LoRa Quake – Building A Self-Configuring LoRa Mesh Network for Finding People under Rubble

07 August 2023

Alphan Şahin

Acknowledgement

- This research is supported by “TUBITAK 2221 - Call for Earthquake Research Opens within the Scope of Supporting Scientist with Guest or Academic Leave Program” between June and July 2023
- The research is conducted at CoSiNC 6G research lab directed by Prof. Hüseyin Arslan, Istanbul Medipol University
  - Many thanks to the students and staff at CoSiNC family!

Agenda

- Part 1: Motivation & Challenges
- Part 2: Proposed solutions
- Part 3: Demonstration

Motivation & Challenges (1/2)

- February 6th Kahramanmaraş Earthquakes (2023 Turkey–Syria Earthquake) showed that it is very difficult to establish timely contact and locate those trapped under the rubble quickly

- Challenge: Even if a trapped person has his/her cell phone nearby, establishing a connection through cellular networks is quite challenging:
  - Significantly attenuated radio signals due to the debris (30-60 dB [1])
  - Damaged, inaccessible, and/or congested cellular networks due to the earthquake

[1] Holloway, C. , Koepke, G. , Camell, D. , Remley, C. , Schima, S. , McKinley, M. and Johnk, R. (2006), Propagation and Detection of Radio Signals Before, During, and After the Implosion of a Large Convention Center, Technical Note (NIST TN), National Institute of Standards and Technology, Gaithersburg, MD

Motivation & Challenges (2/2)

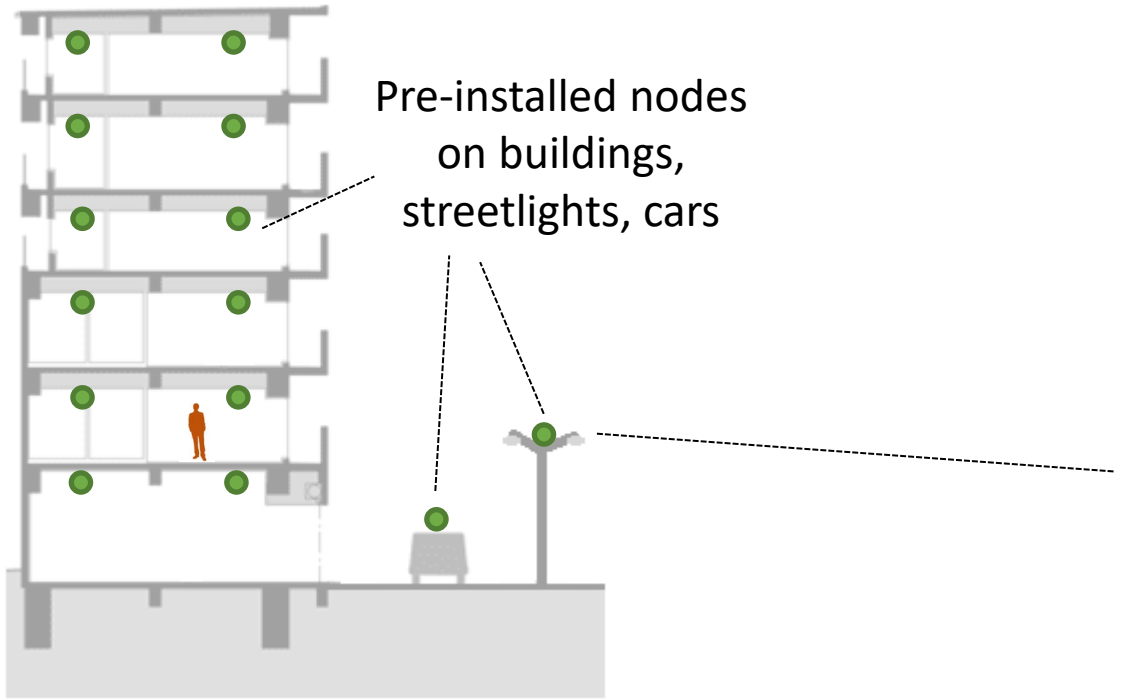
- Establishing contact largely depends on rescue teams and their equipment
  - Methods using heat, sound, or CO2 levels [1]
  - Systems based on electromagnetic or acoustic waves using radar principles
    - Ground penetration radar (e.g., NASA FINDER working at 3 GHz band) [2], [3]

- Challenge: Detection via systems deployed from the surface into the buildings' interiors is limited due to the weakened signals
  - We need innovative communication and sensing technologies to significantly reduce our losses in future disaster situations

[1] Zhang, Di, Salvatore Sessa, Ritaro Kasai, Sarah Cosentino, Cimarelli Giacomo, Yasuaki Mochida, Hiroya Yamada, Michele Guarnieri, and Atsuo Takanishi. 2018. "Evaluation of a Sensor System for Detecting Humans Trapped under Rubble: A Pilot Study" Sensors 18, no. 3: 852.
[2] NASA, https://spinoff.nasa.gov/FINDER-Finds-Its-Way-into-Rescuers-Toolkits, Last accessed: 01/31/2023
[3] V. Cable, J. Lux and S. Haque, "Target & propagation models for the FINDER radar," 2013 IEEE Antennas and Propagation Society International Symposium (APSURSI), Orlando, FL, USA, 2013, pp. 1614-1615

Proposed Approach

- Deploy low-cost Long-Range (LoRa) nodes with sensing capabilities to the buildings and establish a mesh network along with the reference nodes brought by the rescue teams if the building collapses
  - Nodes sense their environments and convey sensing results to the reference nodes

