is rarely the case.

C. Computational Complexity

The overall computational complexity of computing the gradients of both constraints of our optimization problem is $O(S \cdot M^3 + S \cdot M^2 \cdot U)$. The first part of this complexity is dictated by the eigenvalue decomposition, which is known to be $O(M^3)$ for non-sparse matrices [5][2]. We compute this decomposition only once per optimization (see Eq. 17, where $\mathbf{K}^{(s)} = \mathbf{VDV}^{-1}$), for each optimization of $\mathbf{K}^{(s)}$. The second part is dictated by the multiplication of the matrices in Eq. 17, for which the cost is $O(M^2 \cdot U)$. Globally speaking, the computation grows linearly with the number of species and traits, and it grows slightly slower than the cube of the number of tasks. When studying heterogenous system, it is indeed a valuable result that the gradient scales at most linearly with the number of traits and species in order to allow for the exploration of a wider range of robot capabilities. Overall, the average time to compute the gradient for a system with $M = 8$, $U = 4$, and $S = 4$ is around 1.35 ms with $\nu = 0$, and 2.2 ms with $\nu > 0$ (the number of parameters to optimize can be as large as 225 in this case, depending on the graph's adjacency matrix). The code was implemented in Python using the NumPy and SciPy libraries, and tested on a 2 GHz Intel Core i7 using a single CPU.

D. Online Optimization of $\mathbf{K}^{(s)}$

Thanks to the analytical gradients developed above, our optimization problem can be solved efficiently. Building on this result, we implement a continuous, online optimization strategy that allows us to refine the optimal $\mathbf{K}^{(s)\star}$ as a function of the current state of the robot system, in real-time. In noisy systems, where the trajectories of individual agents exhibit deviations from predicted macroscopic trajectories, this strategy inevitably leads to an improvement of the convergence time. Furthermore, as seen in our example in Figures 1 and 2, we may wish to program a sequence of desired trait distributions, with autonomous transition rate updates. The key idea is that by taking the actual robot distribution into account, an online method can recompute updated optimal transition rates. Practically, we initially compute $\mathbf{K}^{(s)\star}$ at time $t = 0$ to control the system over a finite period δ from $t = 0$ to $t = \delta$. After that period (at time $t = \delta$), we optimize a new value of $\mathbf{K}^{(s)\star}$ that controls the system for the next period, as a function of the actual robot distribution that was encountered at time $t = \delta$. This process can be repeated indefinitely. The value δ is called the sampling time. Our online control policy computes the optimal transition rates, and is rewritten as a function of the robot distribution:

[2]In the special case where all eigenvalues are distinct, the eigenvalue decomposition can be reduced to $O(M^{2.376}\log(M))$ [18].

$$\begin{aligned}
\mathbf{K}^{(s)\star}(t), \tau^\star(t) \quad &= \quad \underset{\mathbf{K}^{(s)},\tau}{\operatorname{argmin}}\, \tau \qquad (27)\\
\text{such that} \quad & \quad \mathbf{E_1}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X}(t_p)) \le \epsilon_1\\
& \quad \mathbf{E_2}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X}(t_p)) \le \epsilon_2\\
& \quad k_{ij}^{(s)} \le k_{ij,\max}^{(s)}\\
& \quad \tau > 0,\\
\text{with} \quad & \quad t_p \le t < t_p + \delta\\
& \quad t_p \in k\delta, k \in \mathbb{N},
\end{aligned}$$

where t_p is the time at which optimizations happen. For cases where the optimization time becomes large (implying that δ also becomes large), we need to use a strategy that accounts for computation delay, such as those presented in [17]. Also, we note that we can accelerate the computations by setting the initial values of the present sampling window with optimized values of the preceding sampling window (i.e., warm start).

V. Robot Controller

The previous sections describe the methods with which we obtain optimal transition rates $\mathbf{K}^{(s)\star}$. If we assume an architecture such as described in [16], then the optimization is run off-board, centrally, with knowledge of abstract state information $\mathbf{X}(t)$ (i.e, the current distribution of the robot swarm among tasks). In return, the robots only need to receive information on the optimal transition rates for their species, $k_{ij}^{(s)}$. We note that this information is represented by a small number of values (at most M^2 values per species, or a much smaller number if the graph is sparse), and needs to be transmitted to the robots only at the start of each new redistribution.

