# Mobile Robot Systems

## Lecture 8: Multi-Robot Systems - Task Allocation

Dr. Amanda Prorok

asp45@cam.ac.uk

www.proroklab.org

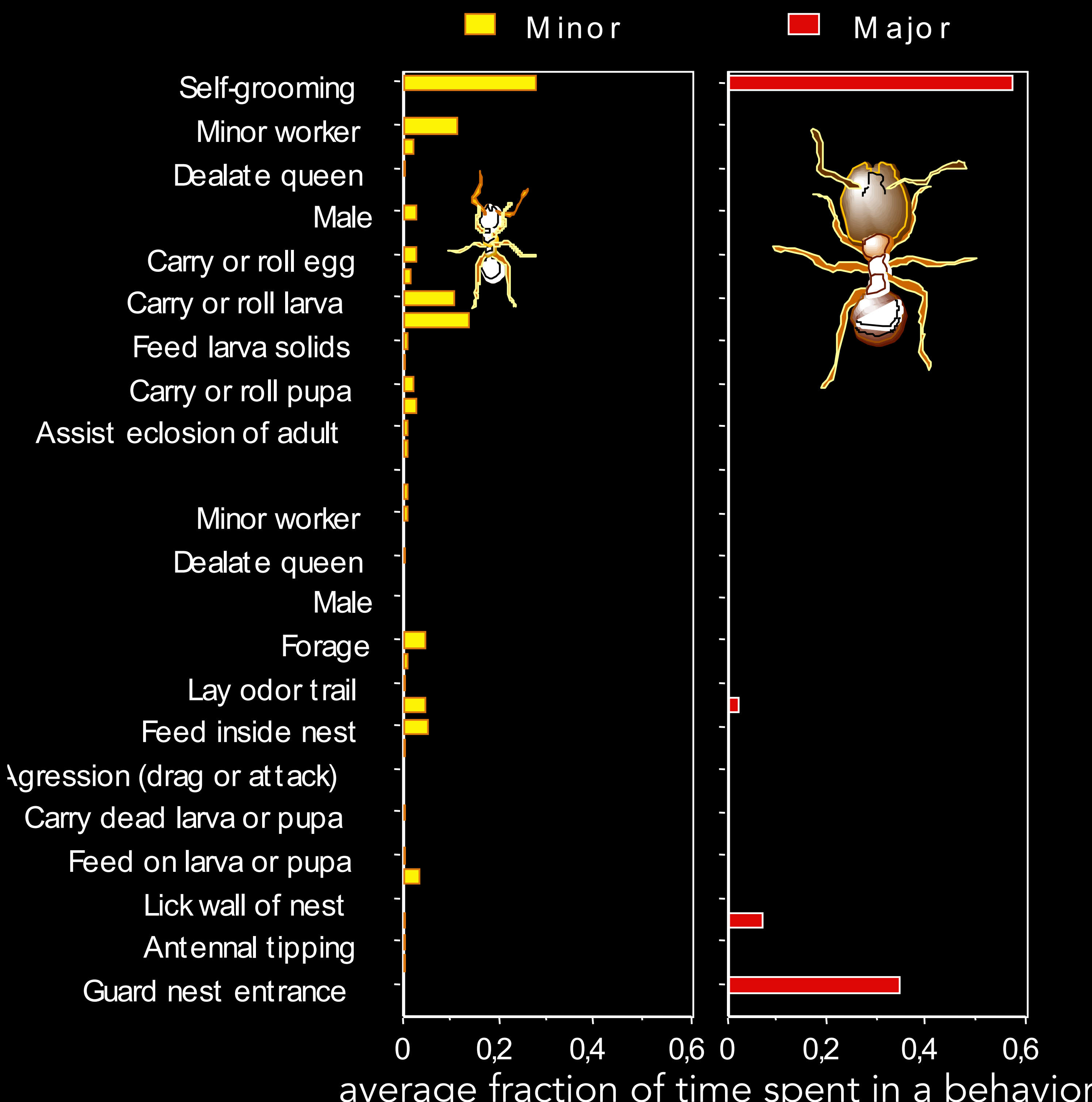
UNIVERSITY OF CAMBRIDGE

# In this Lecture

- Motivation: task allocation in nature
- Assignment algorithms:
    - Hungarian method
    - Swarm distribution mechanisms
    - Market-based
    - Threshold-based
- Credit:
    - Threshold-based example from A. Martinoli's course at EPFL

# Task Allocation vs. Division of Labor

## In **nature**: physical castes

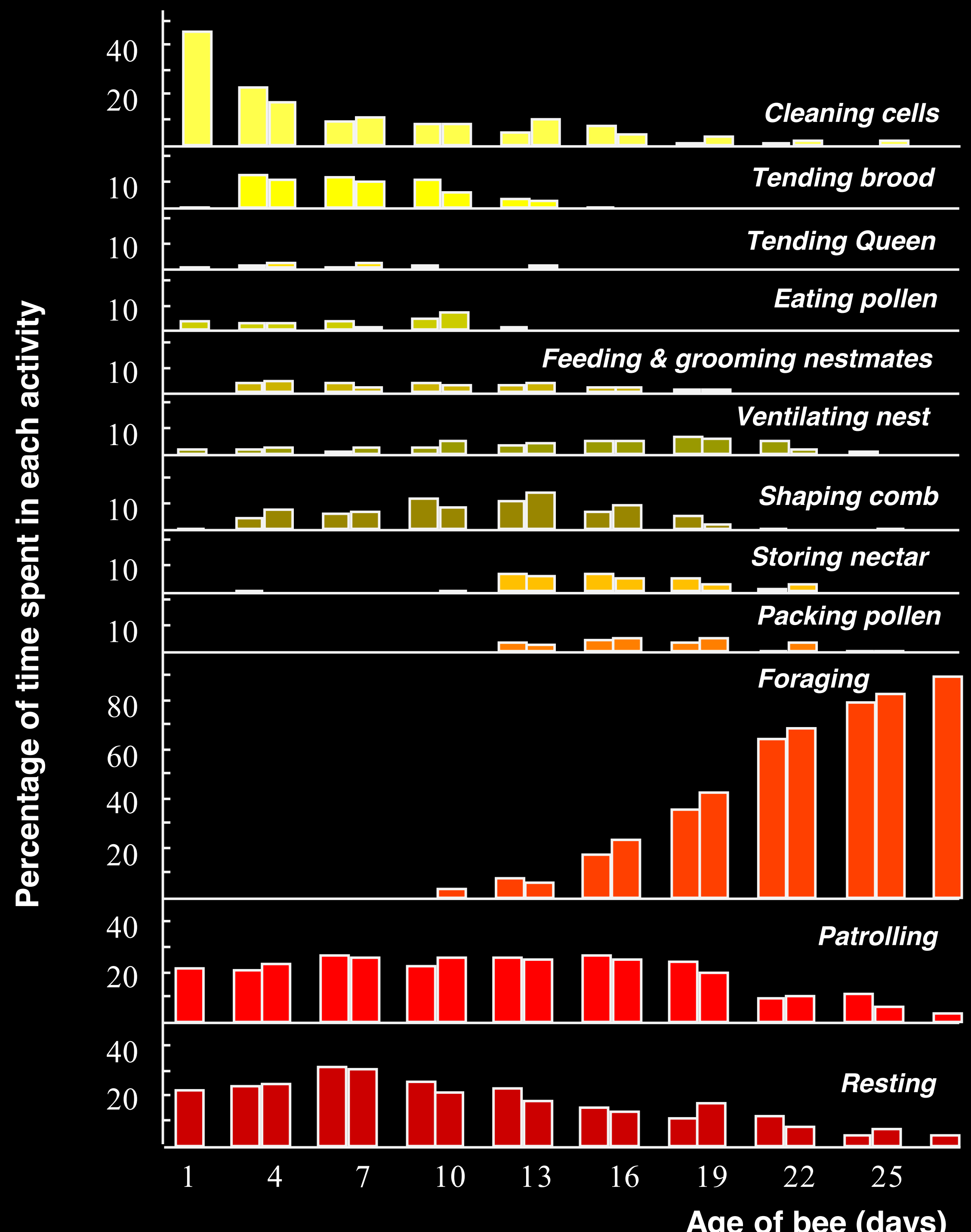



Behavioral repertoire of majors and minors: In *Pheidole guilelmimuelleri* the minors show ten times as many different basic behaviors as the majors.

*image credit: Alcherio Martinoli

# Task Allocation vs. Division of Labor

In **nature**: temporal polyethism



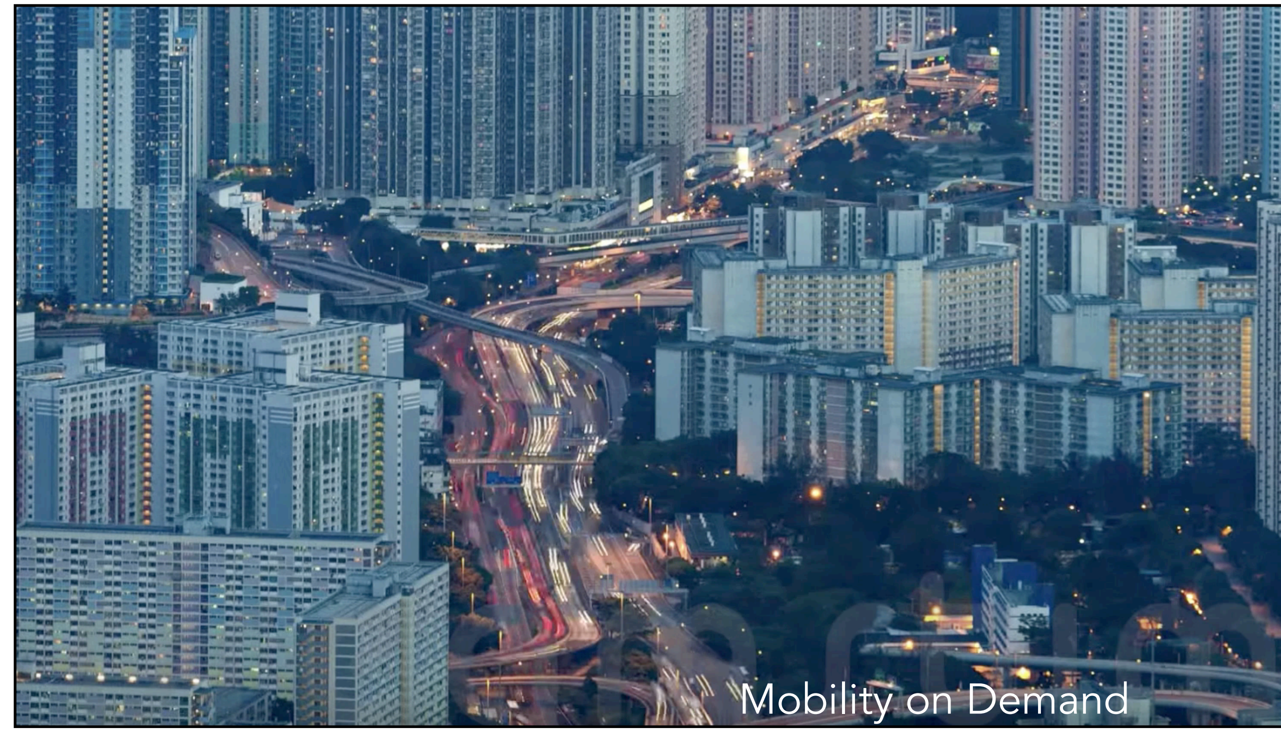

Behavioral change in worker bees as a function of age; young individuals work on internal tasks (brood care and nest maintenance), older workers forage for food and defend the nest.

