Distributed Task Selection in Multi-agent Swarms Using Heuristic Strategies

David Miller[1], Prithviraj Dasgupta[2], Timothy Judkins[3]
[1]Mechanical Engg. Dept, Univ. of Nebraska-Lincoln
[2]Computer Science Department, [3]HPER Biomechanics Lab
University of Nebraska-Omaha

Outline

- Multi-agent swarming scenario
- Task selection problem
  - Theoretical hardness results
- Heuristic-based strategies for task selection
- Experimental results

What is swarming?

- Movement of entities individually of in small-sized units to search and act upon objects of interest in a search space
- Objects of interest are distributed randomly in the search space
- When one unit discovers an object of interest it informs other units
- Other units then converge on the object to act upon it using their combined power

Why do we use swarming?

- Distributed
  - only behavior of individual units are programmed
  - manifests global behavior of system
- Not very difficult to program individual units
- Complex systems can be designed from simple behavior patterns of individual units

How do we implement swarming?

- Program the desired behavior into each swarm unit
- Each swarm unit is a robot
- Each robot’s controller is implemented as a software agent (small footprint, easy to embed)
- We will focus on the algorithms used by this software agent

Distributed Swarming Scenario

- Boundaries of environment are known *a priori* by agents (generalization of detecting walls in a closed room)
- What is a task?
  - set of actions needed to be taken by agent on an object of interest
  - spatial and temporal distribution of tasks are unknown

Distributed Swarming Scenario

- How are tasks executed?
- What can a single agent do?
  - Discover a task
  - Only partially execute a task
- How are tasks completed?
  - Group of agent needed to complete a task
  - Each agent partially executes task

Distributed Swarming Scenario

- How is this group of agents formed?
  - – Agent that discovers object associates a certain amount of synthetic pheromone with the object
  - – communicates pheromone to other agents (robots within comm. range)
  - – some agents receiving communication decide to visit the object to act upon it and complete it

Distributed Swarming Scenario: Task Selection

- How does an agent decide which task to act upon?
  - Task selection mechanism

Previous Work

- Every agent deposits pheromone at a central location or map
- Other agents
  - read this map to get the global picture
  - solve an optimization problem to distribute tasks among themselves
- Not distributed!
- Gaudiano05, Sauter05

Distributed Task Selection

- Each agent has only its “view” of
  - tasks it discovered first-hand and deposited pheromone
  - tasks it became aware of through communication received from other agents
- Each task is represented by a pheromone point
- Agent’s view called its “pheromone landscape”

Research Question

- Given multiple tasks on its pheromone landscape, how does an agent select a task to visit next?

Other Aspects of Swarming Scenario

- How are agents deployed?
  - By a manager from a base station
- How do they avoid collisions (path de-conflict)?
  - Potential field based technique
  - Robots behave like charged particles
  - If a robot comes within a certain threshold distance of other another robot they repel each other

Other Aspects of Swarming Scenario

- How do robots search and discover tasks?
  - Uninformed search
  - Robots have appropriate sensors to recognize a task when it encounters one
- How do robots communicate?
  - Flooding-like algorithm (probabilistic flooding) – used in p2p overlay networks
- How do agents execute tasks?
  - Application specific
  - After completing its portion of all tasks on its task list, a robot reverts to searching