Conversely, if the robots run the optimization algorithm on-board, they need to estimate the abstract state information $\mathbf{X}(t)$ locally. This knowledge can be obtained through communication with neighboring robots (which broadcast their current task allocation). Due to communication constraints, obtaining an accurate representation of $\mathbf{X}(t)$ is often not possible. Hence, our methods must perform acceptably well, even when these values are approximate [14]. In the present work, we use a method that assumes a uniform distribution of all robots for which we do not know the current task allocation.

The agent-level control is based on the transition rates $k_{ij}^{(s)}$ encoded by the transition matrix $\mathbf{K}^{(s)}$: A robot of species s at task i transitions to task j according to probability $p_{ij}^{(s)}$ that is an element of the matrix $\mathbf{P}^{(s)} = e^{\mathbf{K}^{(s)}\Delta T}$, where ΔT is the duration of one time-step. Hence, in order to determine which task the robot must transition to next, it samples a new task with a probability according to $\mathbf{P}^{(s)}$. This is equivalent to sampling from the discrete probability distribution $\mathcal{P}(p_{i1}^{(s)}, \ldots, p_{iM}^{(s)})$, where i represents the current task. This procedure is shown in Algorithm 1. We note that as the robot is transitioning to a new task, it continues the control loop (i.e., sampling new tasks). Although we do not explicitly model transitioning time, the resulting behavior is very close to what is predicted by the macroscopic model in Eq. 3, as is shown later in Section VI.

Algorithm 1 Controller($s, M, N^{(1),\ldots,(S)}, \mathbf{Q}, \mathbf{Y}^\star, \delta, \Delta T$)

```
1:  $i \leftarrow$ GetInitialTask()
2:  $t \leftarrow 0$
3:  while 1 do
4:      BroadcastAllocation($< s, i >$)
5:      if modulo($t, \delta$) $= 0$ then
6:          $A \leftarrow$ GetAllocations()
7:          $\mathbf{X}_{local} \leftarrow$ EstimateRobotDistr($A, N^{(1),\ldots,(S)}, M$)
8:          $\mathbf{K}^{(s)\star} \leftarrow$ Optimize($\mathbf{X}_{local}, \mathbf{Q}, \mathbf{Y}^\star$)
9:          $\mathbf{P}^{(s)} = e^{\mathbf{K}^{(s)\star}\Delta T}$
10:     end if
11:     $t \leftarrow t + \Delta T$
12:     $m \sim \mathcal{P}(p^{(s)}_{i1}, \ldots, p^{(s)}_{iM})$
13:     if $m \neq i$ then
14:         Switch to task $m$
15:         $i \leftarrow m$
16:     end if
17:     Wait $\Delta T$
18: end while
```

VI. Results

In the following, we present results that show that *(i)* our method successfully achieves the deployment of a heterogeneous system of robots so that a desired trait distribution is reached, *(ii)* that our method extends itself naturally to decentralized architectures with communications constraints, and that *(iii)* we are able to relate the performance to the diversity of the system. We evaluate these claims over multiple levels of abstraction.

A. Performance Metric

The degree of convergence to $\mathbf{Y}^\star$ is expressed by the fraction of misplaced traits. For our two goals, we formulate this as:

$$\mu_{\mathcal{G}_1}(\mathbf{Y}) = \frac{\|\mathbf{Y}^\star - \mathbf{Y}\|_1}{2\|\mathbf{Y}^\star\|_1},\ \mu_{\mathcal{G}_2}(\mathbf{Y}) = \frac{\|\max(\mathbf{Y}^\star - \mathbf{Y}, 0)\|_1}{\|\mathbf{Y}^\star\|_1} \quad (28)$$

Previous work has shown the benefit of validating methods over multiple levels of abstraction (sub-microscopic, microscopic, and macroscopic) [20]. In this section, we propose an evaluation of our methods on two levels: macroscopic, and microscopic. Indeed, the most efficient way of simulating a large-scale system of robots is by considering a continuous macroscopic model, derived directly from the ordinary differential equation, Eq. 3. In order to validate our methods at a lower level of abstraction, we also implement a discrete microscopic model that emulates the behavior of individual robot controllers. The agent-level control is based on Algorithm 1. Running multiple iterations of the microscopic model enables us to capture the stochasticity resulting from our control system. In the remainder of this paper, we use $\Delta T = 0.04\,\mathrm{s}$, unless stated otherwise.