*image credit: Alcherio Martinoli

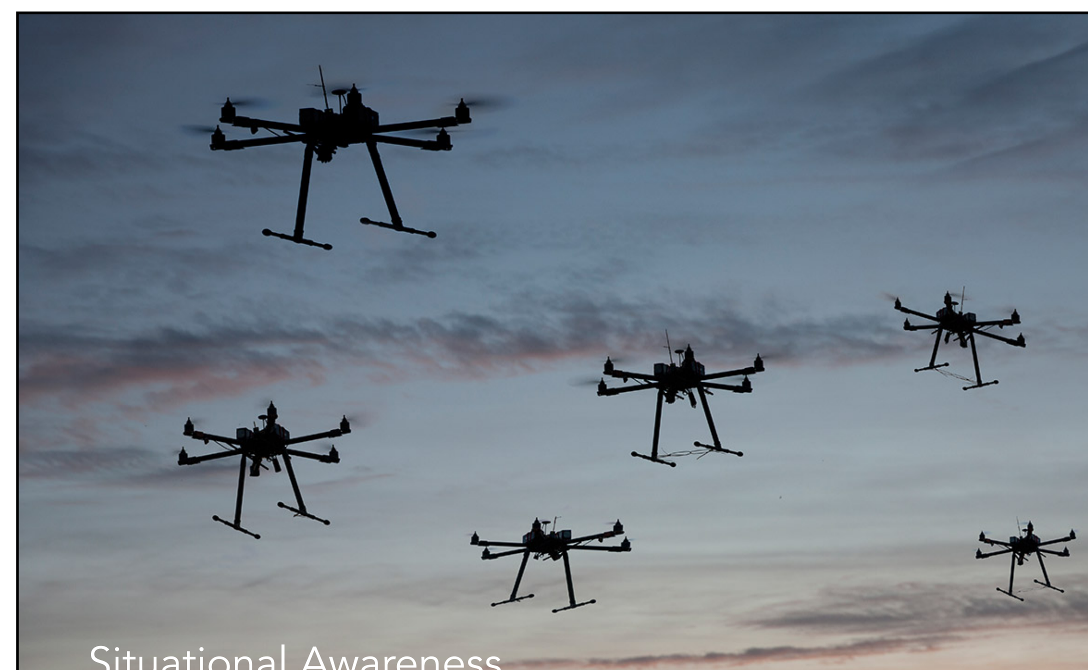
# Task Allocation vs Division of Labor

In robotics:

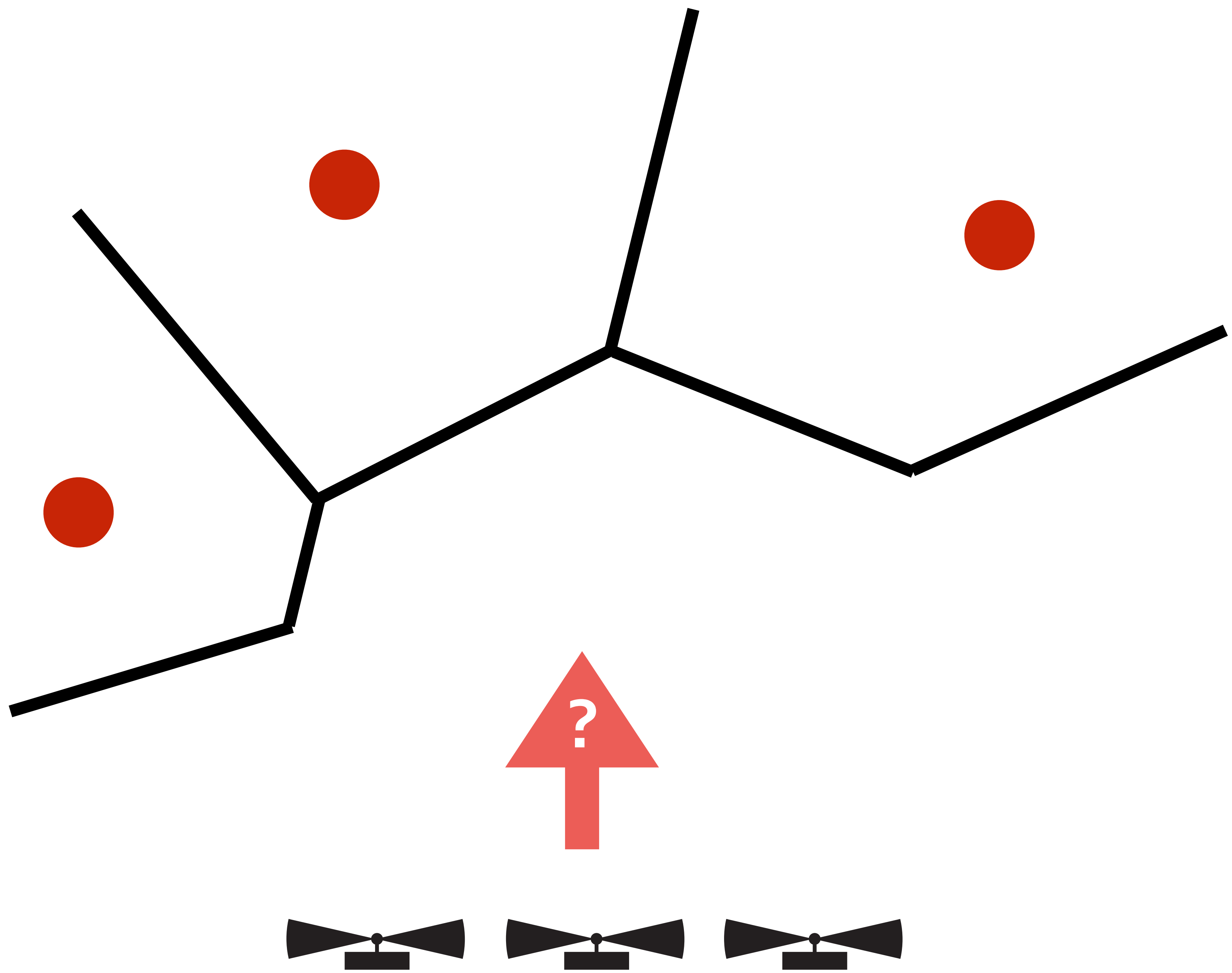

Monitoring


Mobility on Demand

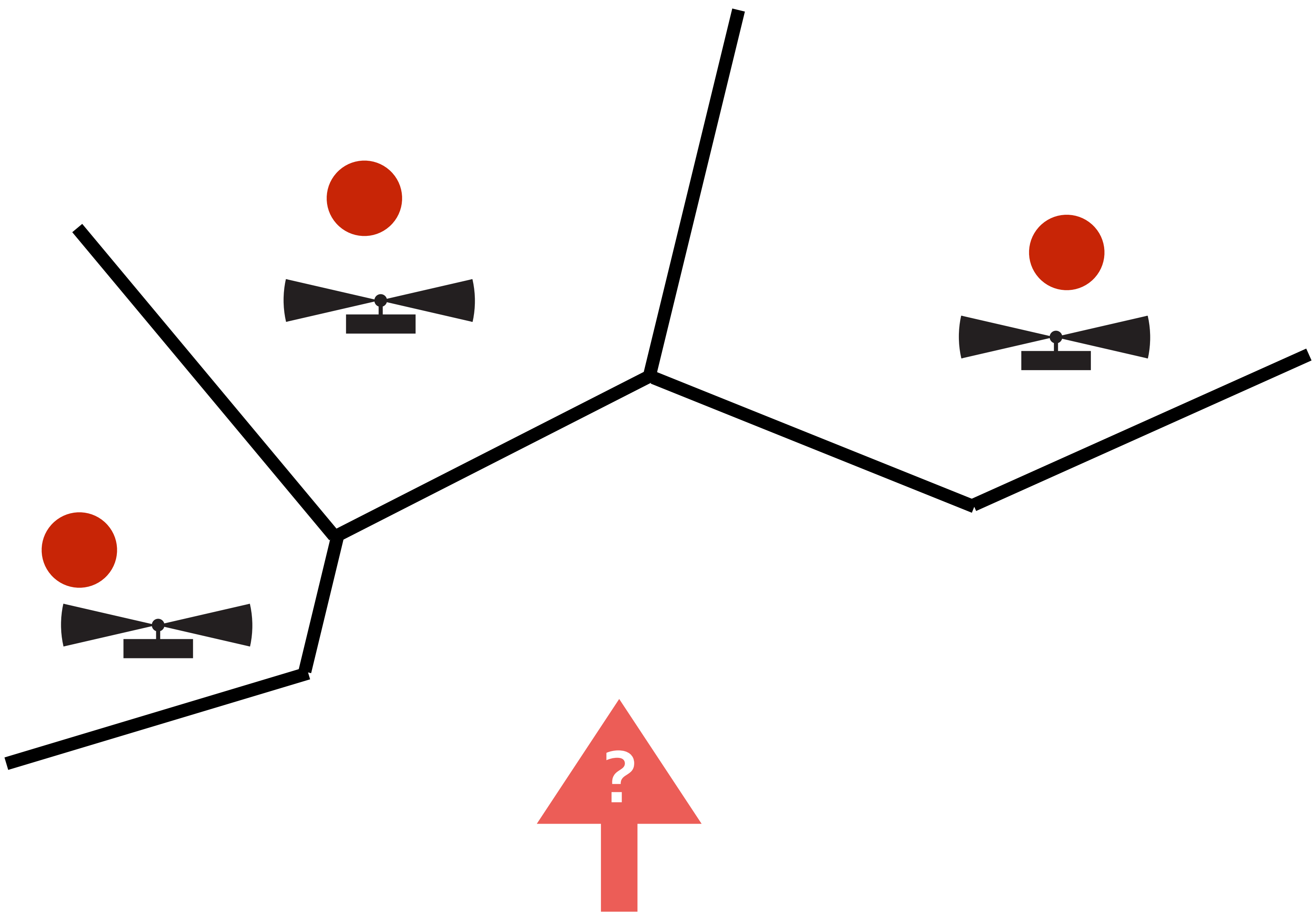
Warehousing and Product Delivery


Situational Awareness

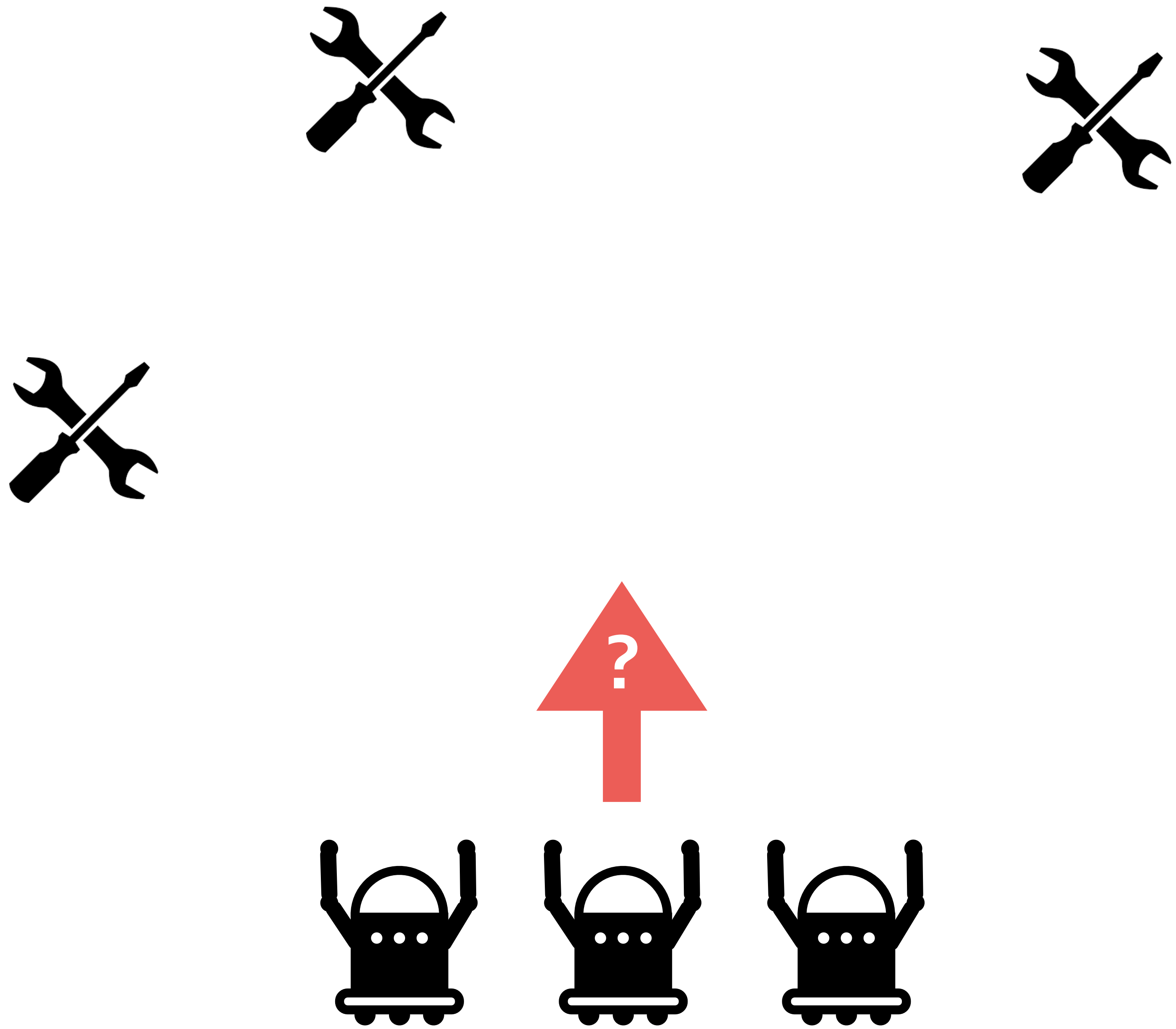
# Assignment Problems



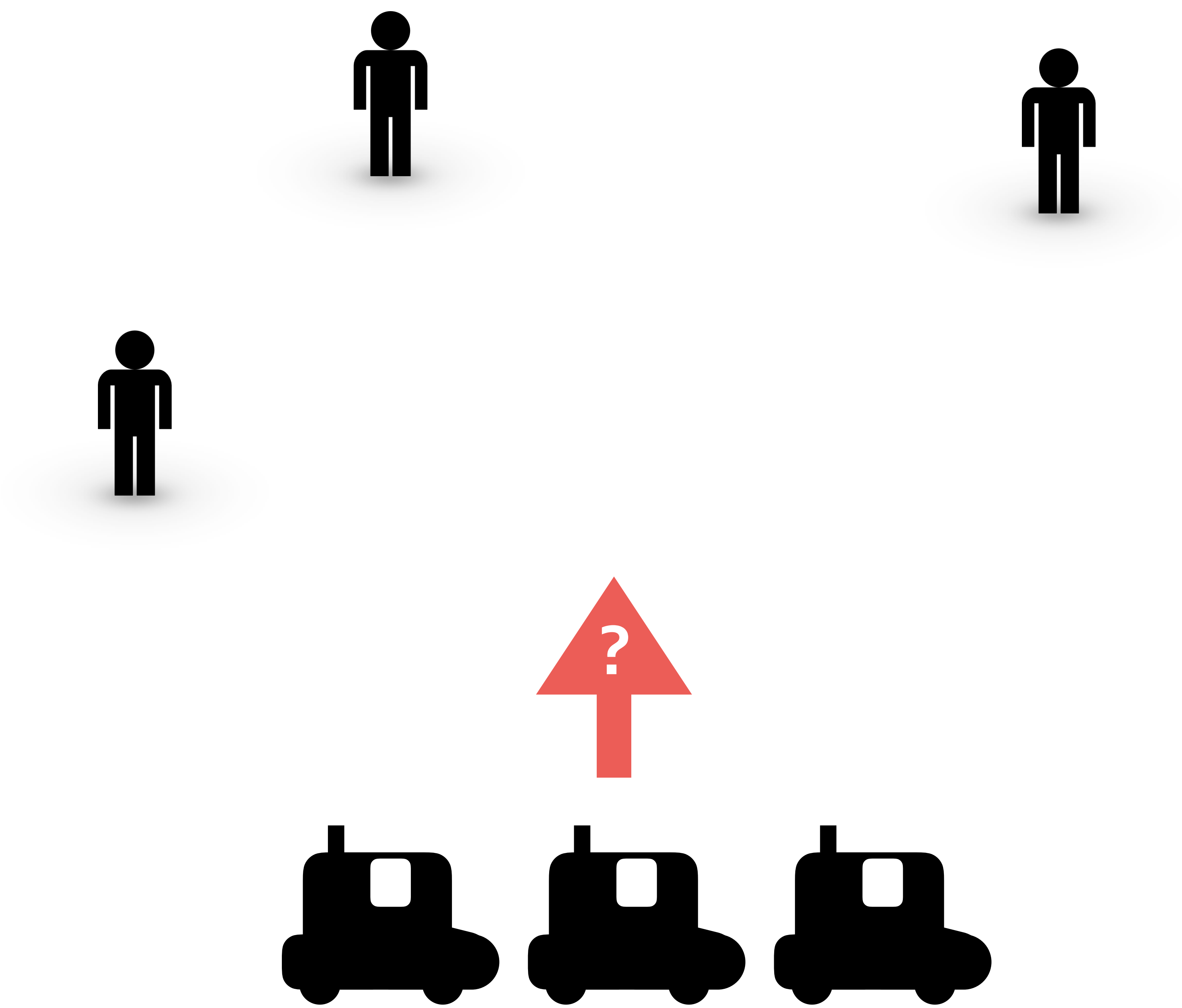
# Assignment Problems



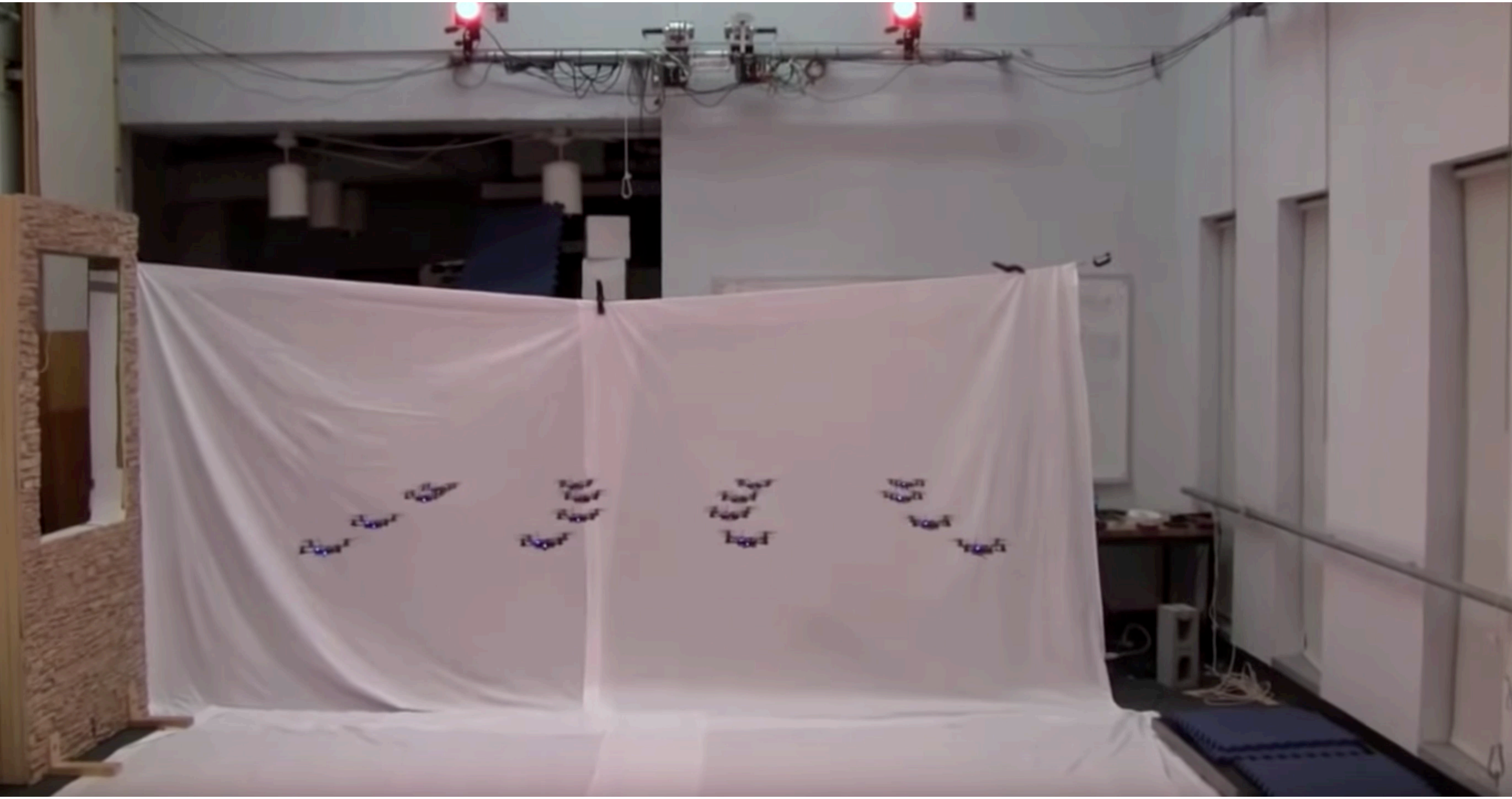
# Assignment Problems



# Assignment Problems



# Assignment Problems



[Kumar et al.; UPenn]

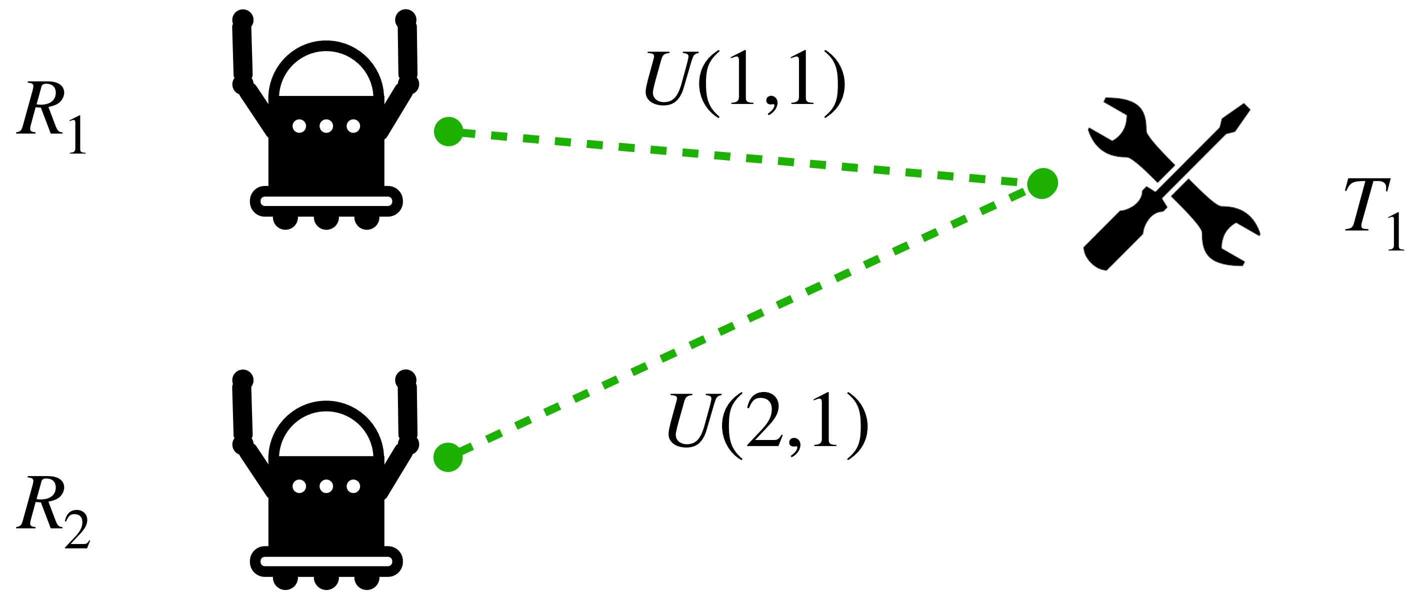
# The Assignment Problem

- What is a **task**?
  - Discrete: e.g., pickup parcel X from location Y, …
  - Continuous: e.g., monitor building X, search area Y…
  - Key assumption: task independence
    (dependent tasks $\longrightarrow$ *scheduling*)
- Assignment methods are drawn from multiple fields: operations research, economics, scheduling, network flows, combinatorial optimization.
- Classical problem formulation: bipartite graph matching

# The Assignment Problem

- What is to be optimized? **Utility:** an individual robot knows the value of executing a certain action.
- Utility, depending on context: value, cost, fitness. Knowing the true (exact) utility is key to finding an optimal assignment.
- Various formulations exist. For example:

$$U(R,T) = \begin{cases} Q_{RT} - C_{RT} & \text{if } R \text{ is capable of executing } T \text{ and } Q_{RT} > C_{RT} \\ 0 & \text{otherwise} \end{cases}$$



# The Linear Assignment Problem

- In an optimal assignment problem, maximize the system performance:

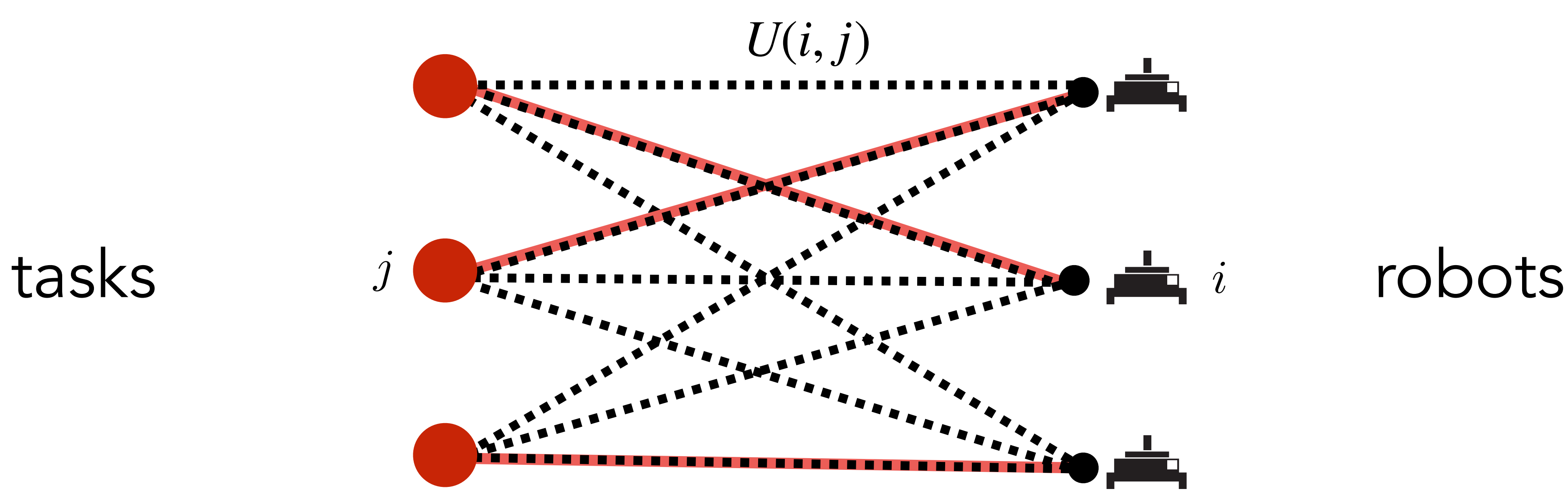
find $x_{ij}$ that maximize:

$$\mathcal{U} = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij}\ U(i,j)$$

subject to

$$\sum_{i=1}^{m} x_{ij} = 1, \quad 1 \leq j \leq n$$

$$\sum_{j=1}^{m} x_{ij} = 1, \quad 1 \leq i \leq m$$




bipartite perfect matching (complete graph)

# The Hungarian Algorithm

- Published by Kuhn in 1955, based on the earlier works of two Hungarian mathematicians: Dénes Kőnig and Jenő Egerváry.
  - $O(n^3)$ running time is possible.
- Steps (input is an *n x n* by matrix with non-negative elements):
  - **Step 1**: Subtract row minima; For each row, find the lowest element and subtract it from each element in that row.
  - **Step 2**: Subtract column minima; Similarly, for each column, find the lowest element and subtract it from each element in that column.
  - **Step 3**: Cover all zeros with a minimum number of lines; Cover all zeros in the resulting matrix using a minimum number of horizontal and vertical lines. If *n* lines are required, an optimal assignment exists among the zeros. The algorithm stops. If less than *n* lines are required, continue with Step 4.
  - **Step 4**: Create additional zeros; Find the smallest element (call it *k*) that is not covered by a line in Step 3. Subtract *k* from all uncovered elements, and add *k* to all elements that are covered twice. Go to Step 3.

