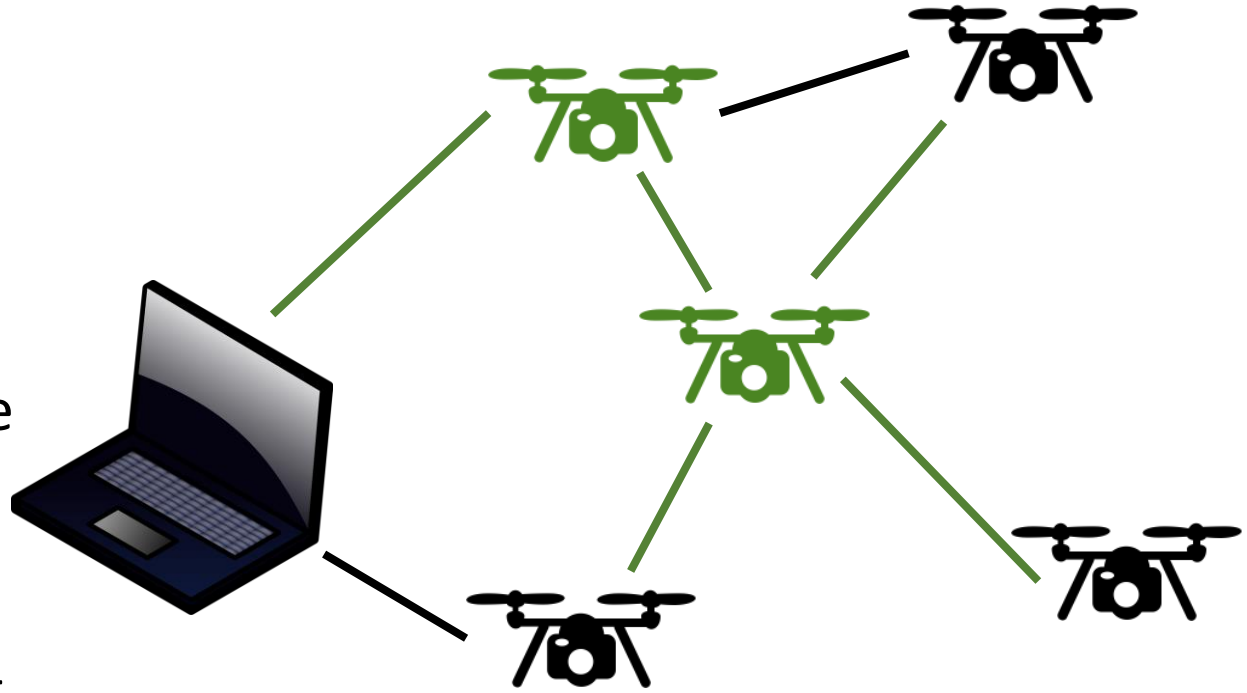
Streaming Changes of Centrality of UAV Networks