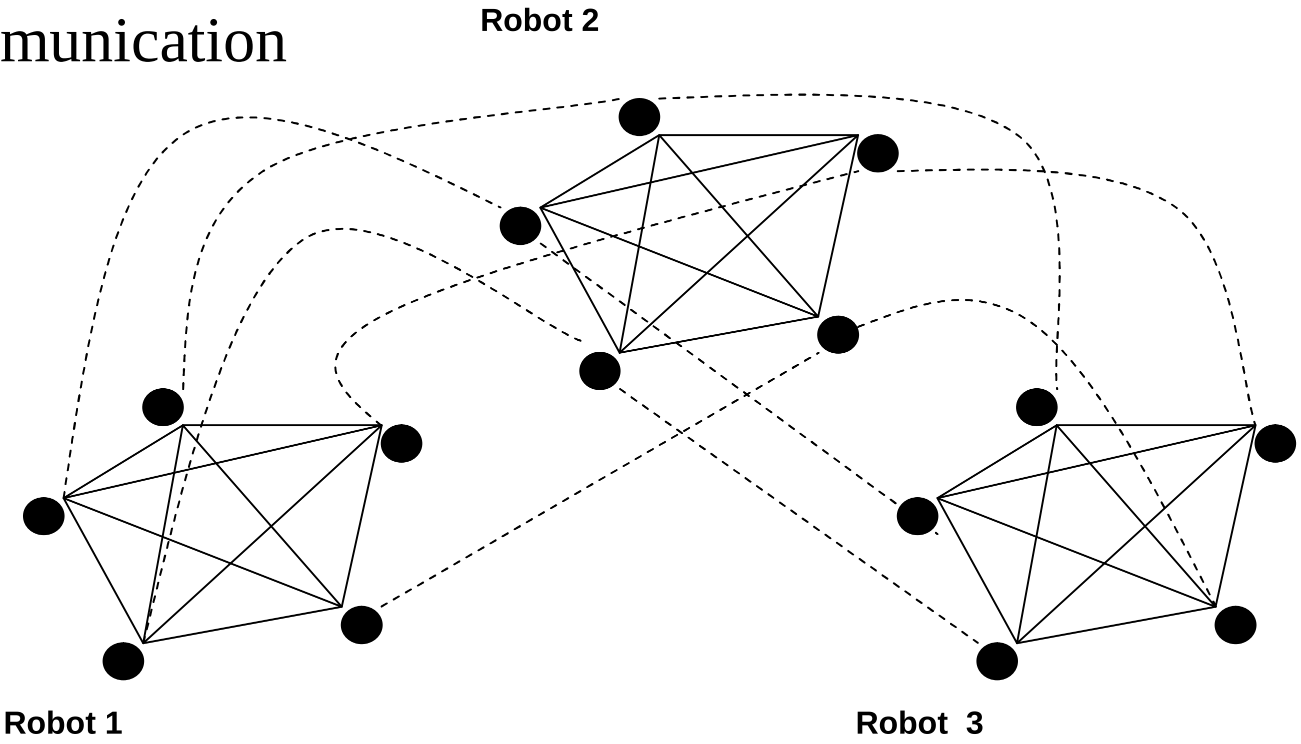
Distributed Task Selection Problem

- Simple scenario
  - 3 robots
  - 5 tasks
  - Assume each task has same amount of pheromone initially
  - Assume each robot is within communication range of the other two robots

Robots’ View of Tasks

\- - - - - - Communication

● Task



Distributed Task Selection Problem

- $w_{t,r}$: time required by robot r to reach task t
- $x_{t,r}$: time required by robot r to execute its portion of the task
- Dynamic optimization problem facing each robot
  - $min \sum_{t \,\varepsilon\, Tr} x_{t,r} + w_{t,r}$

Distributed Task Selection Problem

- Can be modeled as a dynamic traveling salesman problem (DTSP)
    - TSP where the edge weights(distance or time) change dynamically (TSP with traffic jams)
- DTSP is NP-complete (Proof in paper)
- Polynomial time approximation algorithm does not exist (since edges do not follow triangle inequality)

Heuristic-based Solutions

- Four heuristics
  - Each can be calculated in polynomial time

Distance-based Heuristic

- Each robot selects a task that is
  - "closest to me and has highest amount of pheromone"
  - heuristic value: maximize product of distance and amount of pheromone

Robot-density Based Heuristic

- Each robot selects a task that has
  - least number of robots in its vicinity, lowest pheromone (starved tasks)
- But...how can a robot know which task has least robots near it?
  - Locations of robots keep changing and robots do not continuously exchange locations with each other

Robot-density Based Heuristic

- Probabilistically estimate confidence in location of robot, based on time elapsed since last communication about location (Naive)
- Heuristic value: minimize sum of product of location-confidence, amount of pheromone, distance of other robots from task
  - sum is taken over all robots I am aware of

Robot-preference based Heuristic

- Similar to last heuristic (starved tasks first)
- Also considers amount of task outstanding
  - starved tasks nearing completion first
- heuristic value: same as last heuristics but also consider amount of execution left

Robot-proximity based Heuristic

- Similar to last heuristic (starved nearing completion tasks first)
- Also considers effect of other robots
  - How many other robots are likely to be headed (ahead of me) to the task?