# The Hungarian Algorithm - Example

Step 0: robot-task assignment costs

| | T1 | T2 | T3 | T4 |
|---|---|---|---|---|
| R1 | 82 | 83 | 69 | 92 |
| R2 | 77 | 37 | 49 | 92 |
| R3 | 11 | 69 | 5 | 86 |
| R4 | 8 | 9 | 98 | 23 |

Step 1: subtract row minima

| | T1 | T2 | T3 | T4 | |
|---|---|---|---|---|---|
| R1 | 13 | 14 | 0 | 23 | -69 |
| R2 | 40 | 0 | 12 | 55 | -37 |
| R3 | 6 | 64 | 0 | 81 | -5 |
| R4 | 0 | 1 | 90 | 15 | -8 |

Step 2: subtract column minima

| | T1 | T2 | T3 | T4 |
|---|---|---|---|---|
| R1 | 13 | 14 | 0 | 8 |
| R2 | 40 | 0 | 12 | 40 |
| R3 | 6 | 64 | 0 | 66 |
| R4 | 0 | 1 | 90 | 0 |
| | -0 | -0 | -0 | -15 |

Step 3: cover all zeros with a minimum of lines

| | T1 | T2 | T3 | T4 |
|---|---|---|---|---|
| R1 | 13 | 14 | 0 | 8 |
| R2 | 40 | 0 | 12 | 40 |
| R3 | 6 | 64 | 0 | 66 |
| R4 | 0 | 1 | 90 | 0 |

3 lines found

Step 4: create additional zeros

| | T1 | T2 | T3 | T4 |
|---|---|---|---|---|
| R1 | 13 | 14 | 0 | 8 |
| R2 | 40 | 0 | 12 | 40 |
| R3 | 6 | 64 | 0 | 66 |
| R4 | 0 | 1 | 90 | 0 |

-6: unmarked elements
+6: twice marked elements

| | T1 | T2 | T3 | T4 |
|---|---|---|---|---|
| R1 | 7 | 8 | 0 | 2 |
| R2 | 40 | 0 | 18 | 40 |
| R3 | 0 | 58 | 0 | 60 |
| R4 | 0 | 1 | 96 | 0 |

Step 3: cover all zeros with a minimum of lines

| | T1 | T2 | T3 | T4 |
|---|---|---|---|---|
| R1 | 7 | 8 | 0 | 2 |
| R2 | 40 | 0 | 18 | 40 |
| R3 | 0 | 58 | 0 | 60 |
| R4 | 0 | 1 | 96 | 0 |

4 lines found

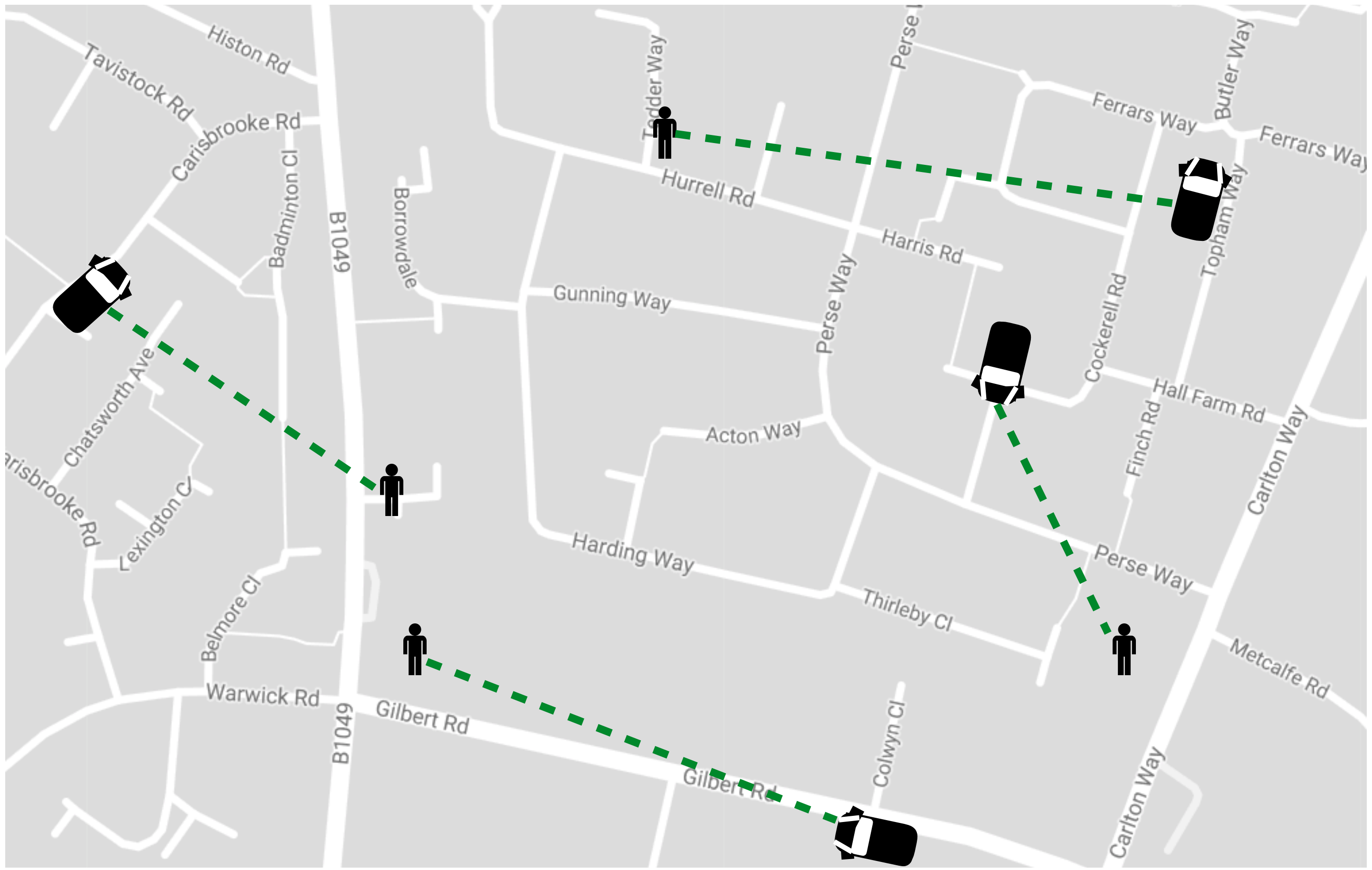
Stop: An optimal assignment exists.

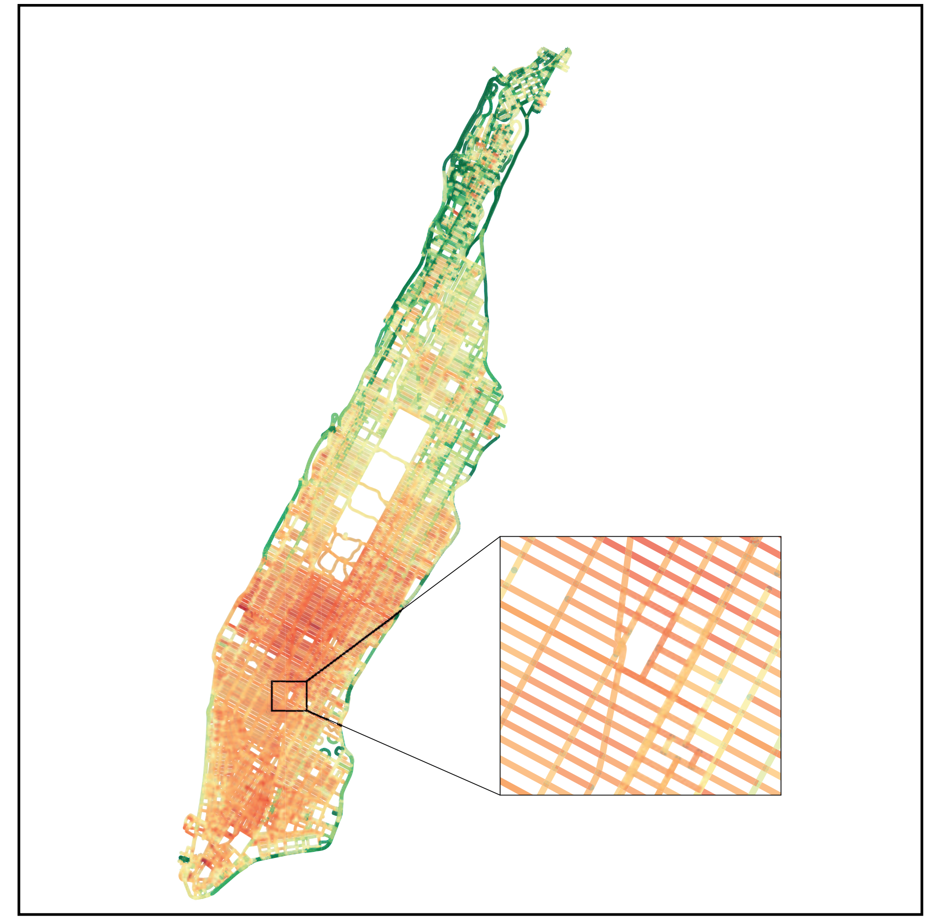
| | T1 | T2 | T3 | T4 |
|---|---|---|---|---|
| R1 | 7 | 8 | 0 | 2 |
| R2 | 40 | 0 | 18 | 40 |
| R3 | 0 | 58 | 0 | 60 |
| R4 | 0 | 1 | 96 | 0 |

unique, optimal assignment found

*Example from www.hungarianalgorithm.com

# Application: Vehicle-to-Passenger Assignment



Goal: find optimal assignment matrix **A***

$$\mathbf{A}^\star = \underset{\mathbf{A}}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{j=1}^{M} c_{ij} a_{ij}$$

Publicly available data:

- OpenStreetMap for whole area
- Convert to graph (4302 vertices, 9414 edges)
- Cost of an assignment ~ distance (time)
- NYC public taxicab dataset

# The Hungarian Algorithm

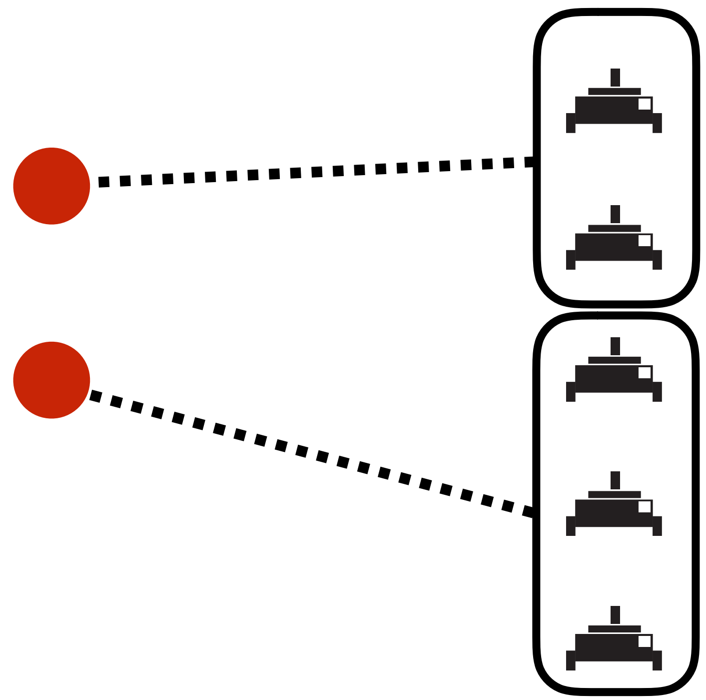
- Assumptions when using an assignment algorithm such as the Hungarian method:
  - Costs (utilities) are known at a **centralized** computation unit.
  - Costs (utilities) are **deterministic** (no noise).
  - Costs (utilities) do not change (**constant**).
  - **1-to-1** assignment (one robot per task, one task per robot).
- Complications:
  - Uncertainty around true utility $U(i,j)$
  - Dynamic environment (changes in utility / agents)
  - Robot / task dependencies (robot heterogeneity / redundancy).