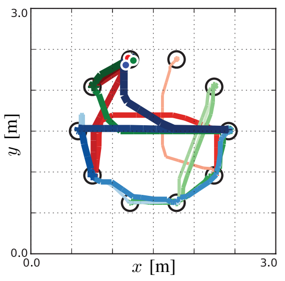

Fig. 4. Trail for 3 robots, one of each species, for the first 700 seconds (segment t_0 to t_1) in Fig. 1. The robots start at sites 1, 2, and 3, respectively, and end at site 4. The earlier the trail, the more transparent the color.

B. Example

To illustrate our method in more detail, let us consider the example portrayed earlier, in Fig. 1. The graph is generated randomly according to the Watts-Strogatz model [27] (with a neighboring node degree of $K = 3$, and a rewiring probability of $\gamma = 0.6$; the graph is guaranteed to be connected). The robot community consists of 3 species and 4 traits, and is defined as follows:

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \text{ with } \mathbf{X}^\top \cdot \mathbf{1} = [150, 50, 300]^\top$$

In this example, $N = 500$ robots transition among $M = 10$ tasks. We sample a random initial robot distribution $\mathbf{X}(t_0)$ with a majority of traits in use at tasks 1, 2, and 3. We specify a randomly generated desired trait distribution, which is visualized in Fig. 1 at t_1. The final robot distribution $\mathbf{X}(t_1)$ then serves as the initial distribution for a subsequent reconfiguration targeting the trait distribution visualized at t_2. As this process is repeated, it demonstrates how our method can be used to redistribute a swarm of robots through time so that changing trait requirements are met.

To illustrate the performance of our method, we implement a kinematic point-simulator, emulating a swarm of robots at a microscopic level. The robots move on a two-dimensional plane, which is 3 m in size, where the tasks are represented by spatially anchored sites that have a radius of 0.05 m, and are placed along a circle of radius 1.75 m. The paths between the sites are defined according to the adjacency matrix of the graph shown in Fig. 1. The robots travel with an average speed of $0.06\,\frac{\mathrm{m}}{\mathrm{s}}$, as they transition from site to site, with a maximum transition rate $k^{(s)}_{ij,\max} = 0.02\,\mathrm{s}^{-1}$. Fig. 4 shows the trail laid by three robots during the period t_0 to t_1, as they travel from their initial sites to the final site. We note that the motion control used for the robots in this simulator is identical to the one used to control physical robots in a previous experiment [22].

In order to quantify the performance of our system, we perform 10 runs of our simulator, and evaluate the ratio of misplaced traits as a function of time. We also evaluate the performance of the system at a macroscopic level. The results are shown in Fig. 5. We observe that the trait error decreases exponentially. Initially, the microscopic and

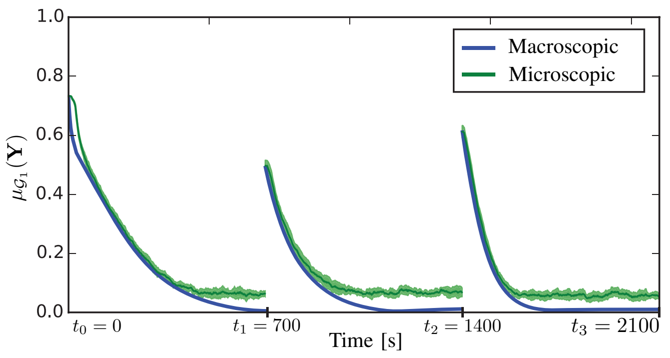

Fig. 5. Ratio of misplaced traits for the redistribution problem shown in Fig. 1. The plot shows the macroscopic model as well as the average over 10 iterations of the microscopic model. The shaded area shows the standard deviation.