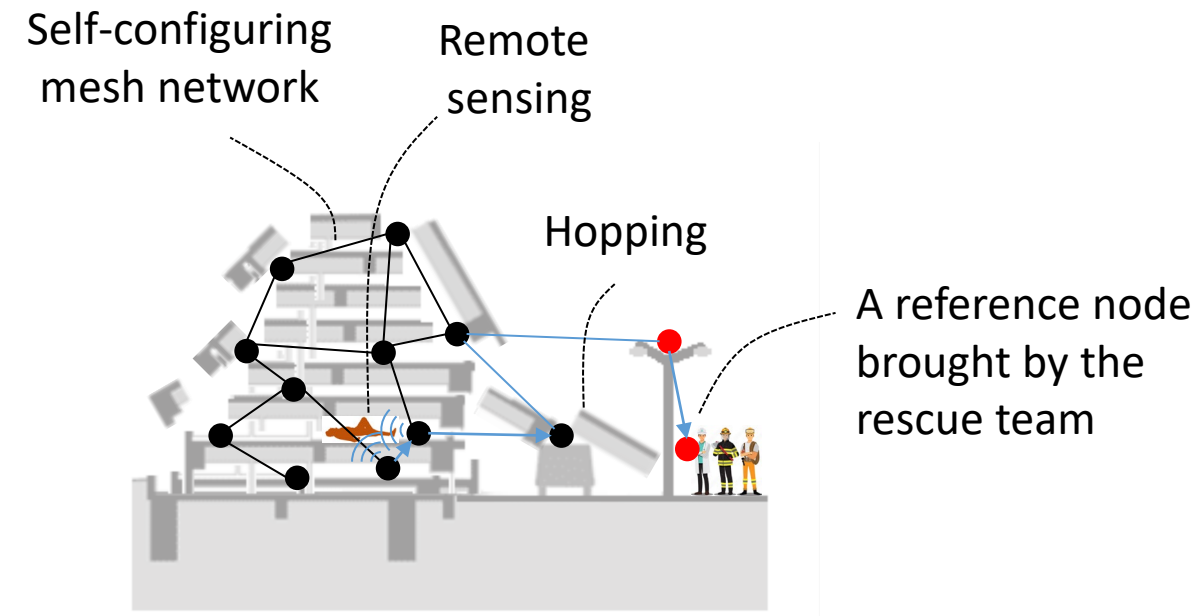


Before the earthquake

A typical LoRa node ($10-$50,
866 MHz, 915 MHz, 2.4 GHz)

Proposed Approach

- Deploy low-cost Long-Range (LoRa) nodes with sensing capabilities to the buildings and establish a mesh network along with the reference nodes brought by the rescue teams if the building collapses
  - Nodes sense their environments and convey sensing results to the reference nodes



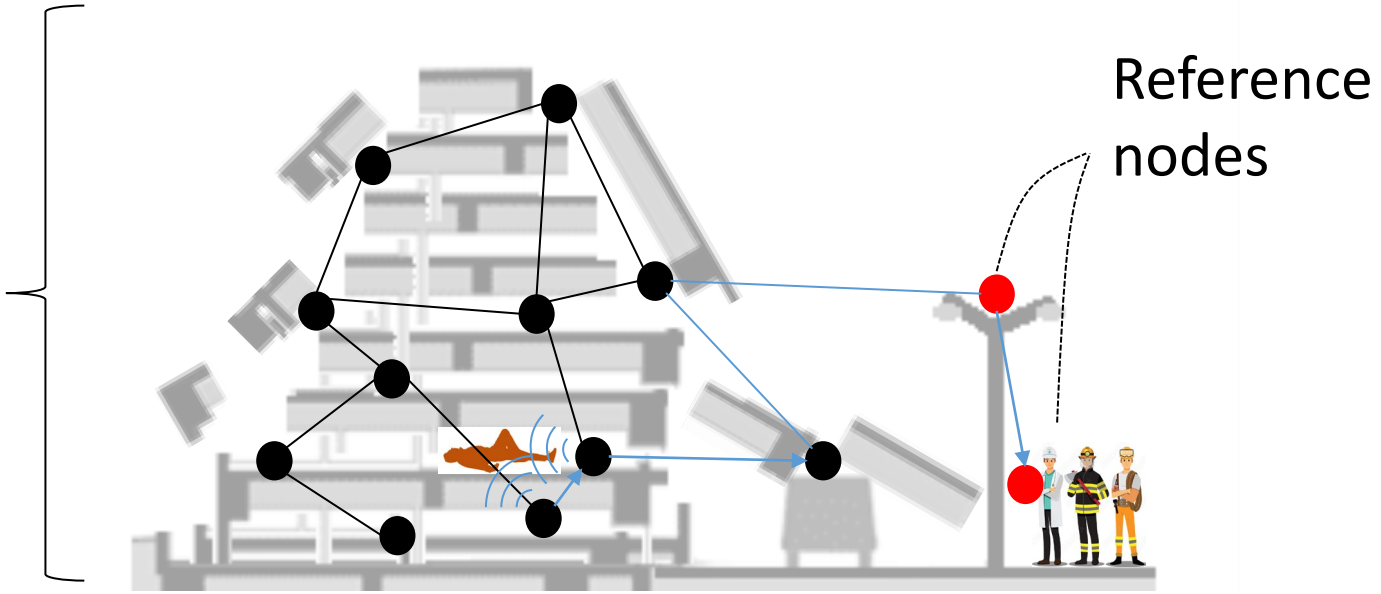
After the earthquake (if the building collapses)

A typical LoRa node ($10-$50, 866 MHz, 915 MHz, 2.4 GHz)

Goals

- Goal 1: Establishing a self-configuring mesh network
  - How to deal with interference problems without a coordinator?
- Goal 2: Transferring messages from nodes to the rescue teams
  - How to decide where to send the message in the mesh?
- Goal 3: Finding the positions of the nodes to find the trapped people
  - How to locate the nodes?

1) Interference?
2) Reaching reference nodes?
3) Localization?