*all of these issues are very common in robotics!!*

- Consequences:
  - Sub-optimality
  - Problems can become NP-hard (for combinatorial matching problems)
  - Practically infeasible (centralized solutions may not be possible)

# Assignment of Robot Coalitions

Some tasks require more than 1 robot.



How many ways to partition *n* robots into *k* non-empty subsets?

Given by the *Stirling number of the second kind.*

E.g.: Ten robots, 5 tasks: $S(10,5) = 42'525$

# Assignment of Robot Coalitions

The problem of forming **robot coalitions**:

$\boldsymbol{E}$ is the ground set (all robots) and $\boldsymbol{X}$ is a family of subsets.

$$y \bigcap z = \varnothing \quad \forall y, z \in X, y \neq z$$ robot subsets are mutually disjoint

$$\bigcup_{x \in X} = E$$ the union of subsets is equivalent to the ground set.

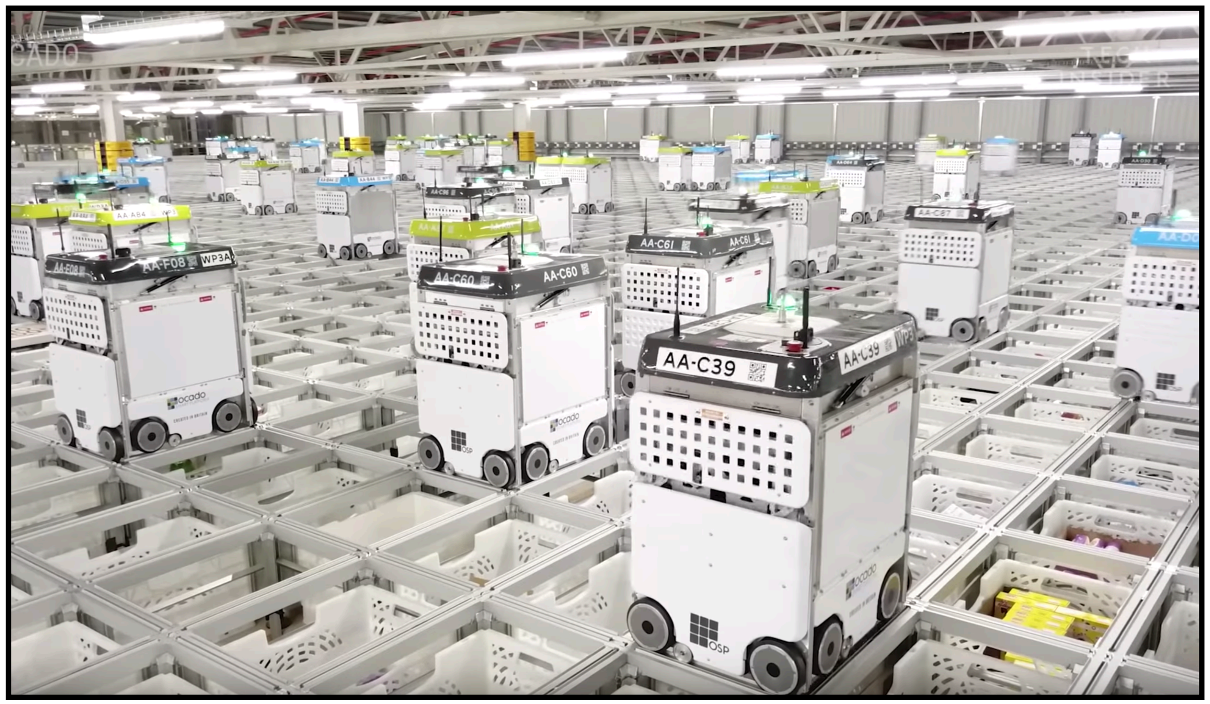
**Set Partitioning Problem:** Given a finite set $E$, a family $F$ of acceptable subsets of $E$, and a utility function $u : F \mapsto \mathbb{R}_+$ , find a maximum-utility family $X$ of elements in $F$ such that $X$ is a partition of $E$.

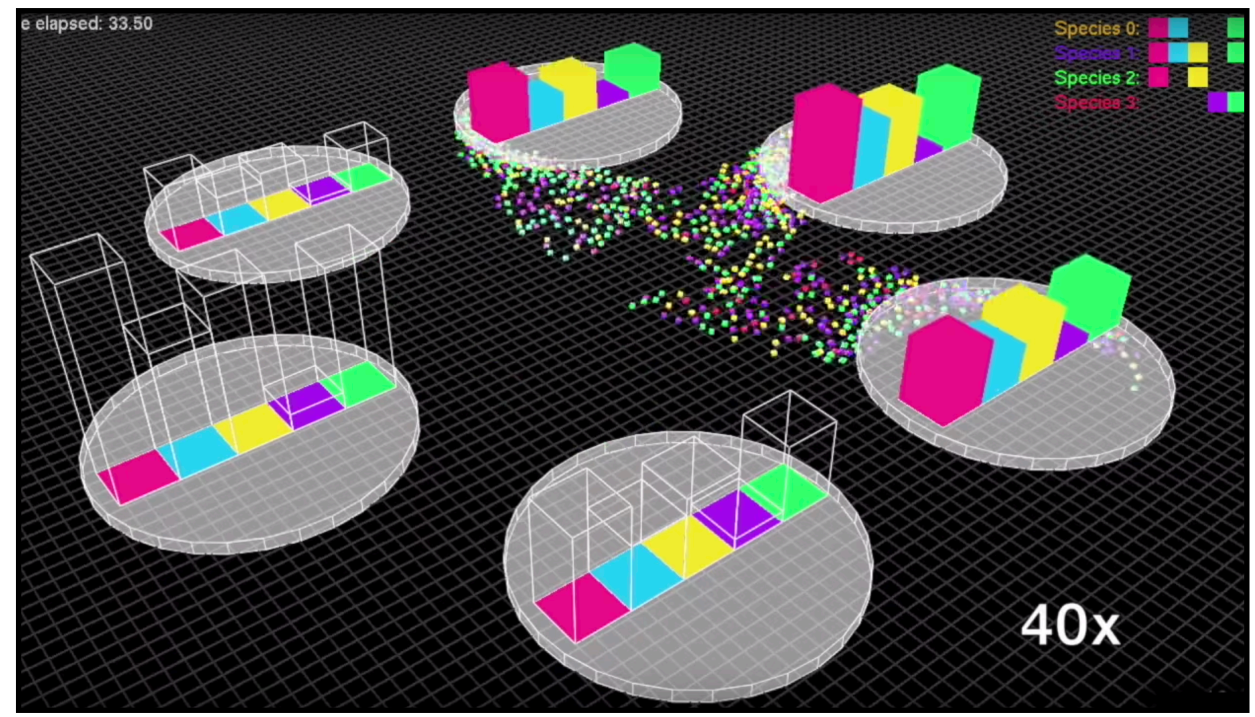
The set-partitioning problem is *strongly NP-hard*. [Garey and Johnson; 1978]

… One potential solution: *relaxation of the problem to the continuous domain.*

# Countable vs Uncountable Systems

- Difference between a multi-robot system and a robot swarm?
- Swarms are larger, but how large…?
- The method is the key!



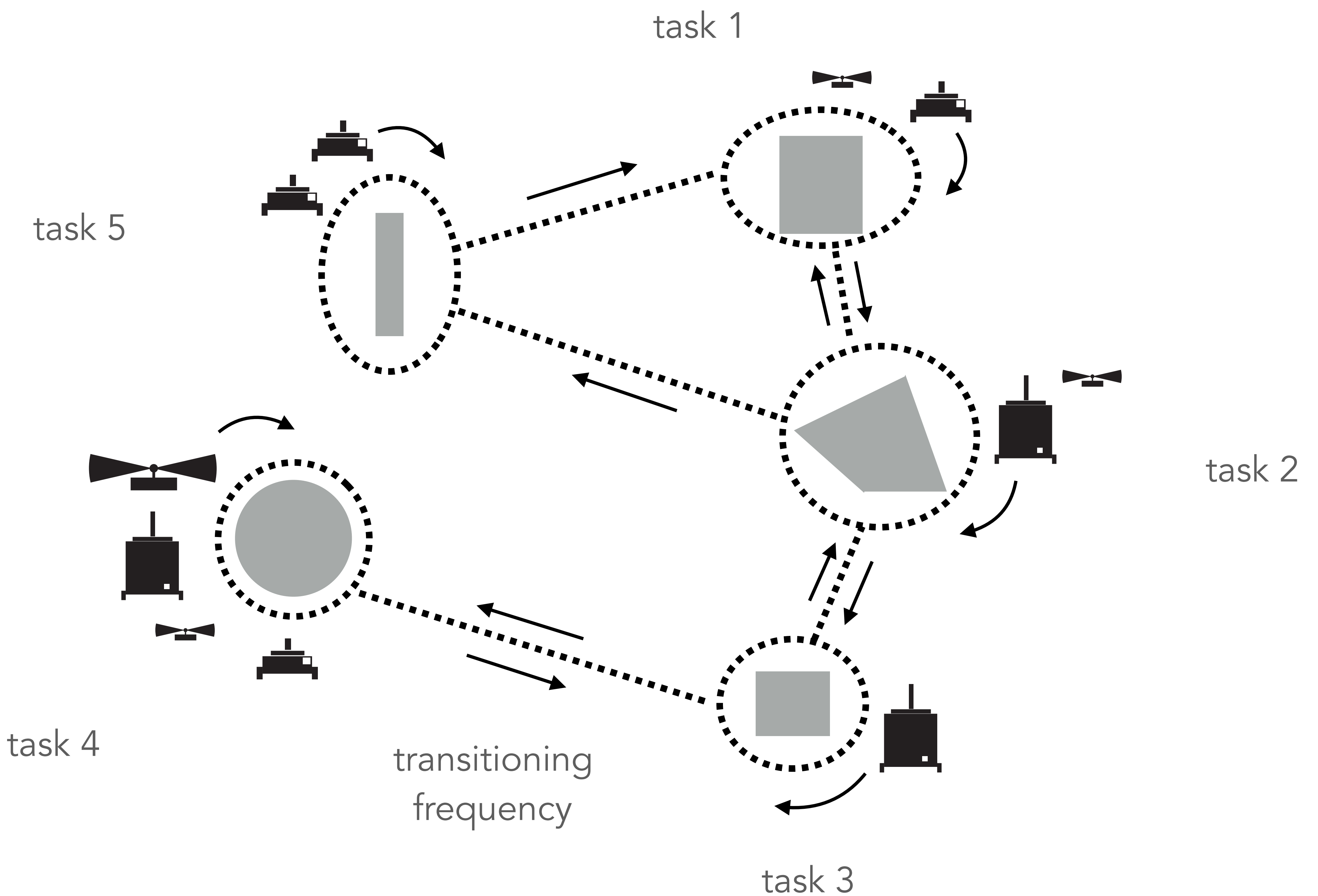




- robot-to-task allocation
- method: **combinatorial approach**
- exact, but computationally demanding

- redistribution of robots among tasks
- method: **mean-field approach**
- approximative, but fast