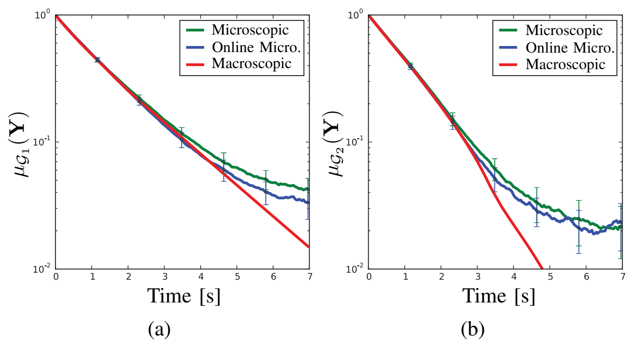

Fig. 6. Ratio of misplaced traits over time, in log-scale, for a graph with $M = 8$ nodes. The plot shows the macroscopic model as well as the average over 50 iterations of the microscopic model, with and without online optimization. The error bars show the standard deviation. (a) Goal function $\mathcal{G}_1$, (b) Goal function $\mathcal{G}_2$.

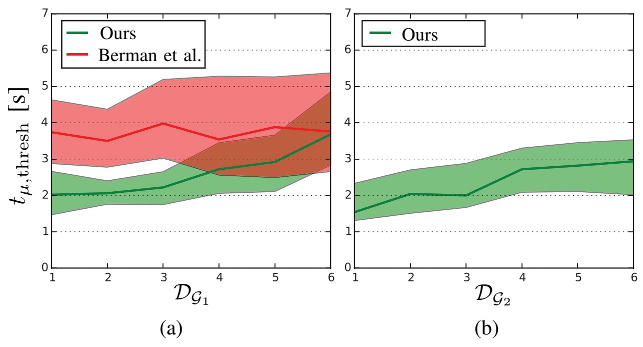

Fig. 7. The plot shows the median convergence time evaluated on the microscopic model, with $t_{\mu_{\text{thresh}}}$ for $\mu_{thresh} = 2.5\%$, as a function of the minspecies cardinality, for 60 random graphs per cardinality value. The system has $M = 10$ tasks and $S = 6$ species. The shaded area shows the 25th and 75th percentiles. (a) Goal function $\mathcal{G}_1$, (b) Goal function $\mathcal{G}_2$.

macroscopic models show good agreement, but as the system approaches steady-state, the stochasticity of the microscopic point-simulator forces the error ratio (which counts absolute differences) to be larger than 0. Note that systems with slower dynamics and more robots have a lower noise intensity, and achieve lower average errors at steady-state. This experiment shows that our framework is able to cope with spatiality and temporal delays, even though these phenomena are not modeled explicitly in our optimization problem. Throughout the rest of this section, we will focus on the core properties of our method, and hence, we simplify our implementation of the microscopic model by considering instantaneous transitions from one task to the next, within non-spatial configurations.

C. Online Optimization

We evaluate the performance of our online optimization algorithm described in Section IV-D, for both goal functions $\mathcal{G}_1$ and $\mathcal{G}_2$, and compare it to the macroscopic model. Fig. 6 shows the ratio of misplaced traits $\mu(\mathbf{Y})$ over time for a graph with $M = 8$ nodes, for $S = 4$ and $U = 5$, and a total number of robots $N = 1000$. The initial distribution consists of traits randomly allocated to one half of the tasks, and the desired distribution consists of traits randomly allocated to the remaining half of tasks. Fig. 6(a) considers goal $\mathcal{G}_1$ (matching the desired trait distribution *exactly*), and Fig. 6(b) considers goal $\mathcal{G}_2$ (matching a *minimum* desired amount of traits per task). For both plots, we run 50 iterations of the discrete microscopic model, with and without online optimization. The online optimization method was implemented with $\delta = 20 \cdot \Delta T$. We observe that the trait error decreases exponentially. Since the online optimization method takes the current robot configuration into account, it produces transition rates that lead to lower errors. Finally, we observe that goal $\mathcal{G}_2$ converges faster than $\mathcal{G}_1$, due to the additional degrees of freedom in the system (i.e., traits that can be allocated to any task in the system, as soon as the minimum amount is reached). As previously observed in Fig. 5, we see that here as well, the stochasticity of the microscopic models prevents the error from continuing to decrease exponentially as the system approaches steady-state.