Experimental Setup

- Automatic target recognition(ATR) application
- Focus is on swarming, not on distributed image recognition or tracking algorithms
- Robots are heterogeneous (in sensors)
- Each task requires 4 robots for being id-ed
  - for e.g.: in ATR each robot executes an image identification algorithm that identifies target with 25-30% certainty, 4 robots need to run image id algorithm separately to id a target

Experimental Setup

- Webots simulator
- 18 robots, 20 targets
- Environment: 50 X 50 sq. units

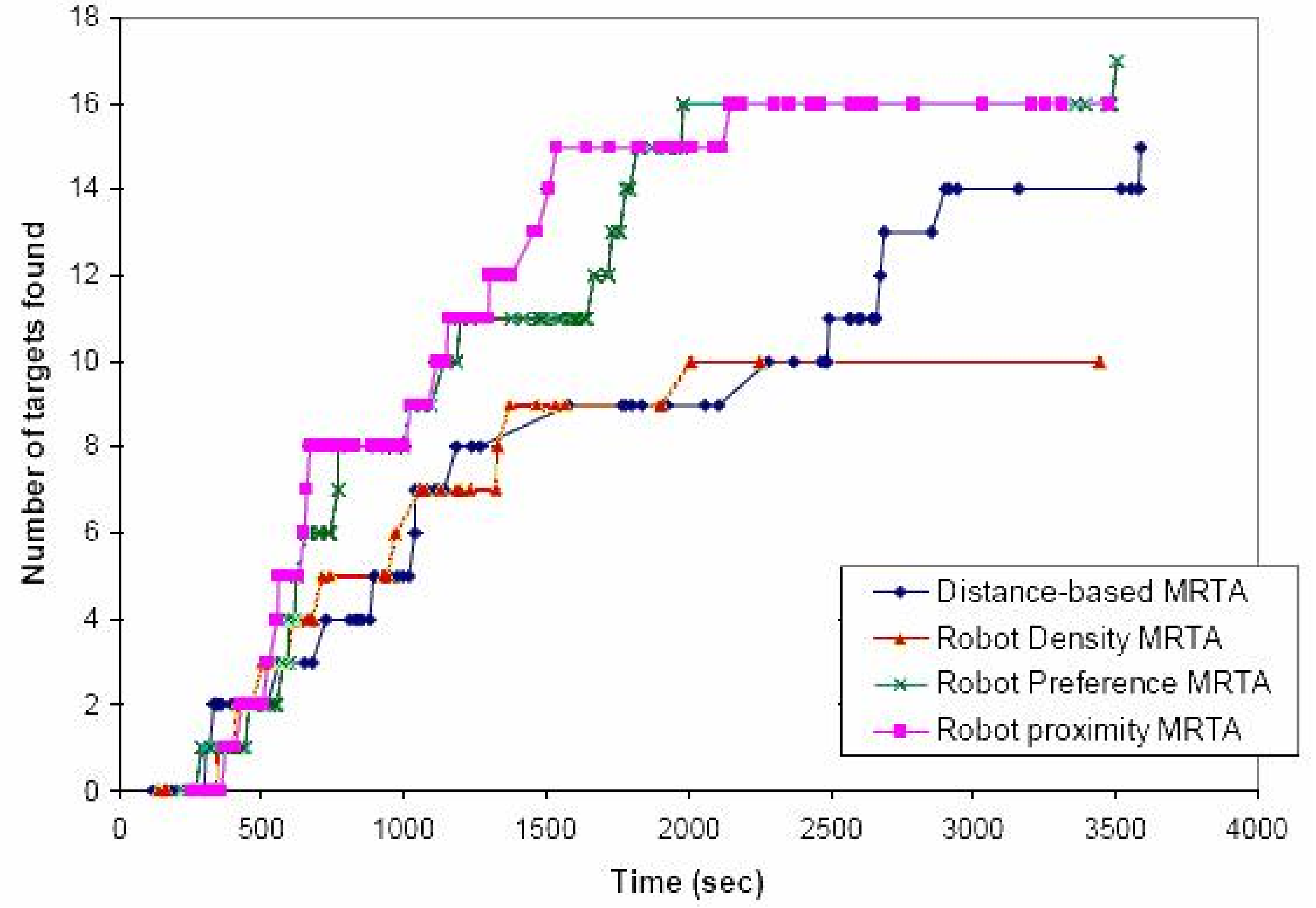
Experimental Setup: Robot

- DifferentialWheels Model, maxspeed = 40
- GPS (x, z, heading)
- Downward looking IR sensor, range 0-2048
- Short range radio transmitter/receiver, for "ping"s (collision avoidance), range 1.5 units
- Long range radio transmitter/receiver, for communication, range 7.5 units