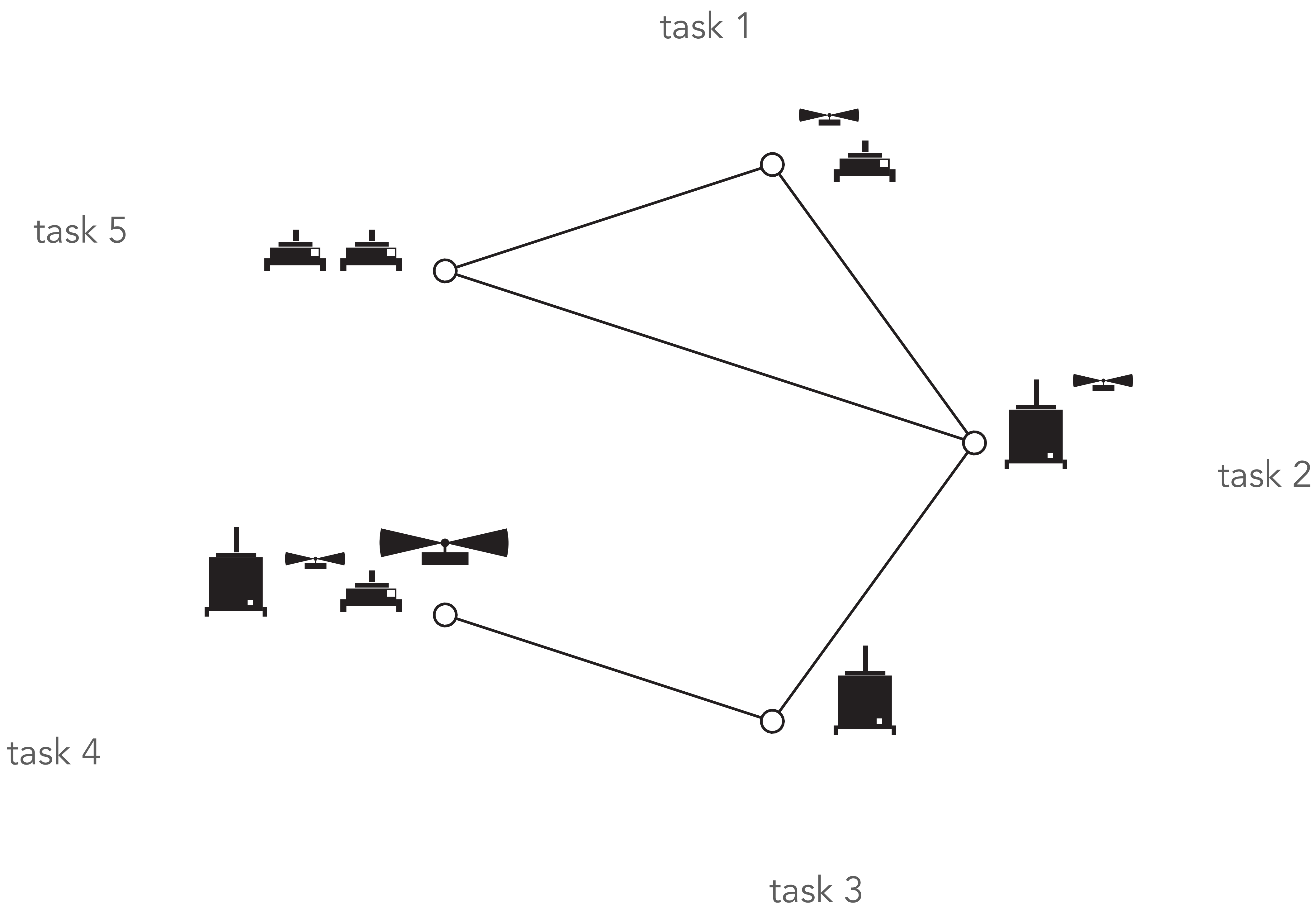
# Redistribution of a Swarm of Robots

Example: monitor geographical sites



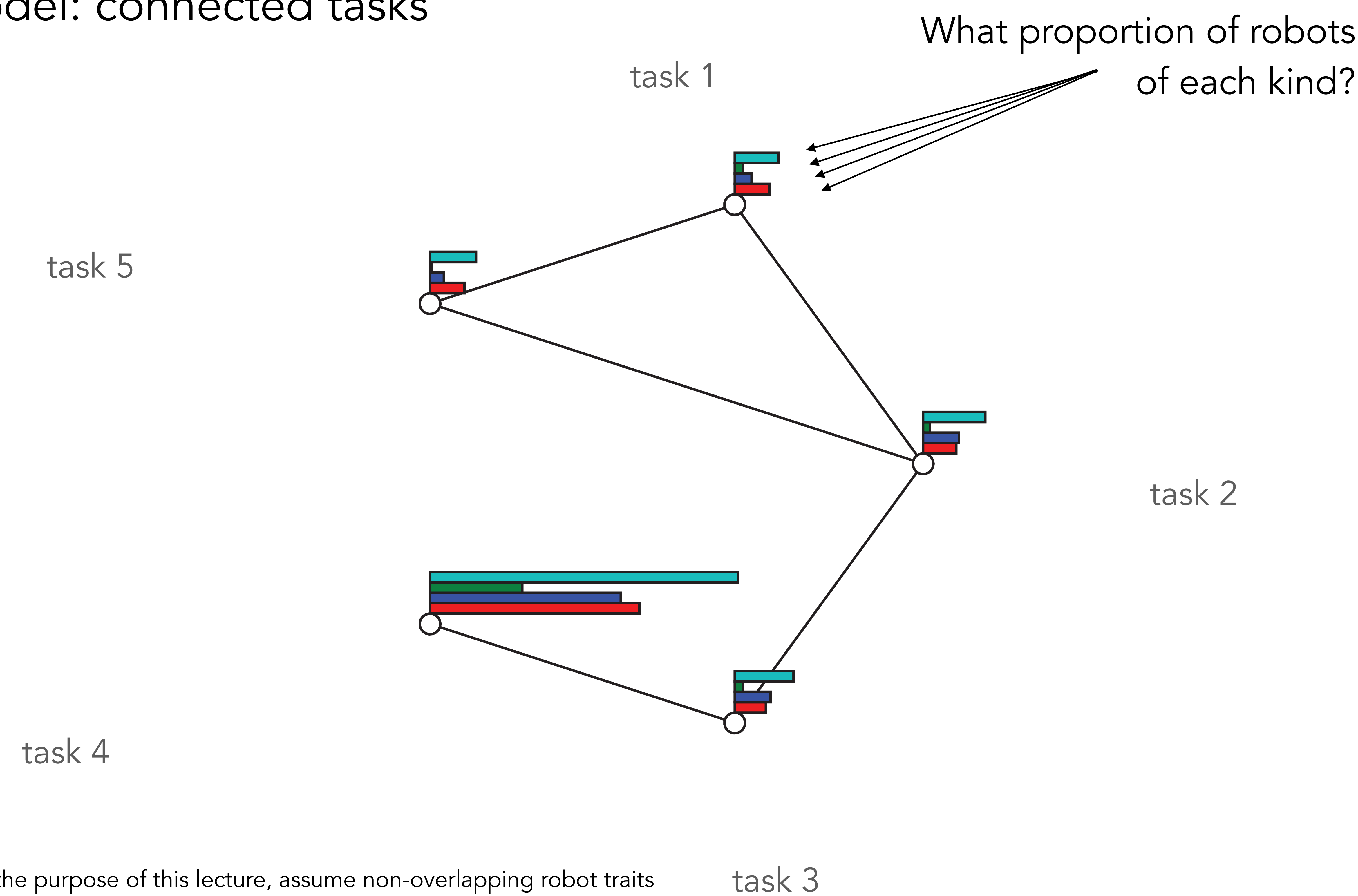
# Redistribution of a Swarm of Robots

Model: connected tasks



# Redistribution of a Swarm of Robots

## Model: connected tasks




*note: for the purpose of this lecture, assume non-overlapping robot traits

# Redistribution of a Swarm of Robots

Insight: we can model the distribution dynamics of the robot swarm as a linear dynamical system!

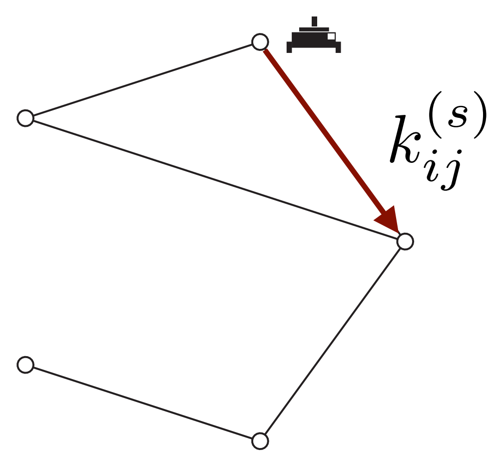
System state, e.g.: $\mathbf{x} = [0.3, 0.2, 0.1, 0.1, 0.3]^\mathsf{T}$

proportion of swarm at task 1

transition rate matrix

distribution of robots over tasks

Distribution dynamics: $$\underbrace{\dot{\mathbf{x}}^{(s)}}_{\substack{\text{change in distribution of} \\ \text{robots of type } (s) \text{ over tasks}}} = \underbrace{\mathbf{K}^{(s)}}_{\substack{\text{rates} \\ \text{M x M}}} \underbrace{\mathbf{x}^{(s)}}_{\substack{\text{robots} \\ \text{M x 1}}}$$




*(s)*: robot species

Note: if matrix $\mathbf{K}$ has certain properties, this system is stable.

# Redistribution of a Swarm of Robots

Robot distribution dynamics: $$\dot{\mathbf{x}}^{(s)} = \underbrace{\mathbf{K}^{(s)}}_{\substack{\text{rates}\\ \text{M x M}}} \underbrace{\mathbf{x}^{(s)}}_{\substack{\text{robots}\\ \text{M x 1}}}$$

Solution: $$\mathbf{x}^{(s)}(t) = e^{\mathbf{K}^{(s)} t}\, \mathbf{x}_0^{(s)}$$

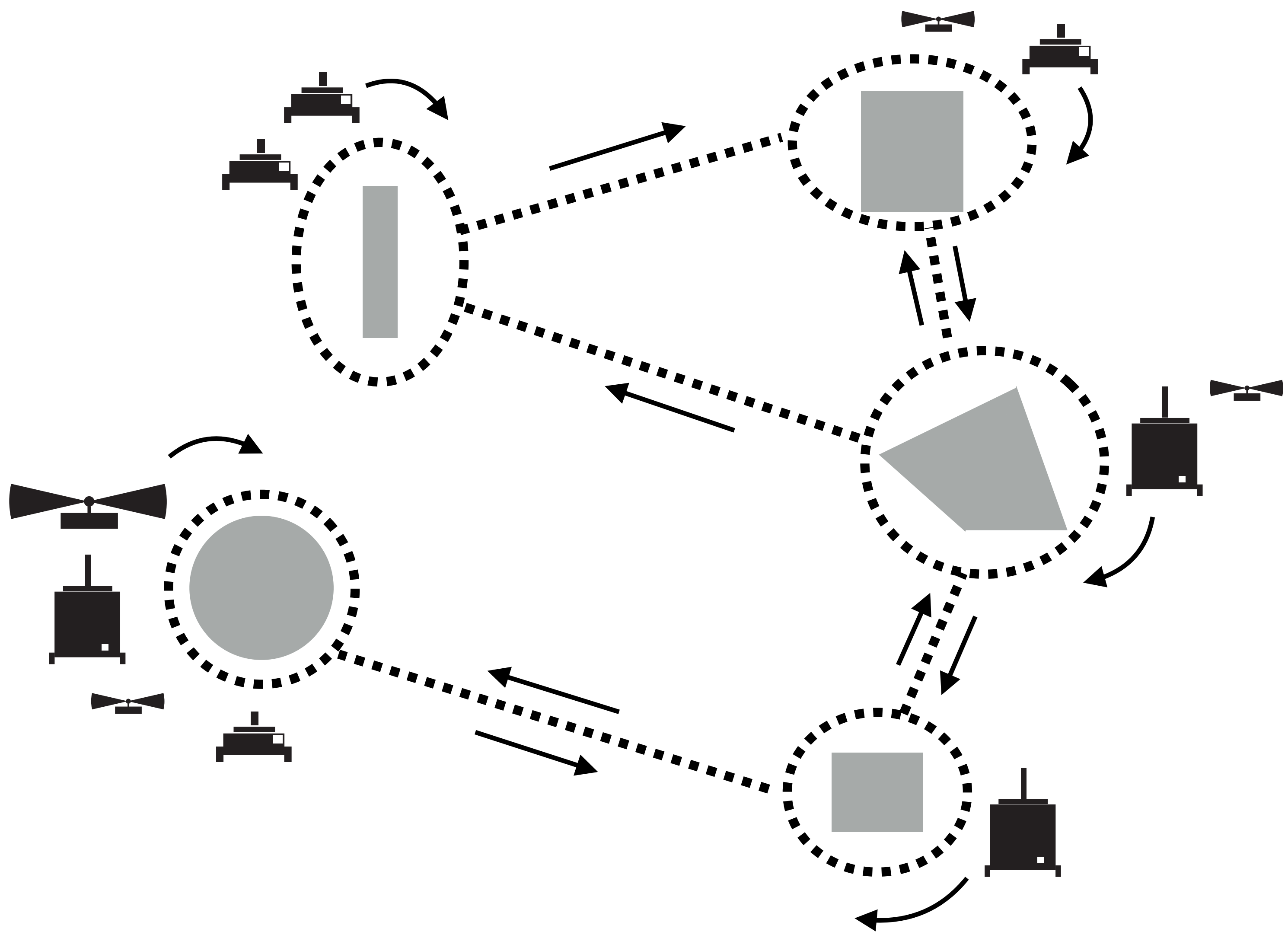
**Given** a desired robot distribution $\mathbf{x}^{(s)\star}$

**Find** transition rates $\mathbf{K}^{(s)\star}$ that are fastest to satisfy $\mathbf{x}^{(s)\star}$

Methods:

1. Explicit optimization; [Prorok 2016]
2. Approximation of **K**; semi-definite programming [Berman 2009]
3. Stochastic optimization [Matthey 2009, Hsieh 2008]