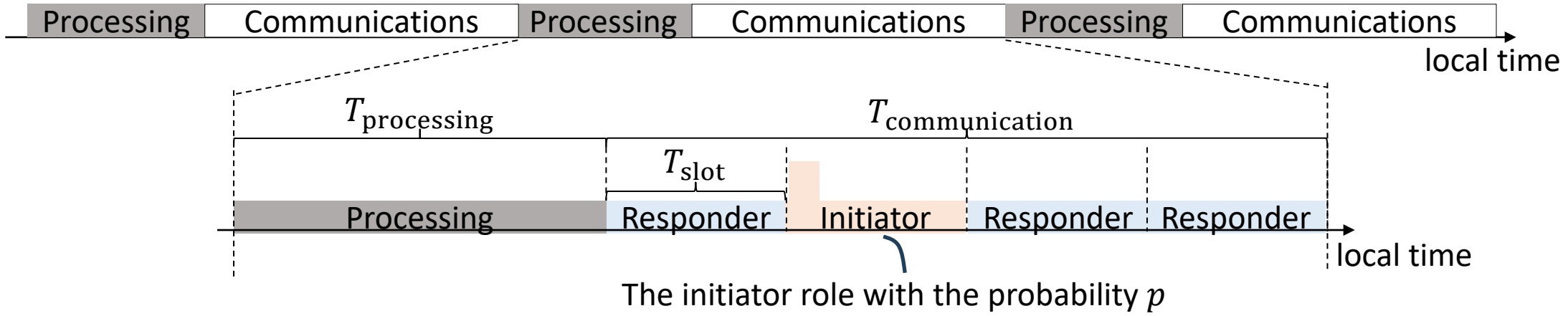


Solutions

Goal 1: Establishing a self-configuring mesh network

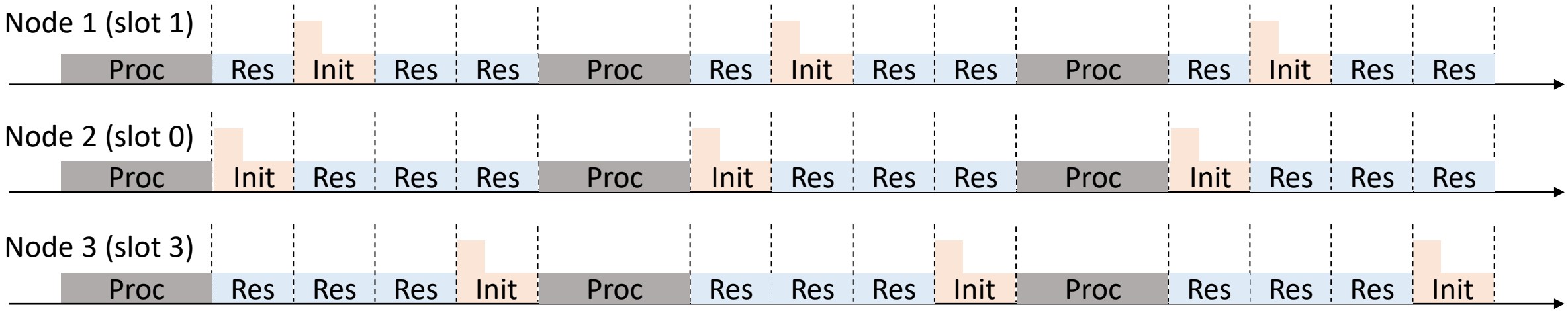
Periodically Alternating States

- A typical LoRa device equipped with a low-cost microprocessor
  - Cannot handle communication (TX and RX) and processing tasks simultaneously
- We propose to alternate processing and communication periodically, further divide communications into N_{slot} slots, and define three states:
  - 1) Processing, 2) Initiator, and 3) Responder
  - A node is granted to be an initiator only with one of the slots with the probability of p



How to Synchronize the Nodes and Choose Slots (1/2)

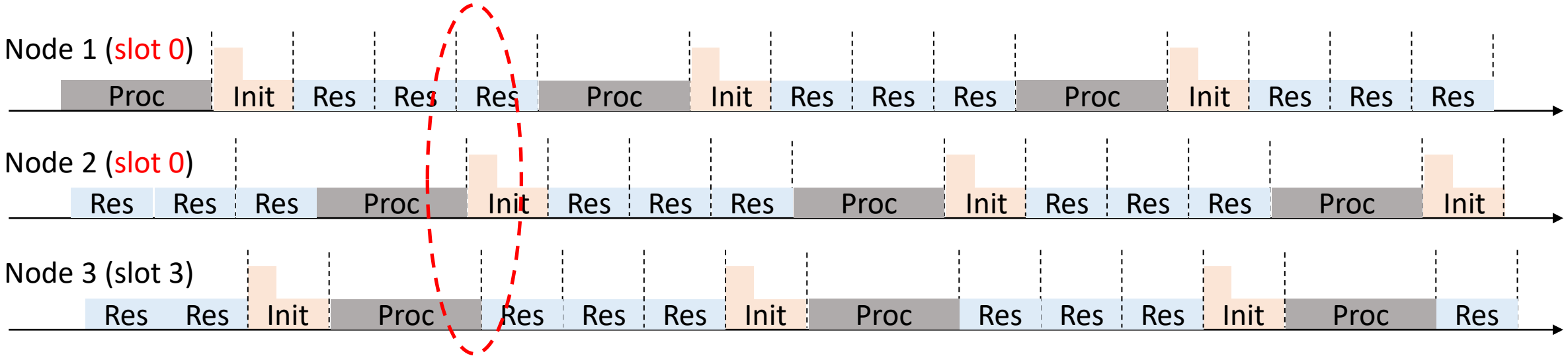
- The nodes wake up at different times
  - There is no global clock that they can use to align their slots
- There is no coordinator to assign slots to the nodes



An ideal slot boundary alignment and slot assignment

How to Synchronize the Nodes and Choose Slots (1/2)

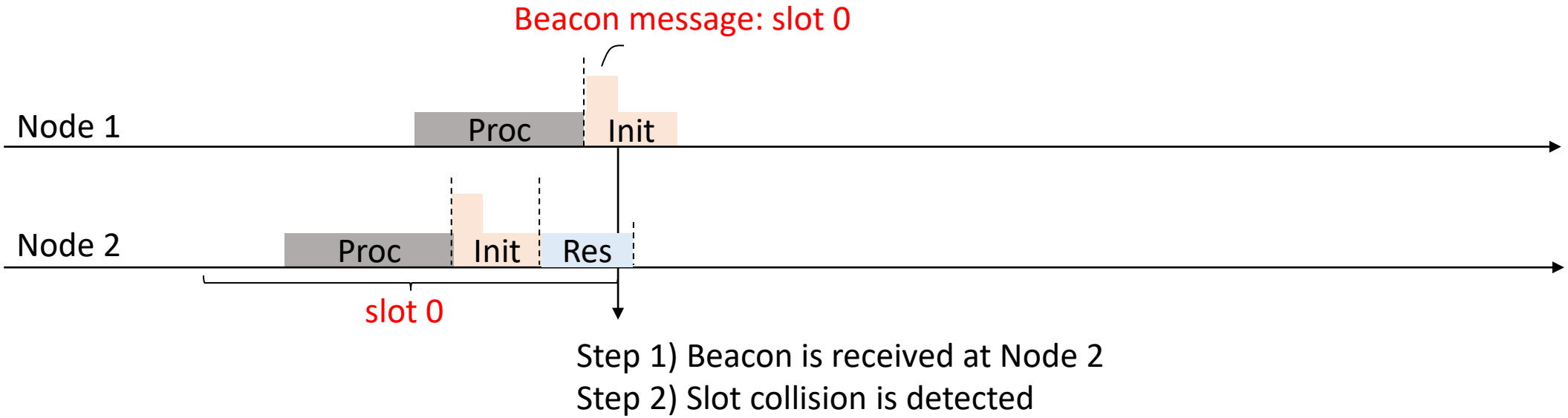
- The nodes wake up at different times
  - There is no global clock that they can use to align their slots
- There is no coordinator to assign slots to the nodes



