Wireless Mesh With Mobility

Michael Lin, Tim Bolbrock, Hosam Rowaihy,Thomas La Porta

Active and passive RFID tags allow dynamic tracking of inventory data in fixed and temporary warehouses. However, neither currently present an ideal solution to the problem of real-time inventory tracking. Passive RFID tags can only be read from a short distance, which makes remote warehouse-wide data querying impossible. Active RFID tags can overcome this limitation by creating a wireless mesh network, but such networks greatly impact useful battery life, especially in high-load situations. By using mobile readers, we can overcome the limitations of both passive and active RFID tags. Mobile readers can be remotely called upon to directly read passive RFID tags, allowing remote queries. Furthermore, mobile readers can directly access large data sets in active tag mesh networks, improving network lifetime.

In previous work, we designed and implemented algorithms for a single robot in an active tag mesh network using TinyOS on Crossbow Motes and a custom robot design. Our current project is the design and implementation of algorithms for remote data access using multiple robots with differing modes of access in a passive RFID tag warehouse. In addition, Dr. Guohong Cao and his students are designing algorithms for centralized and distributed caching of data from passive RFID tags to further improve query resolution time.

Mobile Reader Behavior

★ If queries are uniformly distributed, deciding which robot will respond to a query and how the robots should move is simple

★ When queries are skewed to a certain area of the warehouse, we intuitively want some robots to stay close to that area and not respond to queries that would bring them away from the hot spot

★ We introduce *areas of responsibility* and *rest points* based on this intuition

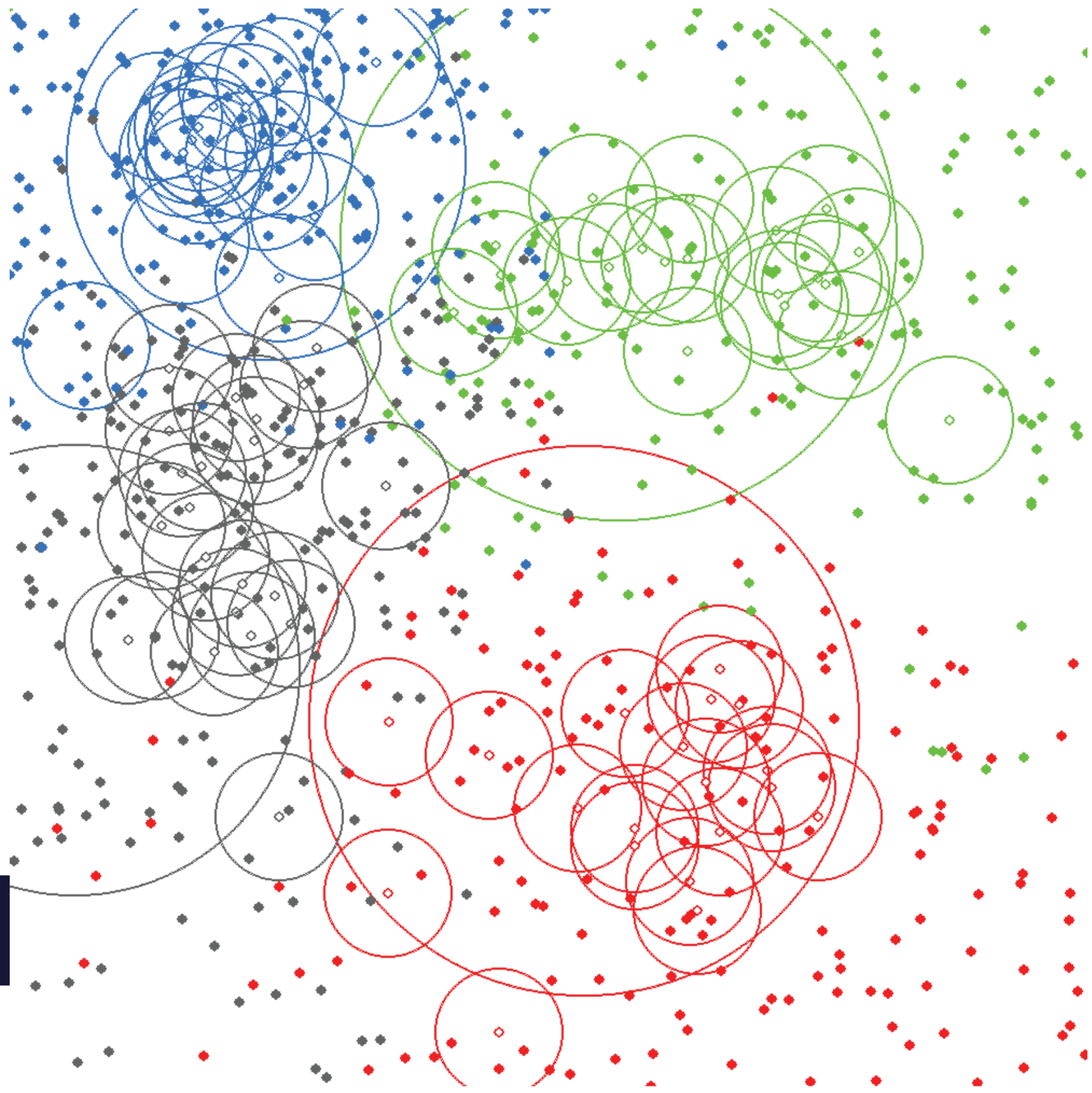
★ Robots have dynamically resizing *areas of responsibility*

★ If two areas overlap, the nearest robot will respond to the query, similarly, if a query is outside any *area of responsibility*, the nearest robot responds

★ *Areas of responsibility* resize based on a weighted moving average of the last queries the robot responded to, ideally shrinking to accommodate hot spots and skew

★ Between queries, robots return to within an *inner radius* of *rest points*, which are based on the average *x, y* coordinates of the last *n* queries the robot responded to, with outliers removed

★ Ideally, *rest points* keep a robot in a hot spot and bring other robots closer



Simulations

★ We simulated our algorithms over 1000 queries in a custom Python simulator with a visualization as shown on the right

★ Filled dots represent data locations, with the color corresponding to the robot that responded to the query

★ Large circles represent the *area of responsibility* at the end of the simulation, centered at each robot's current location

★ Small circles represent the *inner radius*, centered at each *rest point* the robot calculated

★ Hollow dots represent the *rest points* themselves

★ In order to demonstrate our algorithms, the 1000 queries were skewed such that 1/3 of the data locations were in the upper left quadrant

★ The locations of the *rest points* and the final sizes of the *areas of responsibility* show how the algorithm responds to skew

★ The blue robot's *rest points* and final *area of responsibility* show that it remained close to the upper left corner due to the skewed queries

★ In turn, the grey and green robots migrated closer to the upper left corner to maintain coverage

Ongoing Work

★ Here we assume robots can communicate through a base station; we are currently designing algorithms for robots to communicate through a mesh network

★ We are implementing our algorithm in Linux with real RFID tags and readers on a robot platform from prior work

★ Distributed and centralized caching algorithms and means for updating the cache using both handheld RFID readers and our mobile readers

★ Updating through handheld readers will be automatic as warehouse workers walk throughout the warehouse carrying the reader

★ Full integration of caching algorithms and our mobile reader implementation