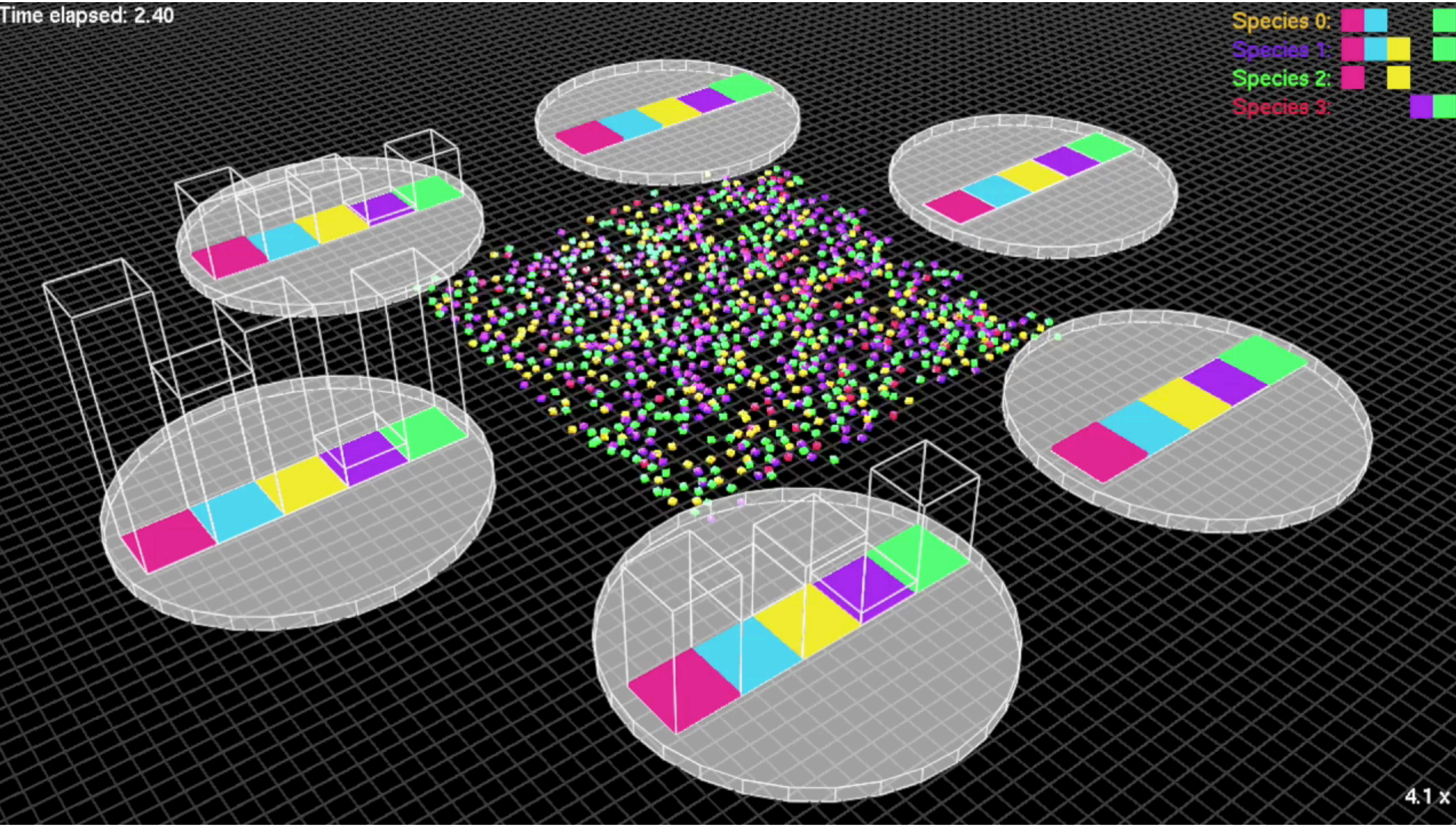
# Controller Synthesis



We extract rates for task-to-task transitions $k_{ij}^{(s)}$, and directly infer the switching probability.

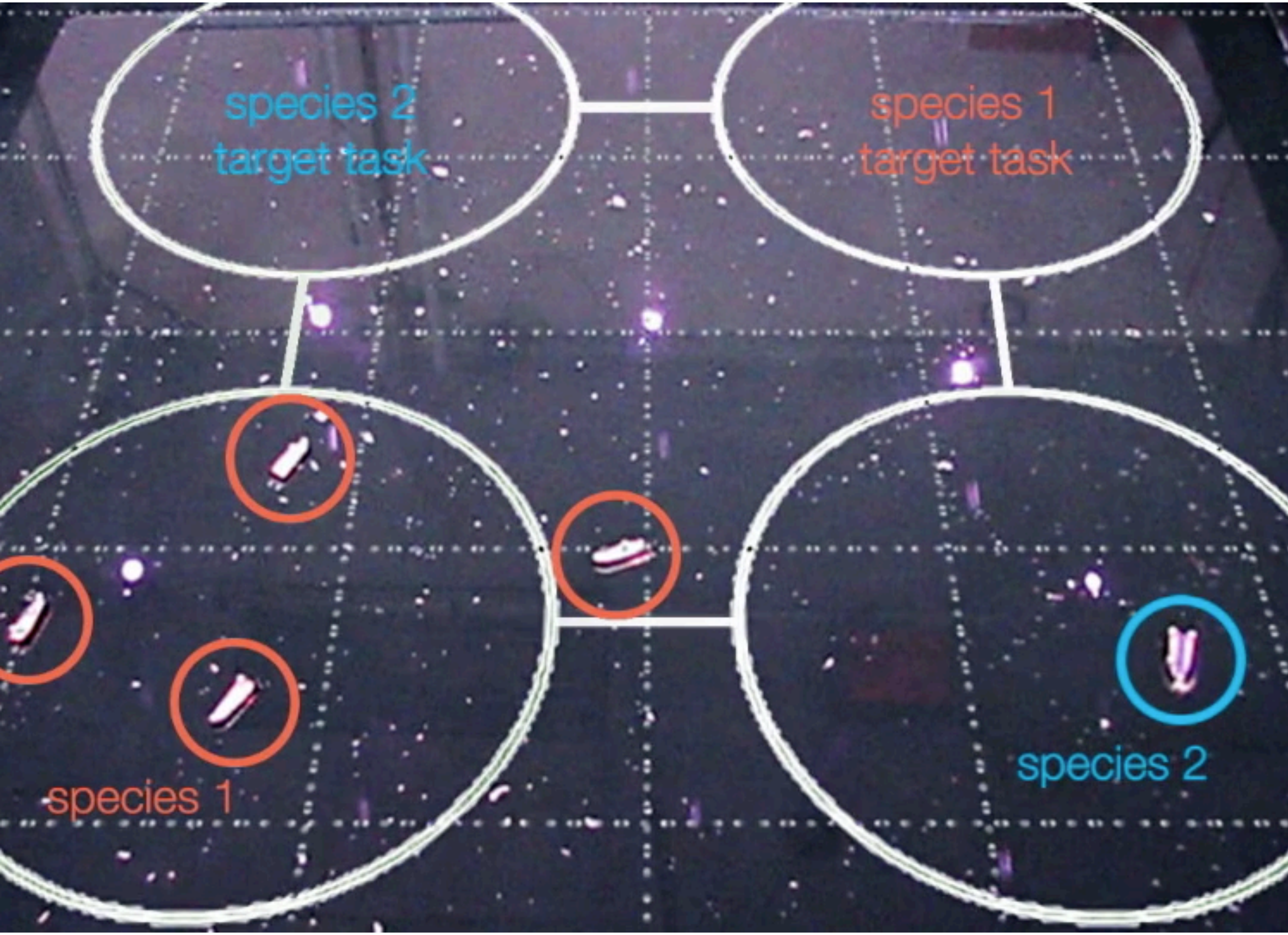
- Probabilistic controller is immediate
- Deterministic controller can also be derived
- Architecture: both open-loop and closed-loop possible

# Redistribution of a Heterogeneous Swarm




[Prorok et al.; 2016]

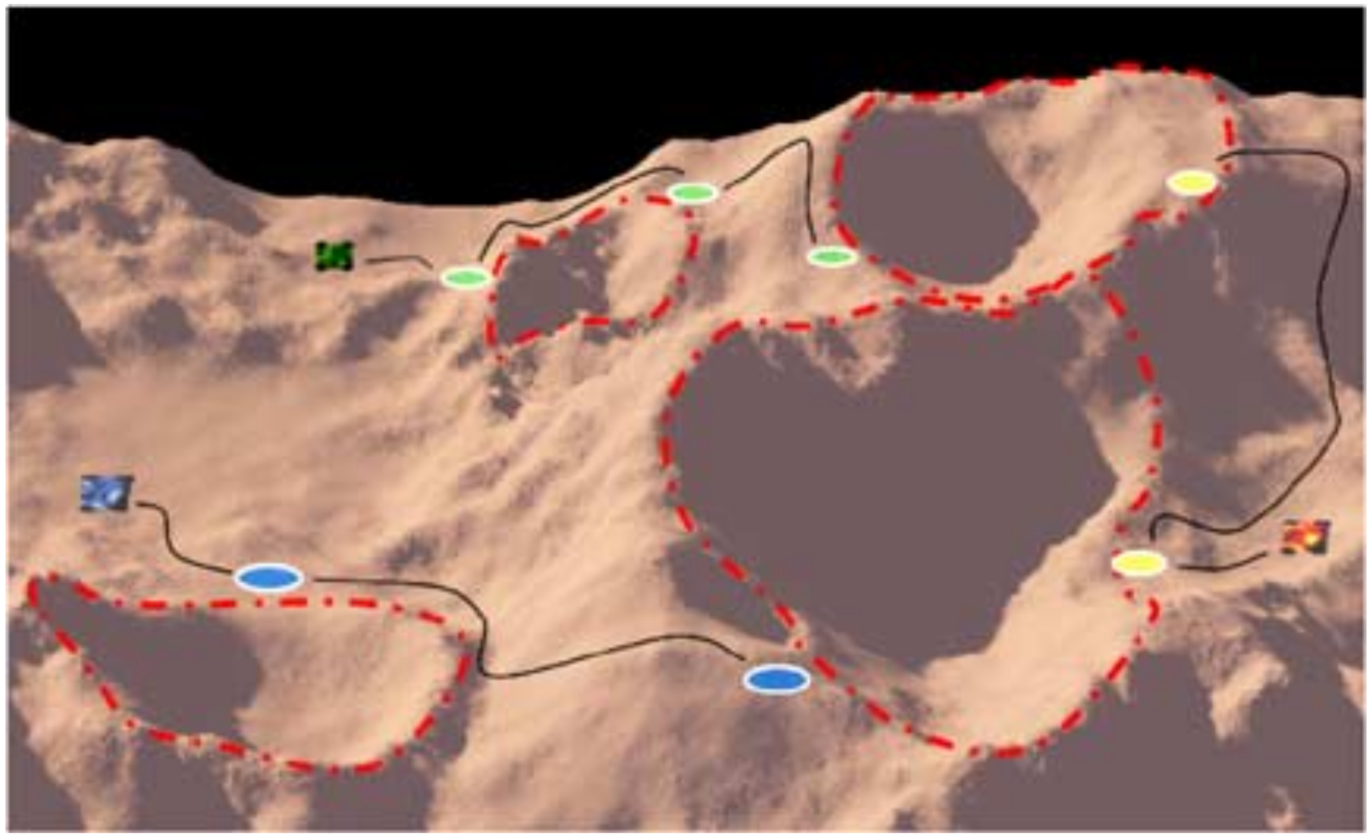
# Redistribution of a Heterogeneous Swarm




[Prorok et al.; 2016]

# Market-Based Coordination

- Robots: "self-interested agents that operate in a virtual economy"
- Tasks: "commodities of measurable worth that can be traded"



Example scenario: three robots exploring Mars. The robots need to gather data around the craters; they need to visit the 7 highlighted sites. Which robot visits each site?

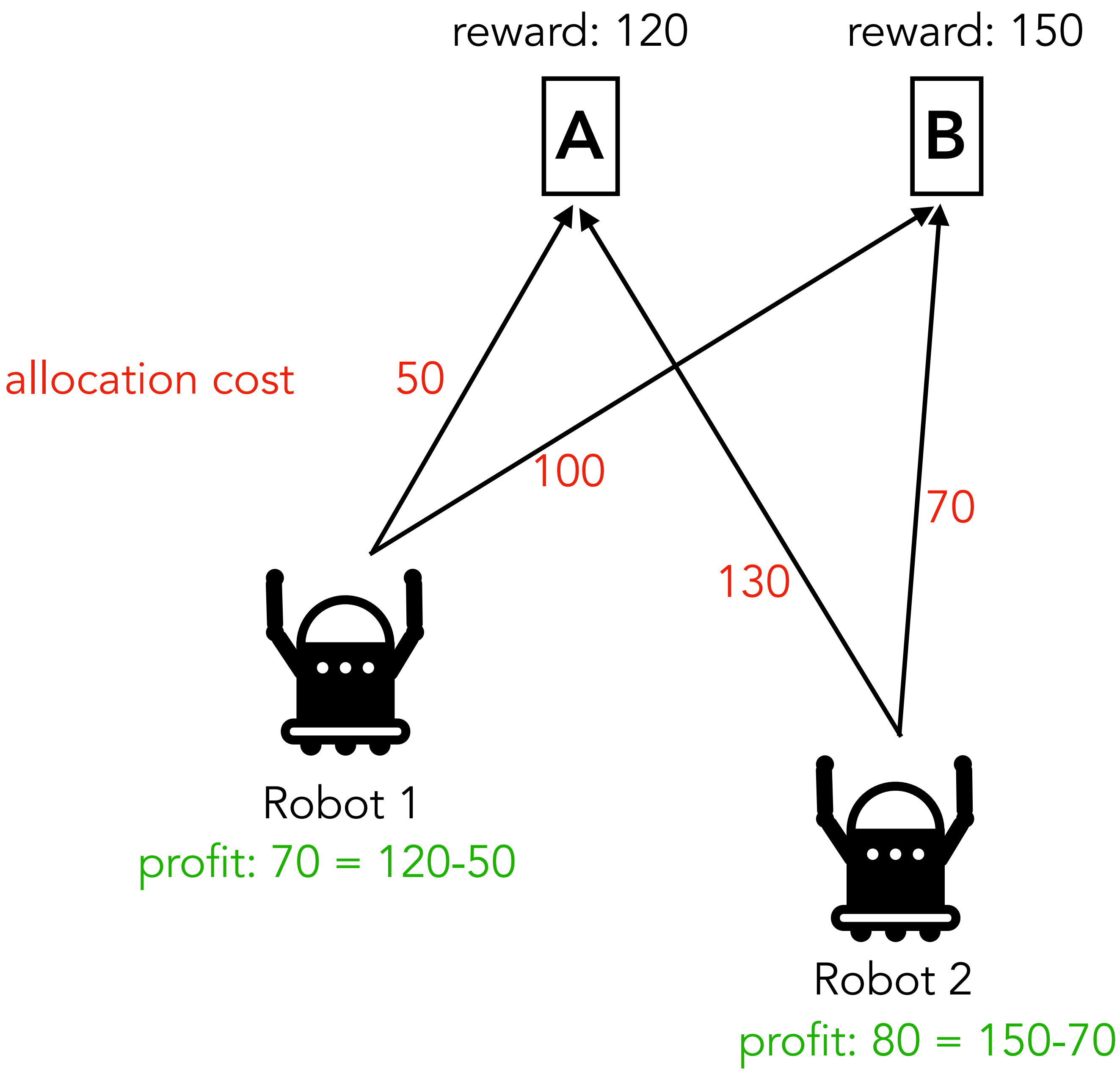
*image credit: Dias et al.