D. Impact of Diversity

Our aim is to observe the impact of diversity on system performance. We accomplish this by evaluating the time of convergence to the desired trait distribution as a function of our proposed diversity measure, the minspecies cardinality. We consider a system of $M = 10$ tasks and $S = 6$ species, and generate random species-trait matrices $\mathbf{Q}$ (for both $\mathcal{G}_1$ and $\mathcal{G}_2$) with minspecies cardinality values ranging from 1 to 6. The system is evaluated on 60 graphs, for each minspecies cardinality value, with a random initial robot distribution and a random desired trait distribution per graph. We measure the time $t_{\mu,\text{thresh}}$ at which the system converges to a value $\mu_{\text{thresh}} = 2.5\%$ of misplaced traits, and say that one system converges faster than another if it takes less time for $\mu(\mathbf{Y})$ to decrease to μ_{thresh}. Similar performance metrics have been proposed in [3, 10].

Fig. 7 shows the results for the goal $\mathcal{G}_1$. Our optimization method is shown in green. We see that as the minspecies cardinality of the system increases, the time to convergence also increases. Indeed, the size of the solution space of Eq. 9

decreases as the minspecies cardinality increases. In other words, the more the species are complementary, the harder the system is to optimize. Also, we compare our method to a benchmark convex optimization approach that stems from [3], denoted in the latter work as **[P1]**[3]. We choose this method because it is to-date one of the most efficient methods that optimizes the convergence time of homogenous swarm systems, and because it has roughly the same computational complexity as our method. The results of this method are shown in red. We see that the performance does not correlate with the minspecies cardinality. Since the method does not optimize the reconfiguration for desired trait distributions, it is input with a potentially sub-optimal final robot distribution (which is exacerbated for low minspecies cardinality). Our method improves upon this state-of-the-art benchmark method by 25% for $\mathcal{D}_{\mathcal{G}_1} = 5$ and by 46% for $\mathcal{D}_{\mathcal{G}_1} = 1$. Fig. 7(b) shows the results for the goal $\mathcal{G}_2$. As before, we see that as the minspecies cardinality of the system increases, the time to convergence also increases. We verify that $\mathcal{D}_{\mathcal{G}_2}$ is a more appropriate measure of diversity than $\mathcal{D}_{\mathcal{G}_1}$ for goal $\mathcal{G}_2$ by computing the Pearson correlation coefficient. Using $\mathcal{D}_{\mathcal{G}1}$ on this data produces a correlation of 0.23 (with p-value $< 10^{-4}$), while $\mathcal{D}_{\mathcal{G}_2}$ produces a correlation of 0.35 (with p-value $< 10^{-4}$), which is a 52% increase over $\mathcal{D}_{\mathcal{G}_1}$. This validates the use of two distinct diversity measures for our two distinct goals.

E. Decentralized Online Performance Optimization

To conclude our results section, we consider the online optimization of transition rates within a decentralized controller that uses only local communication to obtain state information about the robot swarm. We remind the reader that the robots need knowledge of abstract state information (i.e., the distribution of the robot swarm among tasks, $\mathbf{X}(t_p)$), at the start of each optimization, see Eq. 27. Hence, if we intend the robots to obtain this information through local communication channels only, then we need to understand the effects that different communication topologies will have on the performance of the system. Our controller is based on Eq. 27, which updates transition rates at time intervals δ. To emulate communication constraints, we consider incremental coverage: at the most restricted level, we assume that robots are only able to communicate with other robots collocated at the same task; this assumption is incrementally relaxed, as we increase the communication neighborhood to include robots at adjacent tasks, within a fixed hop-count relative to the present task. Fig. 8 illustrates this concept for hop-counts of 0, 1, and 3. For this particular graph, a hop-count of 4 reaches full coverage (shown in Fig. 8(a)).

