Adaptive Distribution of a Swarm of Heterogeneous Robots

Amanda Prorok, M. Ani Hsieh, Vijay Kumar

Workshop on On-line decision-making in multi-robot coordination
IROS 2015
Introduction

How do we design *heterogeneous* multi-robot systems to maximize performance?

**Diversity Metric**

**Design Paradigm**

* Image credits: M. Egerstedt, Georgia Tech
Examples

One robot type cannot cater to all aspects of a task

Collaborative Perception

Collaborative Manipulation

Idea: A task needs certain capabilities
Approach

**Robot community**
- Species
- Binary traits

**Tasks**
- Need traits
- Switching

---

![Diagram showing the relationship between robot community and tasks](image)
Problem Formulation

How do we redistribute a heterogeneous team of robots?

Redistribution of traits (capabilities) among tasks
We consider a community of work, we assume that the tasks have been encoded through binary characteristics that represent the skill or other trait might consider the capability of fitting an sensor, such as a camera or laser range finder. An attempt to develop our formalism, we will borrow terminology via a directed graph, \( \sum \) of edges, \( q \) vertices, and implicitly solve the combinatorial solutions (which is the case in the works presented by [8]).

Each robot belongs to a given capability is needed for a given task, irrespective of the terrain.

\[ \mathbf{Y}(t) = \mathbf{X}(t) \cdot \mathbf{Q} \]

\( \mathbf{Q} \) is defined as species-traits matrix, \( \mathbf{Y} \) as trait distribution, \( \mathbf{X} \) as robot distribution.

\[ \mathbf{Q} = \begin{pmatrix}
1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0
\end{pmatrix} \]
Method

\[
\frac{dx^{(s)}}{dt} = K^{(s)}x^{(s)}
\]

— for a large number of robots, model system as ODE

\[
K^{(s)}
\]

— transition rates for each species

\[
Y(t) = X(t) \cdot Q
\]

— system

\[
Y(t) = \sum_{s=1}^{S} e^{K^{(s)}t} x^{(s)}_0 \cdot q^{(s)}
\]

— solution to the ODE
Method

\[ E = Y^\star - \sum_{s=1}^{S} e^{K^{(s)} \cdot \tau} x_0^{(s)} \cdot q^{(s)} \]  

— error in trait distribution

1. minimize \( J^{(1)} = \| E \|_F^2 \)  

— basic optimization problem

2. minimize \( J^{(2)} = J^{(1)} + \alpha \tau^2 \)  

— explicit opt. of convergence time

3. minimize \( J^{(3)} = J^{(2)} + \beta \sum_{s=1}^{S} \left\| e^{K^{(s)} \cdot \tau} x_0^{(s)} - e^{K^{(s)} (\tau + \nu)} x_0^{(s)} \right\|_2^2 \)  

— reinforcing steady-state
We present a method that distributes a swarm of heterogeneous robots among a set of tasks with the goal of satisfying different initial conditions. To illustrate our method, we consider an example of a team of 800 robots switching between eight tasks. The simulation was run with 800 robots.

The shaded area shows the standard deviation. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.

The optimal transition rates, we pose an optimization problem. We consider the optimization problem posed in Eq. 8. We note that in practical applications, completing the optimization explicitly optimizing for the steady-state, it reaches a steady-state distribution. We measure the time for each graph. We set \( \mu = 2 \) s for 40 random graphs with different initial conditions. The boxplots show the average errors at steady-state.
Experiment

initial \rightarrow target

* Work submitted to ICRA 2016
CONTROLLING DIVERSITY TO MAXIMIZE PERFORMANCE IN A HETEROGENEOUS SWARM OF ROBOTS

Amanda Prorok
M. Ani Hsieh
Vijay Kumar
Continuous Optimization

Fixed K: \( \mathbf{K}^{(s)} \star, \tau \star = \arg \min_{\mathbf{K}^{(s)}, \tau} \mathcal{J}^{(3)} \)

Adaptive K: \( \mathbf{K}^{(s)}(t), \tau \star(t) = \arg \min_{\mathbf{K}^{(s)}, \tau} \mathcal{J}^{(3)}(\mathbf{X}(t_p)) \)

\[ \mathbf{X}(t_p) \]
Results

![Graph showing the ratio of misplaced traits over time for different optimization methods. The graph compares Macroscopic, Adaptive Micro., and Fixed Micro. methods, with a clear indication of the performance of each method over time. The y-axis represents the ratio of misplaced traits, while the x-axis represents time in seconds. The Macroscopic method shows the steepest decline, followed by Adaptive Micro., and then Fixed Micro., indicating a faster convergence to a desirable trait distribution.]
How hard is it to redistribute the robot community as a function of its diversity?

initial  target
Effects of Diversity

\[
\text{rank}(Q) = S \\
Q = \begin{pmatrix}
1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0
\end{pmatrix}
\]

All species are independent

\[
\text{rank}(Q) < S \\
Q = \begin{pmatrix}
0 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1
\end{pmatrix}
\]

There are dependent species
When several are possible that satisfy Eq. 7. The importance of finding a good final robot distribution converges to a value and Fig. 4(b) shows results for the current state of the robot distribution, and to adapt the algorithm so that this method outperforms the other two methods.

Adaptive-NC

\[ \text{rank}(Q) = S \]

Fixed-NC

\[ \text{rank}(Q) < S \]

Adaptive-NC

\[ \text{rank}(Q) = S \]

Fixed-NC

\[ \text{rank}(Q) < S \]

Comparison of time to convergence for different methods: Benchmark, Fixed, and Adaptive. The box plots show the median, 25th and 75th percentiles, and the error bars show the standard deviation. The results demonstrate that, despite the fact that our method minimizes the error with respect to the metric in Eq. 29. The errorbars show the standard deviation of the results for each method.
Conclusions

- Model for heterogeneous robot system
- Efficient optimization algorithm
- Formulation for adaptive control
- Real robot experiments
- Effects of diversity

Further work:
- Automatic generation of task requirements
- Continuous trait instantiations
- Foundations of diversity
Thank you for your attention.

prorok@seas.upenn.edu