Clusterhead & BLE Mesh discovery process

jason.peng

November 2025

1 BLE Discovery for Increasing Mesh Capacity

Traditional BLE mesh networks can only feasibly support up to 100-150 node devices before communication starts failing due to crowding, collisions, and latency being unfeasibly high. The following protocol is a BLE Mesh Discovery network protocol designed for expanding bluetooth mesh to the thousands or even tens of thousands in capacity.

- **Uses**:
  - Ad hoc sensor networks: For example, in a large facility (warehouse, campus, smart-building, industrial site) you might deploy tens of thousands of sensor nodes (temperature, humidity, motion, occupancy). Using BLE discovery you can allow nodes to identify nearby neighbors and then form a mesh for relaying data.
  - Large-scale lighting or automation systems: Large-scale lighting or automation systems For example, commercial lighting in an office building or campus: many nodes need to discover each other (or find the network) and then mesh to reach controls, monitoring, and control panels.
  - Smart city / infrastructure deployments For example, parking sensors, street-lamp sensors, environmental monitors across a large campus or city area. Nodes need to discover their local network, and mesh allows them to extend connectivity to gateways or to each other without needing each node to talk directly to a gateway
- **Restrictions**: Latency must be relatively low - in the seconds count for endpoint to endpoint. Capacity and scalability must be high, capable of supporting up to thousands of nodes.

2 BLE Discovery

Bluetooth Discovery occurs on a single channel. All devices/nodes will propagate a discovery message with the following format:

- **ID**: The unique identifier of the message sender
- **TTL**: Time To Live - the number of hops remaining before the message is discontinued
- **PSF**: Path So Far - the sequence of node IDs the message has traveled through
- **LHGPS**: Last Heard GPS location of the message sender. Each device caches its most recent GPS location and includes it in all discovery messages. If a device has never obtained a GPS lock (e.g., indoors, GPS-denied environments), the LHGPS field is marked as unavailable, and GPS-based proximity filtering is skipped for that device's messages.

During Bluetooth Discovery, each discovery cycle contains four message slots. As a result, a given node will use one slot to send its own discovery message (if applicable) and has up to 3 remaining slots available to forward other nodes' messages. We use three separate metrics in order to determine which messages to forward if there are more messages queued than available slots.

1. Picky Forwarding: Uses the Crowding factor to forward only a given percentage of messages, preventing network congestion.
2. GPS Proximity: If the LHGPS of the previous sender is too close to the current location of the device, it will not forward.
3. Time to Live: The remaining messages are ordered using TTL and the top 3 messages are chosen.

These metrics are chosen in order to throttle the amount of messages being propagated within the network. They are chosen to work ideally with a single Bluetooth cluster having a maximum capacity of 150 devices.

3 Clusterhead Discovery and Identification

BLE Clusterhead election occurs during normal Bluetooth Discovery and is based on connectivity metrics to ensure well-connected nodes become Clusterheads. The election process has 3 rounds to ensure redundancy strength and works as follows:

1. **Connectivity Metric Calculation**: During the discovery phase, each node tracks:
   - Number of direct neighbors (nodes within 1 hop)
   - Number of unique paths discovered through forwarded messages
   - Geographic distribution of neighbors based on LHGPS data

2. **Process Description**: The beginning of discovery operates in the following order:

   - Crowding Factor: During a specificied period, each device will "noisy broadcast" - send a broadcast base level noise message. Each individual device will pick a stochastically randomized time slot to listen. When listening, each device will determine crowding factor as a function $f(RSSI)$.
   - : Clusterhead Broadcast: Nodes will then stochastically transmit a "broadcast" in a small timeslot. Each node will listen the majority of the time, documenting how many nodes heard. This counts as "direct connection" count.
   - Clusterhead Candidacy: Each bluetooth node will elect itself a clusterhead or edge, determined by a function of the ratio between direct connection count and noise heard. The motivation for using ratio lies on two principles:
     - 1) RSSI will be greater than the contribution from direct connections. RSSI will account for noise from near devices that don't have a strong enough signal to parse, but still contribute noise
     - 2) RSSI will factor in "collision signals" as well. When signals collide while broadcasting connection count, this lowers connection count while still amplifying RSSI. As a result, since further connections will use the same hash function parameters, nodes with a lower likelihood of nearby neighbors colliding will have higher direct connection count : noise heard ratios, making them more suitable for clusterheads. As a result, the more reachable direct connections in a noisy area a node has, the more likely it is to become clusterhead.
   - 3) Clusterhead Comparison & Election Announcement: Clusterheads will broadcast their direct device count to remove nearby lower count clusterheads.
   - Cluster Formation: Edge devices will orient themselves with a specific clusterhead and store the path and knowledge of the clusterhead.

3. **Clusterhead Candidacy**: A node becomes a Clusterhead candidate if:

   - It has more than a threshold number of direct connections (dynamically determined based on local crowding factor)
   - Its neighbors are geographically well-distributed (not all clustered in one area)
   - It has successfully received and forwarded messages during discovery

4. **Election Announcement**: Nodes that meet the candidacy criteria announce their Clusterhead status in subsequent discovery cycles by adding a Clusterhead flag to their discovery messages. Election announcement

will happen 3 times in 3 separate rounds to ensure high stochastic probability of reaching all devices.

Devices will broadcast their direct neighbor count to resolve conflicts between clusters. The election announcement message will be structured in the following manner:

- **Class ID**: The class ID of a clusterhead class to self-identify as a (potential) clusterhead
- **ID**: The unique identifier of the message sender
- **PDSF**: Predicted devices reached so far. This function is computed by doing the summation of devices reached at each step. General structure:
  $$t(x) = \sum_i \Pi_i(\mathrm{x}_i)$$
  where x represents the direct connection count at each step i, excluding the device(s) message has reached previously.
- **PSF**: Path So Far - the sequence of node IDs the message has traveled through
- **Score**: Clusterhead's score as determined by candidacy.
- **Hash h(ID)**: Unique FDMA / TDMA Hash Function for edges to reach that specific clusterhead.

Election announcements will "flood" areas by propogation outwards from clusterhead through retransmission. When PDSF reaches cluster capacity, devices will stop retransmitting the election announcement, and no more devices will add themselves to the clusterhead.

5. **Conflict Resolution**: If multiple Clusterhead candidates have overlapping coverage:
   - Nodes compare the number of direct connections
   - The candidate with more direct connections becomes the Clusterhead
   - In case of ties, the node with the lower device ID takes precedence
6. **Election Re-announcement** Nodes that initially qualified for candidacy but heard a clusterhead candidate that had a higher direct connection count will re-announce upon the follwing cycles that they have renounced their candidacy.
7. **Cluster Formation**: Once Clusterheads are established, Edge nodes align themselves with the Clusterhead from which they received the lower length path or the highest "direct count" message. Devices will "memorize" all available paths to the clusterheads, and construct a tree to the clusterhead using available paths. In the case of multiple interconnected

paths to one cluster, Djikstra's will be used to compute the shortest path - this will be added to feasible paths as well the "memorized path," all of which will be used as feasible paths to the target clusterhead.

While not implemented in the scope of this project, this is the theoretical basis for setting up full discovery:

> Upon completion, edge devices will use FDMA/TDMA slotting based on stochastic initital hash to reach the clusterhead. Regarding paths to other clusterheads: additionally, at this stage, devices that have been found by multiple "clusterheads" will use their determined paths to relay information about all the clusterheads they are connected to - this information will be sent to each individual clus- terhead. Clusterheads will use a few shortest-hop paths to determine the viable paths to any other clusterhead.

8. **Result**: Devices will know they themselves are clusterheads, and all edge devices will be able to identify clusterheads and the path to each respective clusterhead. As clusterhead - clusterhead communication is established, these devices can reach any other device on edge.