Experimental Setup: Targets

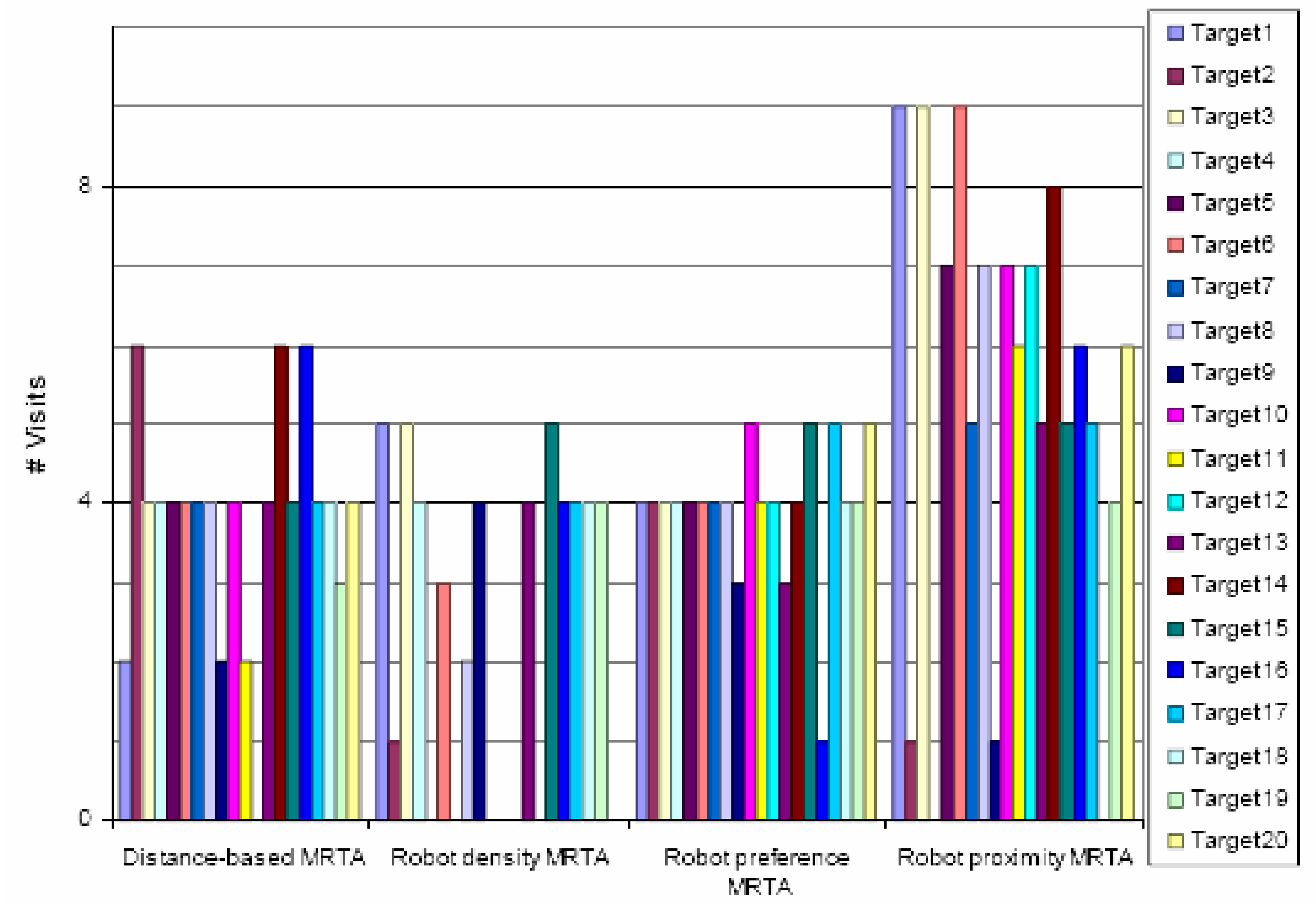
- 20 targets
- 4 target types
  - colors = {red, green, blue, purple}
- 5 targets of each type
- Floor of environment: black (zero reading on IR sensor)
- Targets detection: IR sensor of robot returns non-zero reading when robot passes over target