Joshua Huseman

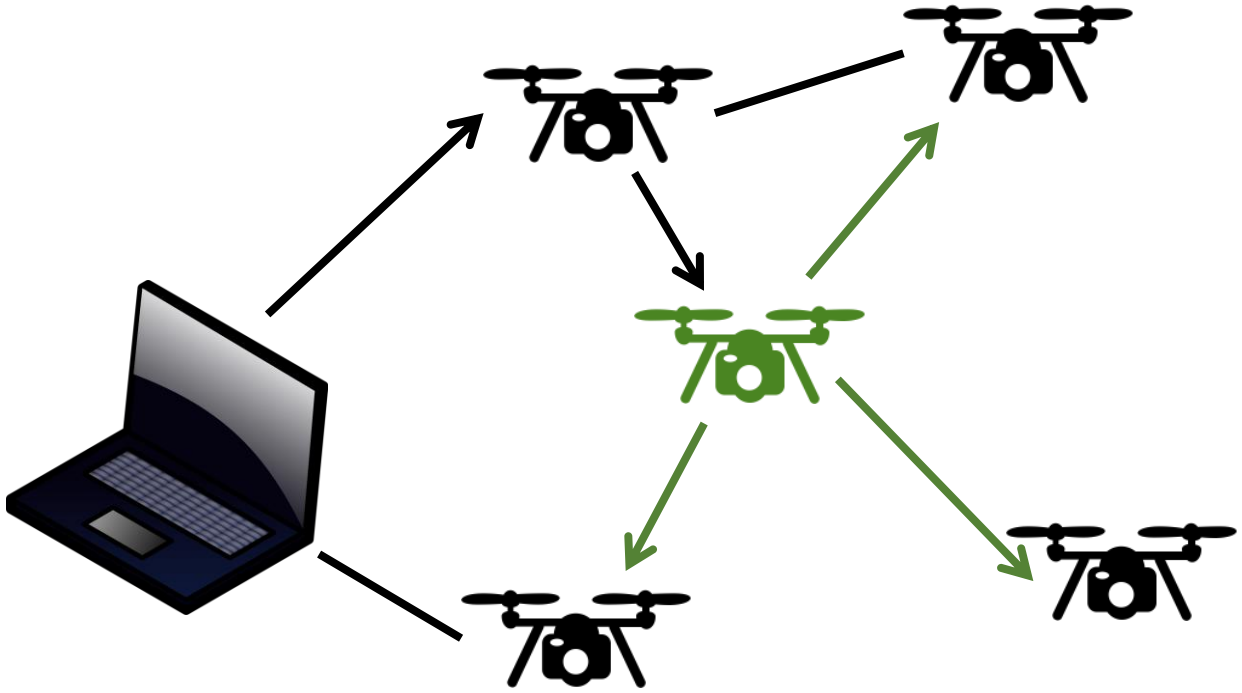
Application – Drone Networking

- Mesh Network of UAVs
  - Wi-Fi links between UAVs
    - Signal strength decreases as UAVs get further apart
  - A few UAVs linked to the Ground Control Station (GCS)
  - Each needs to communicate with the GCS
    - Can either communicate directly or through another UAV
  - Overhead cost when communicating through other UAVs



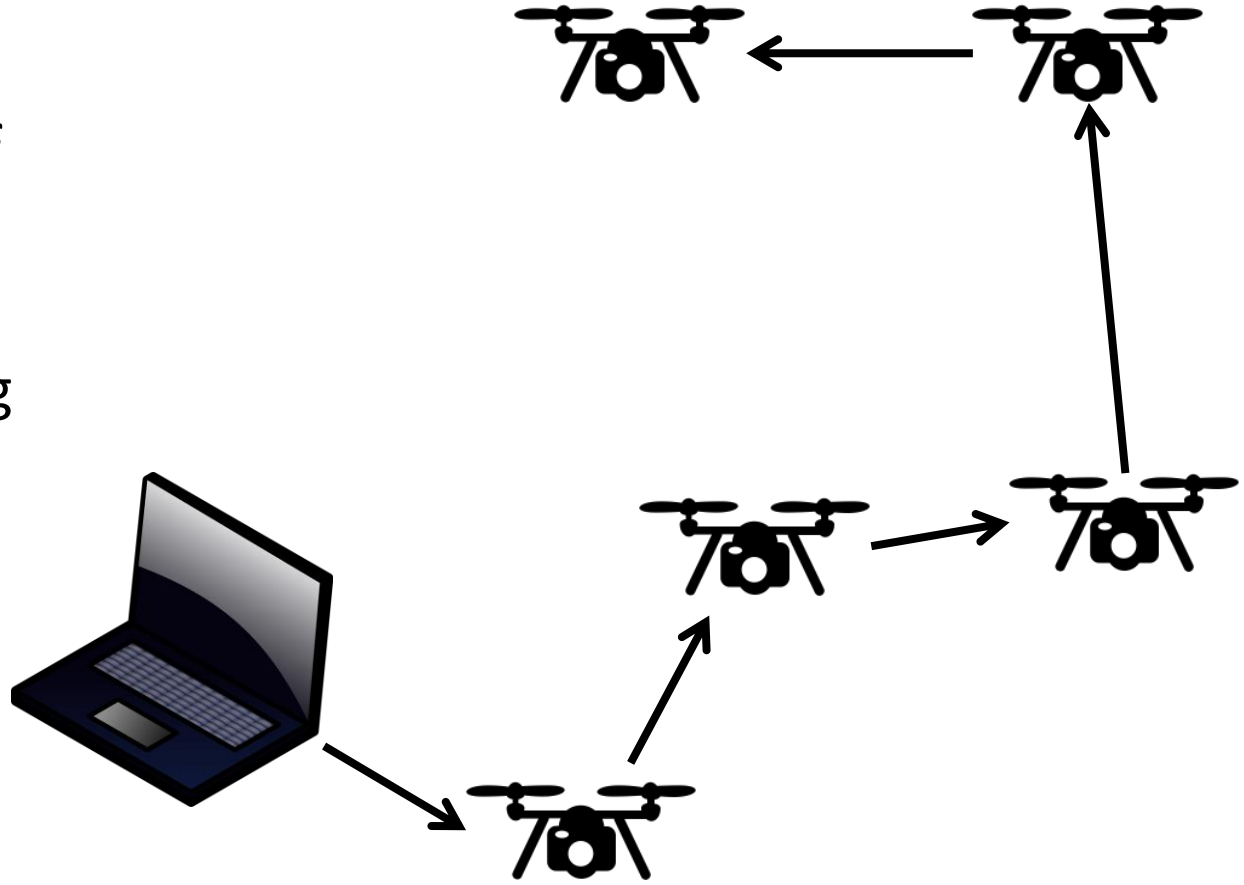
Kernel – Best Network Layout

- Central UAVs can act as “hubs” for other UAVs, providing communication with the GCS
  - Overhead cost to the hub in passing other UAVs’ communication
- Determine “best” UAVs to use as hubs
  - Closeness centrality seems like a good option