Time synchronization and slot assignment problems

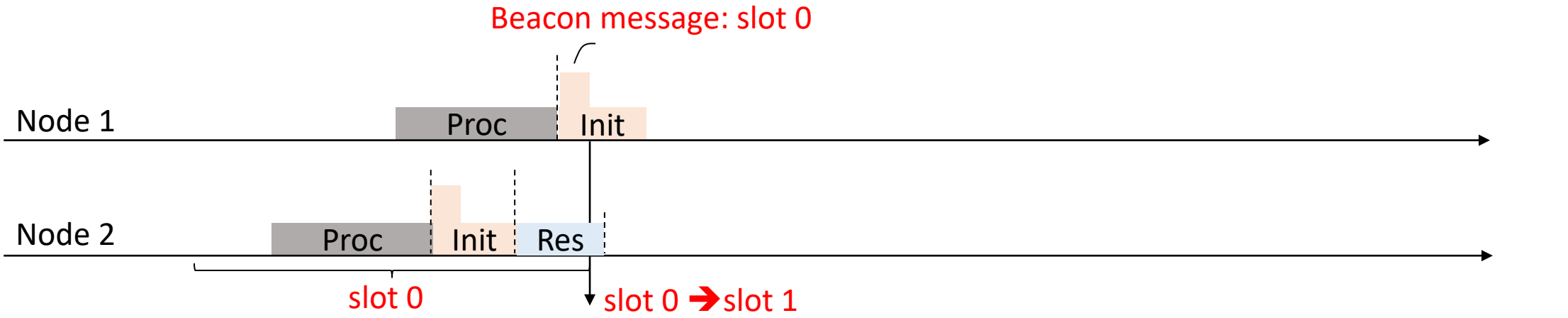
Joint Synchronization & Slot Adjustment Method

- We propose a joint synchronization & slot adjustment method:
  - 1) All the nodes broadcast a beacon signal which contains its own slot and its neighbors' slots (to address hidden node problems)
  - 2) If a node receives a beacon signal, the node selects a slot not listed in the beacon signal and adjusts its timing
- Example:



Joint Synchronization & Slot Adjustment Method

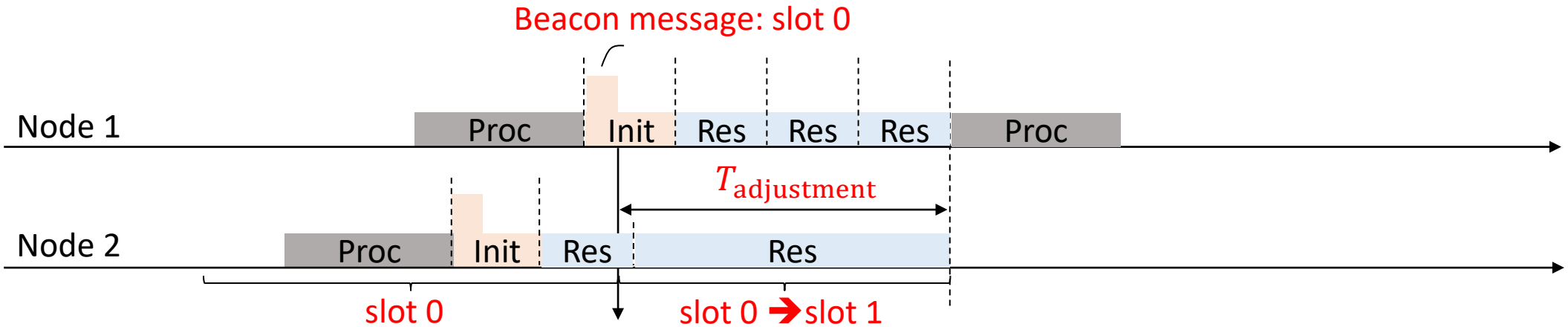
- We propose a joint synchronization & slot adjustment method:
  - 1) All the nodes broadcast a beacon signal which contains its own slot and its neighbors' slots (to address hidden node problems)
  - 2) If a node receives a beacon signal, the node selects a slot not listed in the beacon signal and adjusts its timing
- Example:



Step 1) Beacon is received at Node 2
Step 2) Slot collision is detected, hence choose a different slot, e.g., slot 1

Joint Synchronization & Slot Adjustment Method

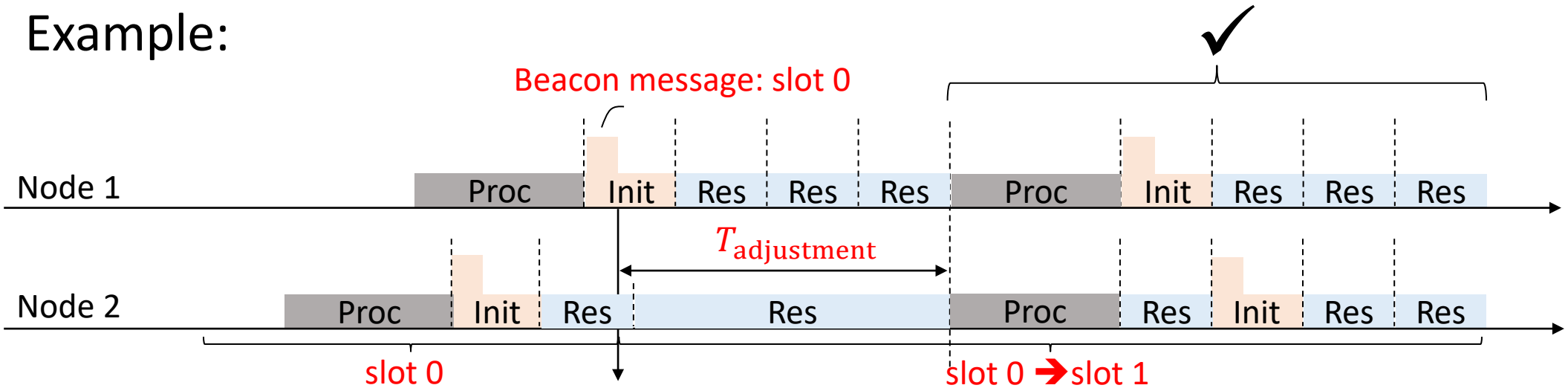
- We propose a joint synchronization & slot adjustment method:
  - 1) All the nodes broadcast a beacon signal which contains its own slot and its neighbors' slots (to address hidden node problems)
  - 2) If a node receives a beacon signal, the node selects a slot not listed in the beacon signal and adjusts its timing
- Example:



Step 1) Beacon is received at Node 2
Step 2) Slot collision is detected, hence choose a different slot, e.g., slot 1
Step 3) Calculate $T_{\text{adjustment}}$ and remain as responder

Joint Synchronization & Slot Adjustment Method

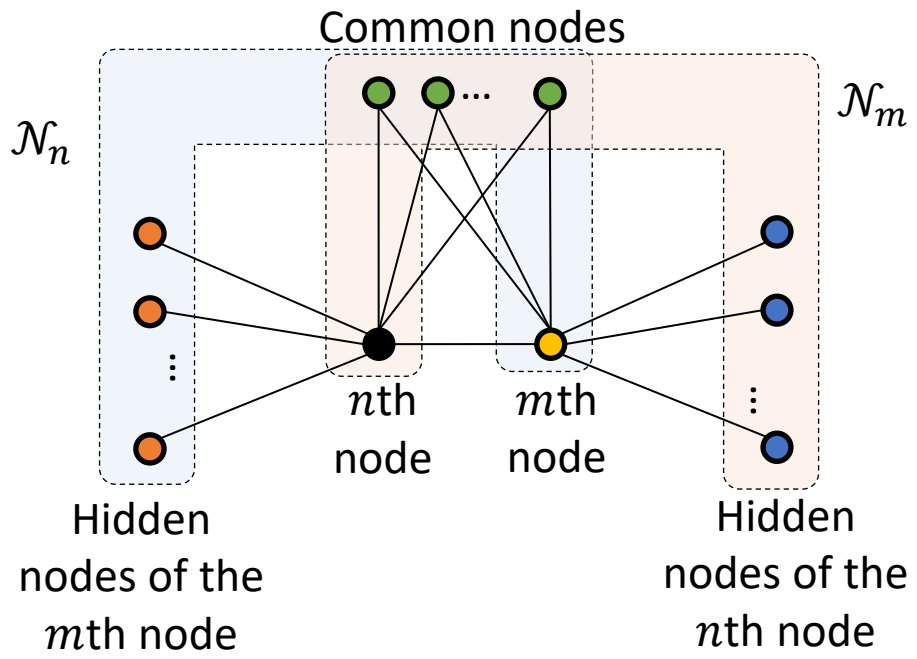
- We propose a joint synchronization & slot adjustment method:
  - 1) All the nodes broadcast a beacon signal which contains its own slot and its neighbors' slots (to address hidden node problems)
  - 2) If a node receives a beacon signal, the node selects a slot not listed in the beacon signal and adjusts its timing
- Example:



Step 1) Beacon is received at Node 2
Step 2) Slot collision is detected, hence choose a different slot, e.g., slot 1
Step 3) Calculate $T_{\text{adjustment}}$ and remain as responder

Addressing Hidden Node Problem (1/2)

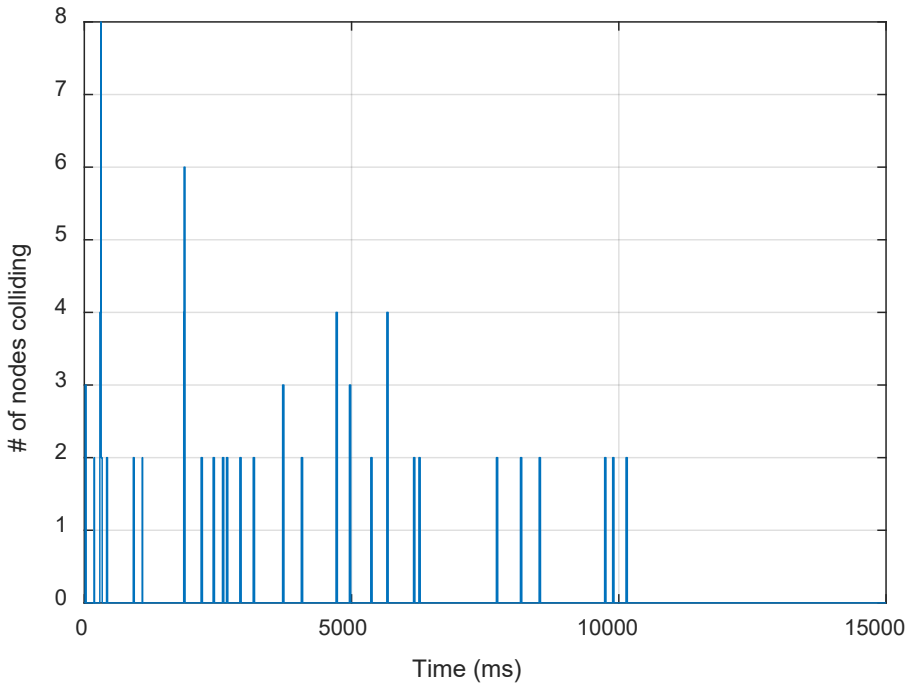
- The proposed approach reduces the collisions over time and addresses the hidden node problem

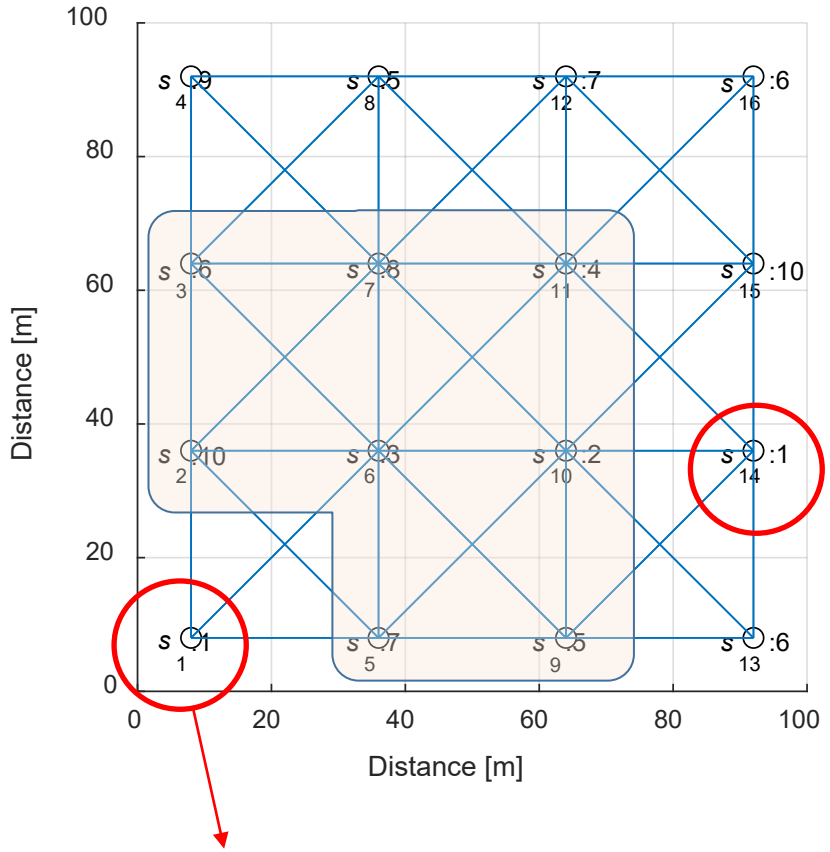


All the nodes in $\mathcal{N}_n \cup \mathcal{N}_m$ should use a different slot to avoid interference between the mth and the nth node