[3]This method implicitly optimizes the convergence time by optimizing the asymptotic convergence rate (of a system of homogenous robots). We adapt the method to our problem: we minimize the second eigenvalue λ_2 of a symmetric matrix $\mathbf{S}^{(s)}$, such that $\lambda_2(\mathbf{S}^{(s)}) \geq \mathrm{Re}(\lambda_2(\mathbf{K}^{(s)}))$. Since this method requires the knowledge of the desired species distribution $\mathbf{X}^{\star}$, we artificially bootstrap the method by computing a random instantiation of $\mathbf{X}^{\star}$ that satisfies the desired trait distribution defined by Eq. 1. We note that in practical applications, computing a good instantiation of $\mathbf{X}^{\star}$ is not trivial.

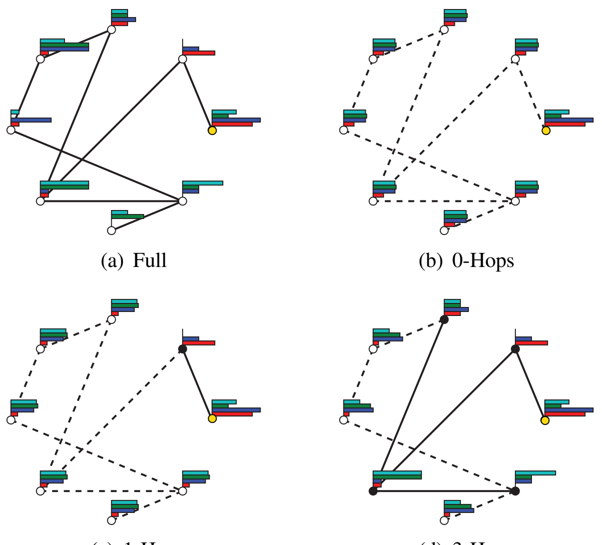

Fig. 8. Communication topology, as perceived by robots located at the task marked in yellow. The robots have knowledge of the trait distribution in their local neighborhood, which is defined by the hop-count. We show the topology for 0, 1, and 3-hop neighborhoods, where tasks within the neighborhood are marked in black. The remaining traits are assumed to be distributed uniformly among all tasks that are outside of the local neighborhood.

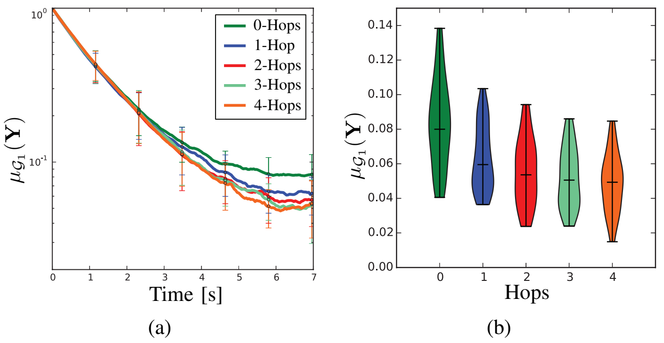

Fig. 9. (a) Ratio of misplaced traits over time, in log-scale. The results are averaged over 4 iterations for 8 different for a graphs with $M = 8$ nodes. The error bars show the standard deviation. (b) Violin plots with marked median values for data in the final time-step of plot (a).

Fig. 9(a) shows the ratio of misplaced traits as a function of time, for five different hop-counts. Fig. 9(b) shows the distribution of the data points of the final time-step, with inclusion of all extrema. We observe that, the more we restrict our communication topology, the higher the error at steady-state. We note that this performance degradation is graceful — as we reduce the size of our communication neighborhood from 4 hops to 0 hops, the final ratio of misplaced traits increases by 3%, 6%, 11% and 34%, respectively, and still converges to a modest error of 8% for the most restrictive communication radius.