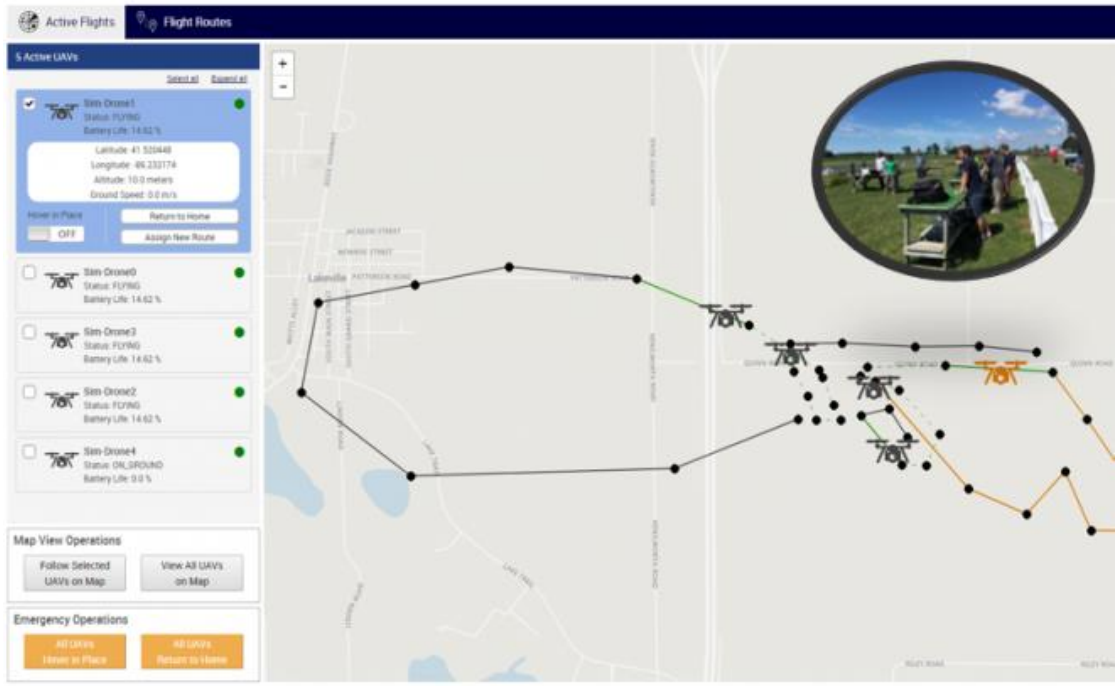
Kernel – Incremental / Distributed Changes

- UAVs move, changing the shape of the network
  - Evolving the network will improve performance compared to rebuilding after each change
- Distributed algorithm
  - A decentralized, distributed algorithm allows UAVs to determine the network's structure independently



Data Sets

- generated randomly
  - random walk/diffusion pattern of UAV movements
  - communication signal strength based on distance (with maximum bound)
- I may have access to some data logs from previous flights/simulations for more realistic data



Screenshot of Dronology project UI during a previous test

Pseudocode – Closeness

- For each node x
  - Initialize farness variable to 0
  - For each node y
    - Calculate shortest path between x and y
      - (Maybe use Dijkstra's algorithm)
        - Simple implementation worst case $O(n^2)$
        - Better implementation O(e + n log(n))
    - Add shortest path length to farness variable
  - Closeness(x) = (# nodes) / farness

- Time complexity:
  - $O(e\ n^2 + n^3 \log(n))$
  - (e = # edges)
  - (n = # nodes)

Pseudocode – Building Network (Tree Traversal?)

- Order nodes by closeness (descending)
- While there are remaining nodes:
  - Set first node as a hub
  - Find shortest path to any existing hub or GCS
    - Set all of these nodes as hubs
- When setting node as hub:
  - Remove node and all directly connected nodes from list of remaining nodes
  - Add all directly connected nodes to list of “slaves” for the new hub

Still Unknown/Working Out Details

- Faster algorithms
  - Time complexity quite slow – great optimization needed
  - Slight imperfections in the results may be acceptable for time efficiency
- Distributed algorithm implementation
  - Not sure yet how to do this with incomplete knowledge of graph
- Is Closeness the best measure to use?
  - Another measure, like # adjacent edges, etc. might be better (and faster)
- Benchmarking
  - Method for analyzing/comparing the efficiency of the created network after generation

Resources

- https://en.wikipedia.org/wiki/Closeness_centrality
  - Equation used for implementation
- https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm
  - Time complexity of Dijkstra's Algorithm
- https://dronology.info/
  - Information on UAV project
  - Used for picture on Data Sets slide