# Market-Based Coordination

- Underlying mechanism: **auctions**
- Auctioneer: offers items (tasks or resources) in announcement
- Participants (robots) submit bids to negotiate allocation of items
  - *sealed-bid* vs. *open-cry*
  - *first-price* vs. *Vickrey auction*
- **Single-item** auction:
  - highest bidder wins task
  - if no bid beats *reserve-price*, then auctioneer can retain item
- **Combinatorial** auction:
  - multiple items, robots bid on bundles
  - a bid expresses synergies between items
- **Multi-item** auction:
  - a robot can win at most one item apiece
  - special case of combinatorial auction for bundle of size 1

# Market-Based Coordination

A simple example (multi-item auction)




bids placed for tasks

| | A | B |
|---|---|---|
| Robot 1 | 50 | 100 |
| Robot 2 | - | 70 |

reserve price not met

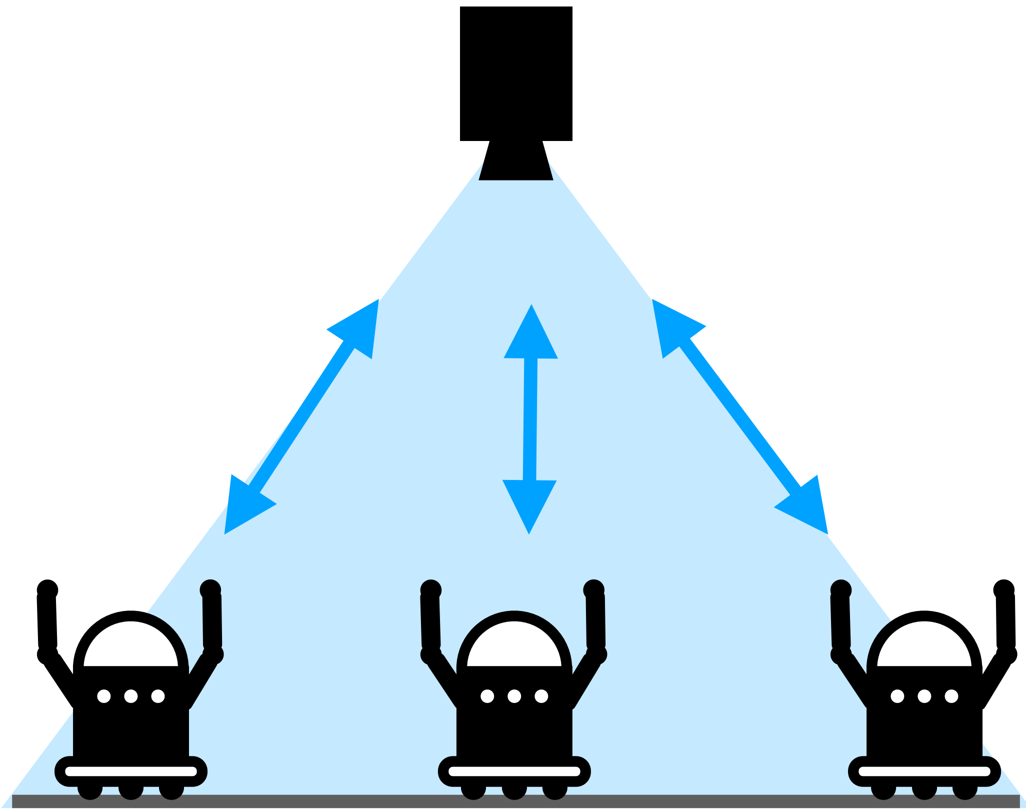
system cost: 50+70 = 120

Running time: $O(NRM)$ (greedy) or $O(N^2R)$ (optimal) [T. Sandholm; 2002]

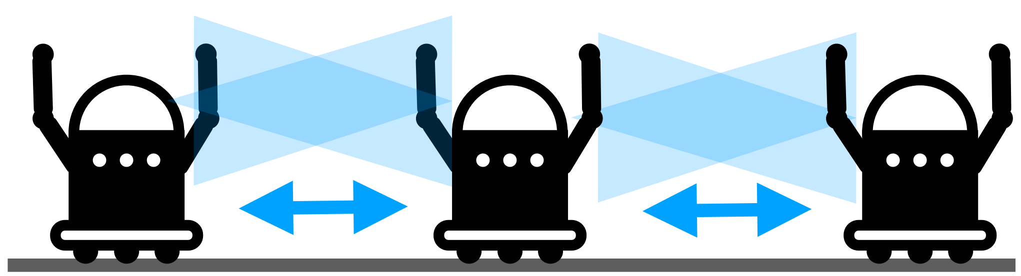
# Market-Based Allocation Frameworks

- Murdoch [Gerkey, Mataric; 2002]
  - loosely coordinated tasks
  - demonstrated on box pushing
  - demonstrated robustness, fast auctioning
- TraderBots [Dias et al.; 2004]
  - loosely coordinated tasks
  - demonstrated on exploration tasks
  - demonstrated robustness, scalability, auction types, task trees
- Hoplites [Kalra, Stentz; 2005]
  - tightly coordinated spatial tasks
  - robots auction plans not tasks
  - demonstrated on perimeter sweeping, constrained exploration

# Centralized vs Decentralized Assignment



centralized



decentralized

- Centralized assignment. Cost estimates are known at a central point (computational unit). The unit performs the assignment and communicates with all robots.

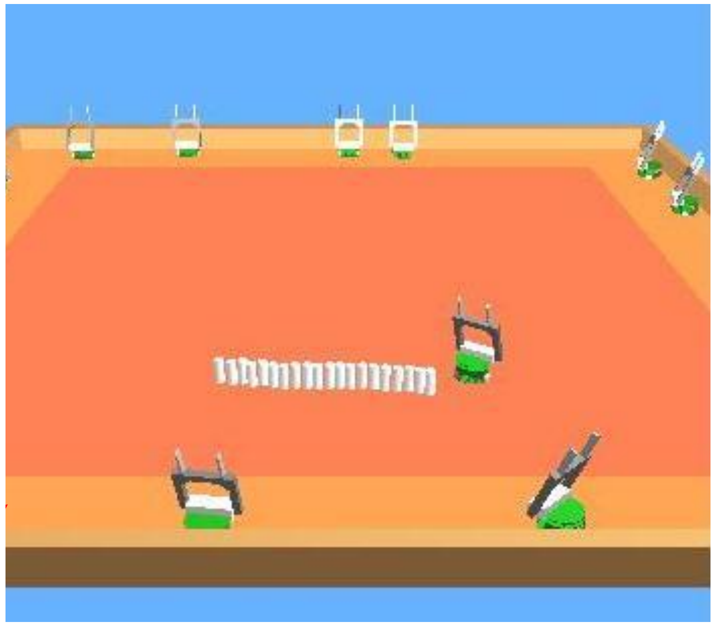
- Decentralized assignment. Robots do not have global knowledge of each other’s costs. They locally negotiate assignments.

Hybrid mechanisms: locally defined robot cliques can elect ‘leader’ robots and perform centralized mechanisms.

# Threshold-Based Assignment

- Fully **decentralized** mechanism.
- Each robot has an **activation threshold** for each task that needs to be performed.
- A **stimulus** reflects the urgency of a tasks; is continuously perceived by the robots;
- Example: threshold-based control of *aggregation* [Agassounon, Martinoli; 2002]
  - Goal: aggregate all sticks into 1 cluster
  - End criterion: robots should stop working once task is achieved

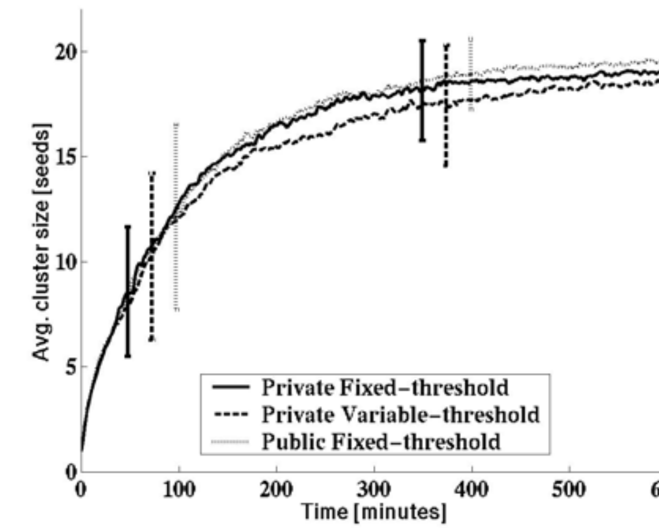
initial situation



final situation

# Threshold-Based Assignment

- **Stimulus**: time needed to find a stick to manipulate (the longer the time, the lower the stimulus associated with the task).
- **Threshold** is self-calibrated (fully decentralized).
- The number of manipulation sites (either end of line of sticks) decreases as global task nears completion.
- If time to find next stick goes beyond threshold $T$, then agent switches to resting behavior.




$$T = f \cdot \frac{1}{K} \sum_{k=1}^{K} t_k$$

threshold

number of sticks successfully collected so far

time taken to find k[th] stick

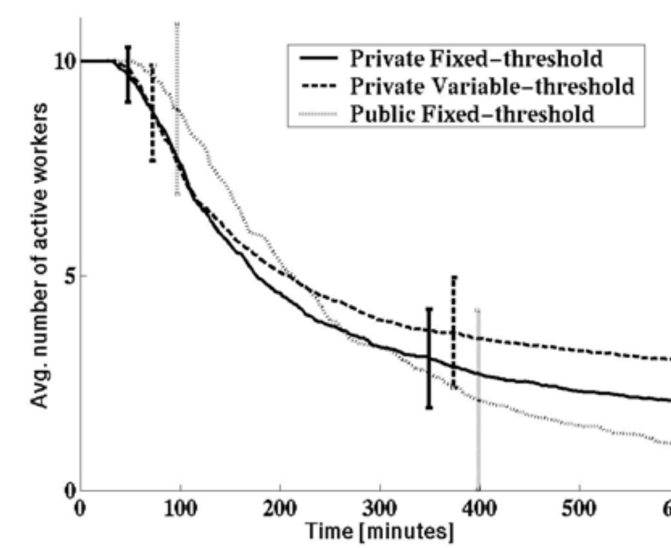



*image credit: Agassounon et al.

# Overview of Allocation Methods

| | centralized vs decentralized | optimality | completeness |
|---|---|---|---|
| Hungarian method | centralized | optimal | guaranteed |
| Mean-field approach | centralized or decentralized | approximative | The system converges. With high probability, completeness is guaranteed |
| Market-based approach | centralized or decentralized | greedy (sub-optimal) or optimal | depends on reserve price |
| Threshold-based approach | decentralized | suboptimal | not guaranteed |

# Further Reading

Nice overview of the classical problem:

http://www.assignmentproblems.com/

Seminal papers:

- B. Gerkey and M. Mataric, "A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems". Int. Journal of Robotics Research, 2004.
- M. B. Dias et al; "Market-Based Multirobot Coordination: A Survey and Analysis"; 2006
- D.P. Bertsekas, "The Auction Algorithm: A Distributed Relaxation Method for the Assignment Problem"; 1988.
- N. Kalra, A. Martinoli, "Comparative study of market-based and threshold-based task allocation"; 2006

Some new approaches for those interested:

- Redundant robot assignment under uncertainty: A. Prorok, Redundant Robot Assignment on Graphs with Uncertain Edge Costs, 14th International Symposium on Distributed Autonomous Robotic Systems (DARS), 2018
- Assignment in heterogeneous robot swarms: A. Prorok, M. A. Hsieh, and V. Kumar. The Impact of Diversity on Optimal Control Policies for Heterogeneous Robot Swarms. IEEE Transactions on Robotics (T-RO); 2017.
- Assignment under privacy constraints: A. Prorok, V. Kumar, Privacy-Preserving Vehicle Assignment for Mobility-on-Demand Systems, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017