Experimental Results



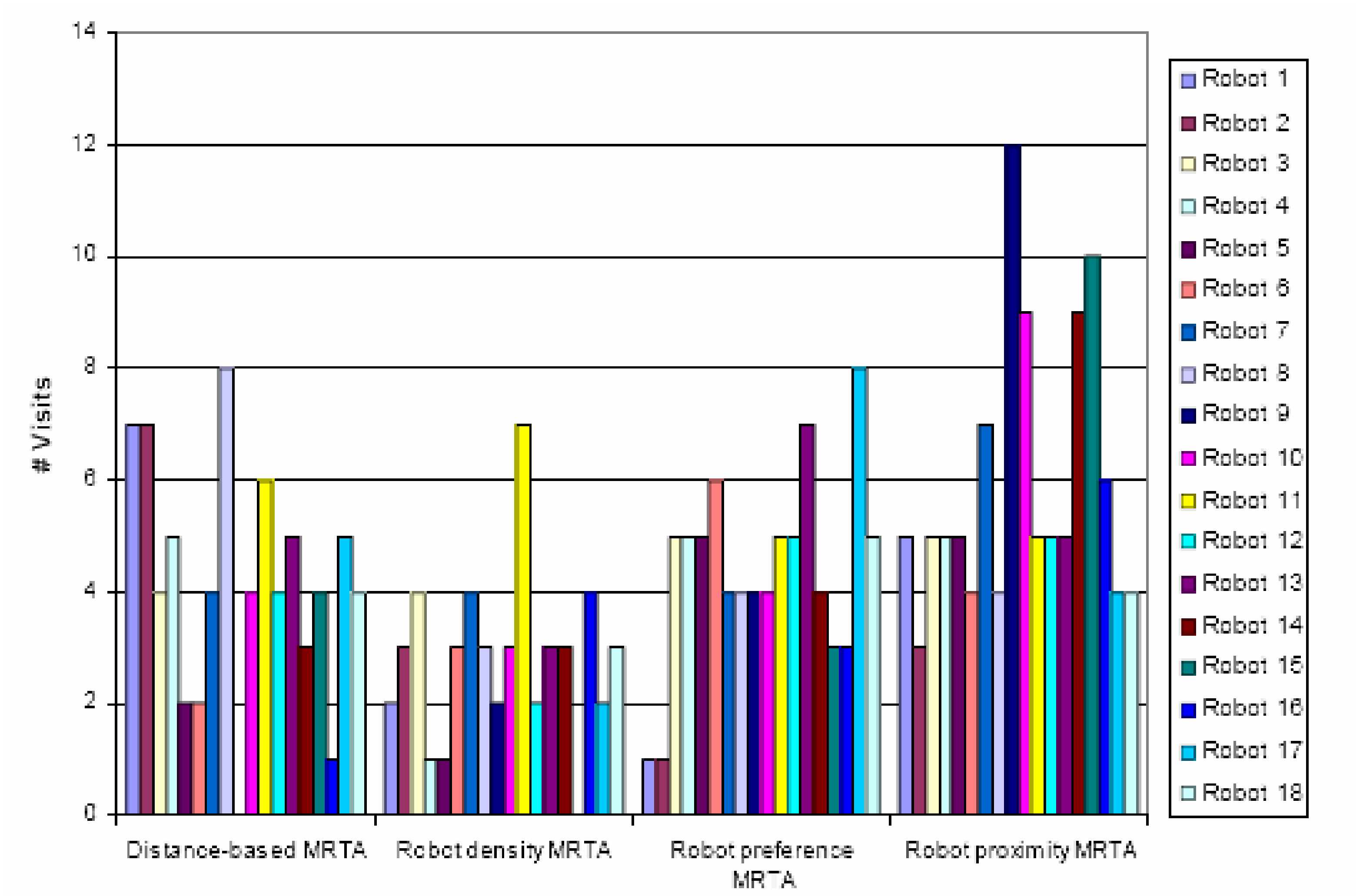
Time taken to identify targets with different strategies

Experimental Results



No. of times each target is visited by a robot

Experimental Results



No. of targets visited by each robot using different strategies

Conclusions

- Preference and Proximity-based heuristics are better
- Preference-based: better task and robot distribution, slightly less computation
- Proximity-based: better completion time

Future Work

- Look at the type of interactions between robots/tasks
  - overall objective: minimize task completion time
  - targets compete with each other to get id-ed faster
- Model interactions between robots and targets as a game
- Use a market-based algorithm for task selection (paper at Intelligent Agent Technology Conf., Dec. 2006).
- Only slightly better than heuristics...What next?

Future Work

- Communication overheads are significant when no. of robots or targets increases
  - not scalable
- Intelligent communication mechanisms