VII. Conclusion

Overall, this work shows how global design variables (such as diversity) are incorporated at a high level of abstraction (i.e., macroscopic level), to produce optimal controllers that can also be implemented in decentralized form, and that are able to take realistic constraints into account (such as limited communication). By considering the specific problem of distributing a heterogeneous swarm of robots among a set of tasks with the goal of satisfying a desired distribution of robot capabilities among those tasks, we contribute to the understanding of the effects of *diversity* in heterogeneous swarms. We propose a formulation for heterogeneous robot systems through *species* and *traits*, and show how this formulation is used to achieve an optimal distribution of robots by specifying the desired final trait distribution. Using this formulation, we propose a diversity metric based on *minspecies* that indicates how performance is affected by diversity. We show that the more the robot community is diverse, the harder it is to optimize: by adding redundant (non-complementary) species, we increase the size of the solution space and facilitate the optimization. In particular, we show that this conclusion is valid for two different goals that require specific implementations of our diversity metric. The latter implementations are based on specializations of the minspecies, and are referred to as *eigenspecies* and *coverspecies*.

Our method consists of a constrained optimization problem, for which we find a computationally efficient solution that is capable of producing fast convergence times, even for large numbers of species and traits. Indeed, our computation is fully scalable with respect to the number of robots, number of species and number of traits. Building on this result, we propose a real-time optimization method that enables an online adaptation of transition rates as a function of the state of the current robot distribution. We evaluate our methods by means of microscopic simulations, and show how the performance of the latter is well predicted by the macroscopic equations.

Future work will consider the development of methods that automatically generate desired trait distributions as a function of underlying real-world problems.

VIII. Acknowledgment

We gratefully acknowledge the support of ONR grants N00014-15-1-2115, N00014-14-1-0510, and N00014-13-1-0731, ARL grant W911NF-08-2-0004, NSF grant IIS-1426840, and TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

Appendix

The optimization in Eq. 14 is reformulated for goal $\mathcal{G}_2$ with

$$\mathbf{E_1} = \left\| \max(\mathbf{Y}^\star - \mathbf{Y}, 0) \right\|_F^2 \tag{29}$$

and the derivative (analogous to Eq. 16) is

$$\frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} = -2 \left[\max(\mathbf{Y}^\star - \mathbf{Y}, 0) \right] \cdot \left[\mathbf{x}_0^{(s)} \cdot \mathbf{q}^{(s)} \right]^\top \tag{30}$$

References

[1] W. Abbas and M. Egerstedt. Characterizing heterogeneity in cooperative networks from a resource distribution view-point. *Communications in Information and Systems*, 14:1–22, 2014.

[2] T. Balch. Hierarchic Social Entropy: An Information Theoretic Measure of Robot Group Diversity. *Autonomous Robots*, 8:209–237, 2000.

[3] S. Berman, Á. Halasz, M. A. Hsieh, and V. Kumar. Optimized Stochastic Policies for Task Allocation in Swarms of Robots. *IEEE Transactions on Robotics*, 25:927–937, 2009.

[4] V. Chvatal. A Greedy Heuristic for the Set-Covering Problem. *Mathematics of Operations Research*, 4(3):233–235, Aug. 1979.

[5] J. Demmel, I. Dumitriu, and O. Holtz. Fast Linear Algebra is Stable. *arXiv.org*, pages 1–26, 2007.

[6] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz. Market-Based Multirobot Coordination: A Survey and Analysis. *Proceedings of the IEEE*, 94(7):1257–1270, 2006.

[7] M. Dorigo, D. Floreano, L. M. Gambardella, F. Mondada, S. Nolfi, T. Baaboura, M. Birattari, M. Bonani, M. Brambilla, A. Brutschy, D. Burnier, A. Campo, A. L. Christensen, A. Decugniere, G. Di Caro, F. Ducatelle, E. Ferrante, A. Forster, J. M. Gonzales, J. Guzzi, V. Longchamp, S. Magnenat, N. Mathews, M. Montes de Oca, R. O'Grady, C. Pinciroli, G. Pini, P. Retornaz, J. Roberts, V. Sperati, T. Stirling, A. Stranieri, T. Stutzle, V. Trianni, E. Tuci, A. E. Turgut, and F. Vaussard. Swarmanoid: A Novel Concept for the Study of Heterogeneous Robotic Swarms. *IEEE Robotics & Automation Magazine*, 20(4):60–71, 2013.