Addressing Hidden Node Problem (2/2)

- The proposed approach reduces the collisions over time and addresses the hidden node problem





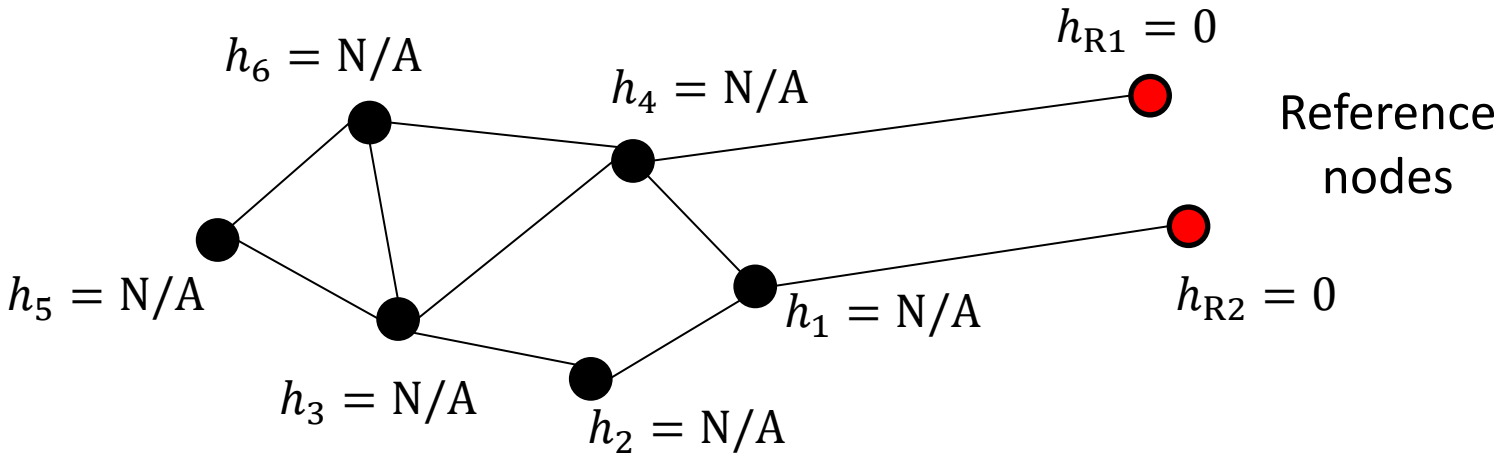
An example of slots assignment after the proposed method runs for 15 seconds (10 slots, 16 nodes): Slot 1 is not used at the neighbors and neighbors' neighbors

Solutions

Goal 2: Transferring messages from nodes to the reference nodes

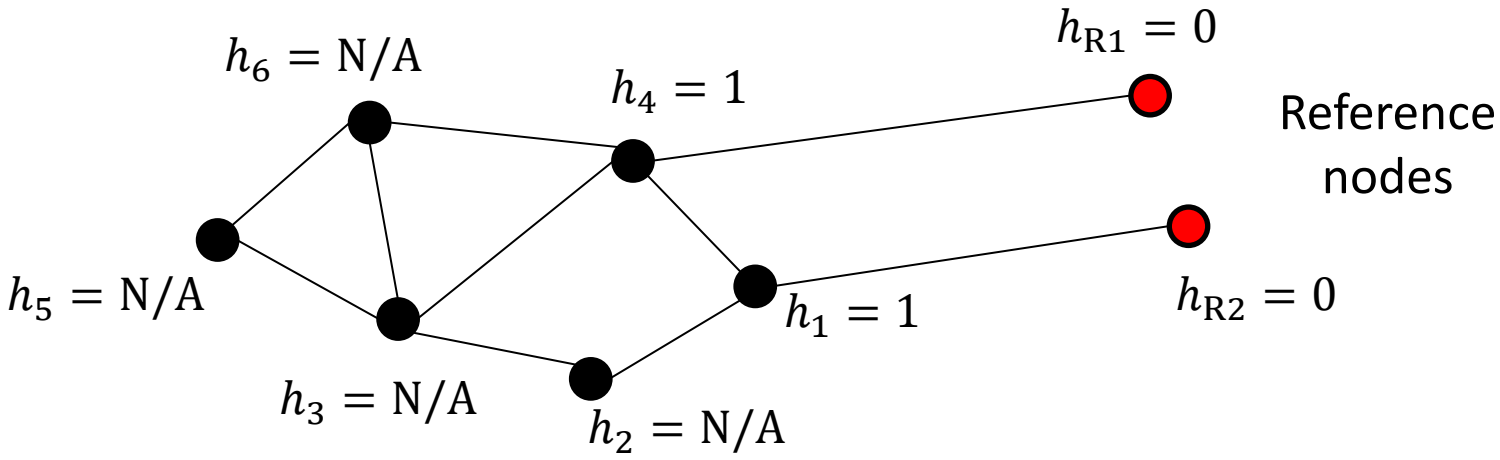
Propagating the Hopping Distance in Mesh Network

- Our approach is simple:
  - We define “hopping number” as # of hops to reach a reference node
    - The hoping number of a reference node is 0
  - The nth node determines its hopping number h_n based on its neighbors’ hopping numbers as $h_n \triangleq 1 + \min_{m \in \mathcal{N}_n} h_m$
  - Example:



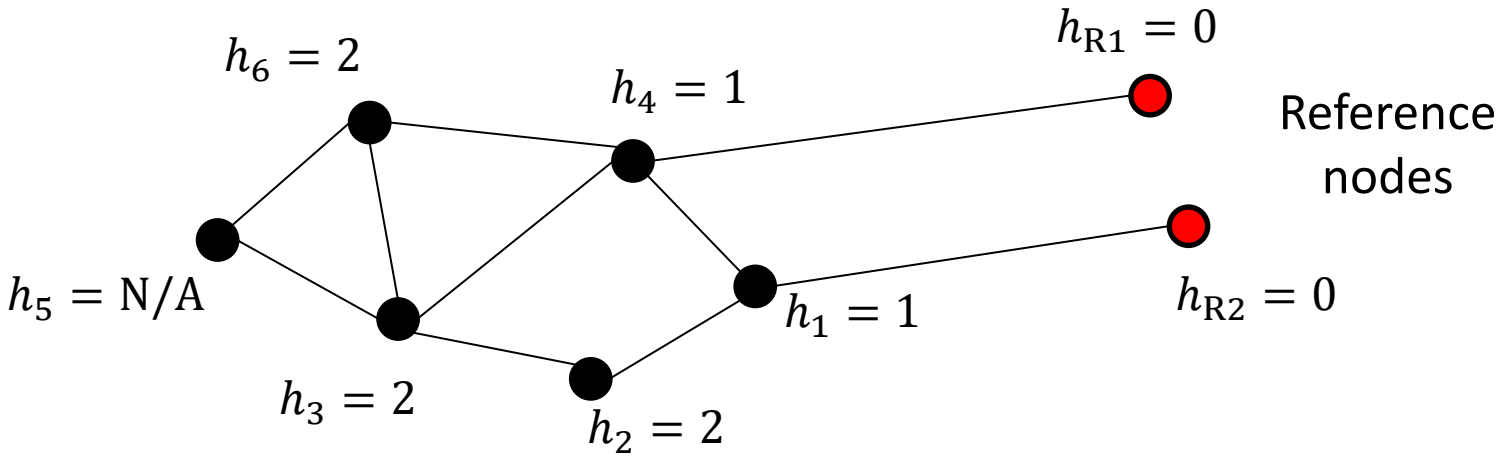
Propagating the Hopping Distance in Mesh Network

- Our approach is simple:
  - We define “hopping number” as # of hops to reach a reference node
    - The hoping number of a reference node is 0
  - The nth node determines its hopping number h_n based on its neighbors’ hopping numbers as $h_n \triangleq 1 + \min_{m \in \mathcal{N}_n} h_m$
  - Example:



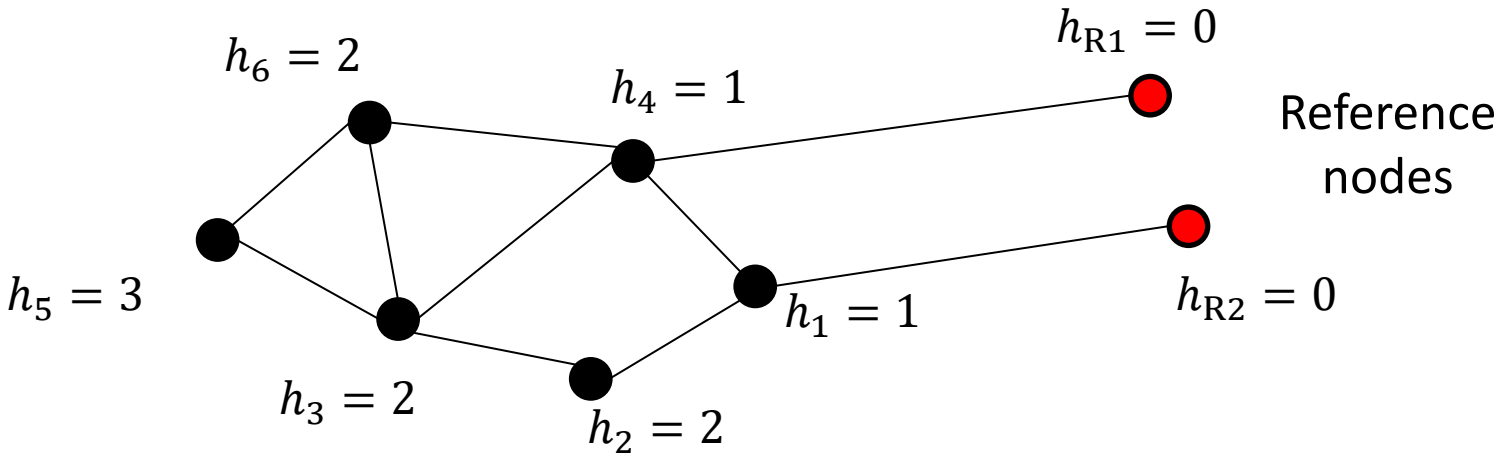
Propagating the Hopping Distance in Mesh Network

- Our approach is simple:
  - We define “hopping number” as # of hops to reach a reference node
    - The hoping number of a reference node is 0
  - The nth node determines its hopping number h_n based on its neighbors’ hopping numbers as $h_n \triangleq 1 + \min_{m \in \mathcal{N}_n} h_m$
  - Example:



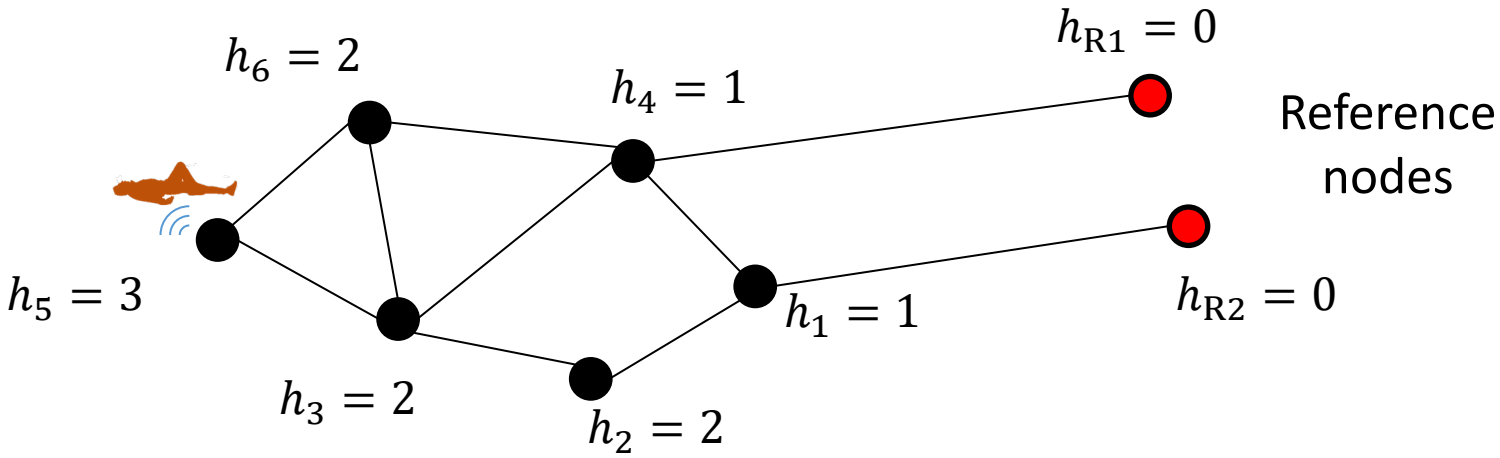
Propagating the Hopping Distance in Mesh Network