[8] B. P. Gerkey and M. J. Mataric. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *Interntional Journal of Robotics Research*, 23(9):939–954, 2004.

[9] M. A. Hsieh, A. Cowley, F. J. Keller, L. Chaimowicz, B. Grocholsky, V. Kumar, C. J. Taylor, Y. Endo, R. C. Arkin, B. Jung, F. D. Wolf, G. S. Sukhatme, , and D. C. MacKenzie. Adaptive teams of autonomous aerial and ground robots for situational awareness. *Journal of Field Robotics*, 24:991–1014, 2007.

[10] M. A. Hsieh, Á. Halasz, S. Berman, and V. Kumar. Biologically inspired redistribution of a swarm of robots among multiple sites. *Swarm Intelligence*, 2(2-4):121–141, 2008.

[11] E. G. Jones, B. Browning, M. B. Dias, B. Argall, and M. Veloso. Dynamically Formed Heterogeneous Robot Teams Performing Tightly Coordinated Tasks. *International Conference on Robotics and Automation*, pages 570–575, 2006.

[12] J. D. Kalbfleisch and J. F. Lawless. The Analysis of Panel Data Under a Markov Assumption. *Journal of American Statistical Association*, 80:863–871, 1985.

[13] B. Korte and J. Vygen. Combinatorial Optimization:

Theory and Algorithms. Springer-Verlag, Berlin., 2000.

[14] T. W. Mather and M. A. Hsieh. Synthesis and analysis of distributed ensemble control strategies for allocation to multiple tasks. *Robotica*, 32(02):177–192, Dec. 2013.

[15] L. Matthey, S. Berman, and V. Kumar. Stochastic strategies for a swarm robotic assembly system. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1953–1958. IEEE, 2009.

[16] N. Michael, J. Fink, S. Loizou, and V. Kumar. Architecture, Abstractions, and Algorithms for Controlling Large Teams of Robots: Experimental Testbed and Results. In *Robotics Research; Springer Tracts in Advanced Robotics*, pages 409–419. Springer Berlin Heidelberg, 2011.

[17] M. Milam, R. Franz, J. Hauser, and M. Murray. Receding horizon control of vectored thrust flight experiment. *IEE Proceedings on Control Theory and Applications*, 152:340–348, 2015.

[18] V. Y. Pan and Z. Q. Chen. The Complexity of the Matrix Eigenproblem. *ACM Symposium on Theory of Computing*, pages 507–516, 1999.

[19] O. L. Petchey and K. J. Gaston. Functional diversity (FD), species richness and community composition. *Ecology Letters*, pages 402–411, 2002.

[20] A. Prorok, N. Correll, and A. Martinoli. Multi-level spatial modeling for stochastic distributed robotic systems. *The International Journal of Robotics Research (IJRR)*, 30:574–589, Apr. 2011.

[21] A. Prorok, A. M. Hsieh, and V. Kumar. Adaptive Redistribution of a Swarm of Heterogeneous Robots. *Special Issue for the Workshop on On-line decision-making in multi-robot coordination, Acta Polytechnica; to appear*, 2016.

[22] A. Prorok, A. M. Hsieh, and V. Kumar. Formalizing the Impact of Diversity on Performance in a Heterogeneous Swarm of Robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[23] A. Prorok, M. A. Hsieh, and V. Kumar. Fast Redistribution of a Swarm of Heterogeneous Robots. In *International Conference on Bio-inspired Information and Communications Technologies*, 2015.

[24] O. Shehory and S. Kraus. A kernel-oriented model for autonomous-agent coalition-formation in general environments. In *Distributed Artificial Intelligence Architecture and Modelling*, pages 31–45. Springer Berlin Heidelberg, Berlin, Heidelberg, June 2005.

[25] G. W. Stewart. *Matrix Algorithms I. Basic Decompositions*. SIAM, 1998.

[26] P. Tokekar, J. Vander Hook, D. Mulla, and V. Isler. Sensor Planning for a Symbiotic UAV and UGV system for Precision Agriculture. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5321–5326, 2013.

[27] D. J. Watts and S. H. Strogatz. Collective dynamics of ‘small-world’ networks. *Nature*, 393:440–442, 1998.