- Our approach is simple:
  - We define “hopping number” as # of hops to reach a reference node
    - The hoping number of a reference node is 0
  - The nth node determines its hopping number h_n based on its neighbors’ hopping numbers as $h_n \triangleq 1 + \min_{m \in \mathcal{N}_n} h_m$
  - Example:



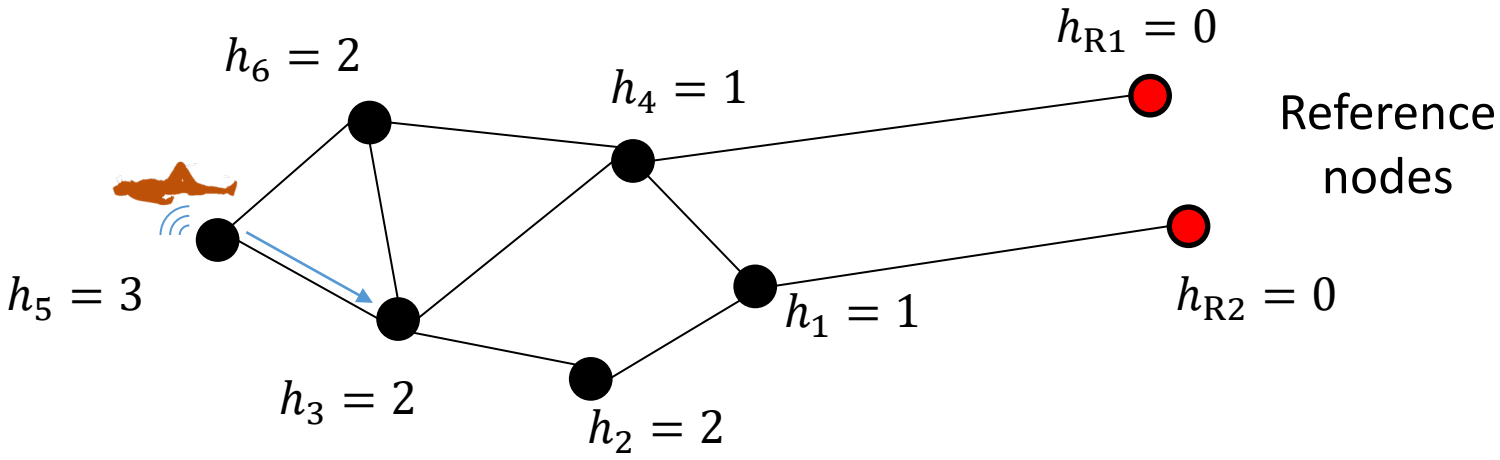
Transmitting Message Based on Hopping Distance

- Our approach is simple:
  - We define “hopping number” as # of hops to reach a reference node
    - The hoping number of a reference node is 0
  - The nth node determines its hopping number h_n based on its neighbors’ hopping numbers as $h_n \triangleq 1 + \min_{m \in \mathcal{N}_n} h_m$
  - If a node transmits a message, it forwards its message to a node that has the smallest hopping number in its neighbor list



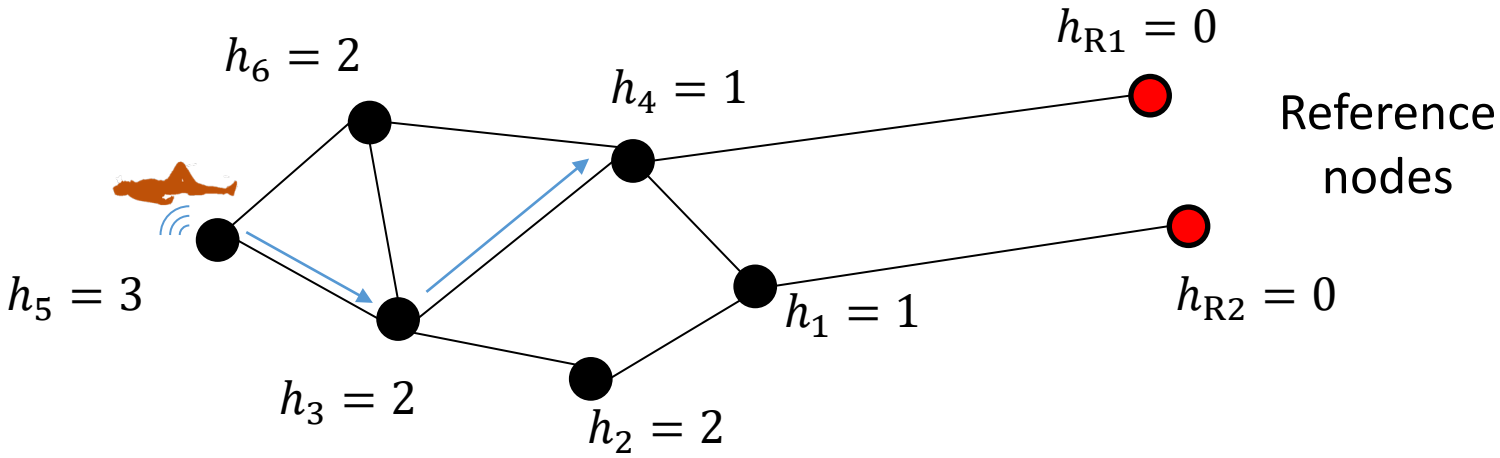
Transmitting Message Based on Hopping Distance

- Our approach is simple:
  - We define “hopping number” as # of hops to reach a reference node
    - The hoping number of a reference node is 0
  - The nth node determines its hopping number h_n based on its neighbors’ hopping numbers as $h_n \triangleq 1 + \min_{m \in \mathcal{N}_n} h_m$
  - If a node transmits a message, it forwards its message to a node that has the smallest hopping number in its neighbor list



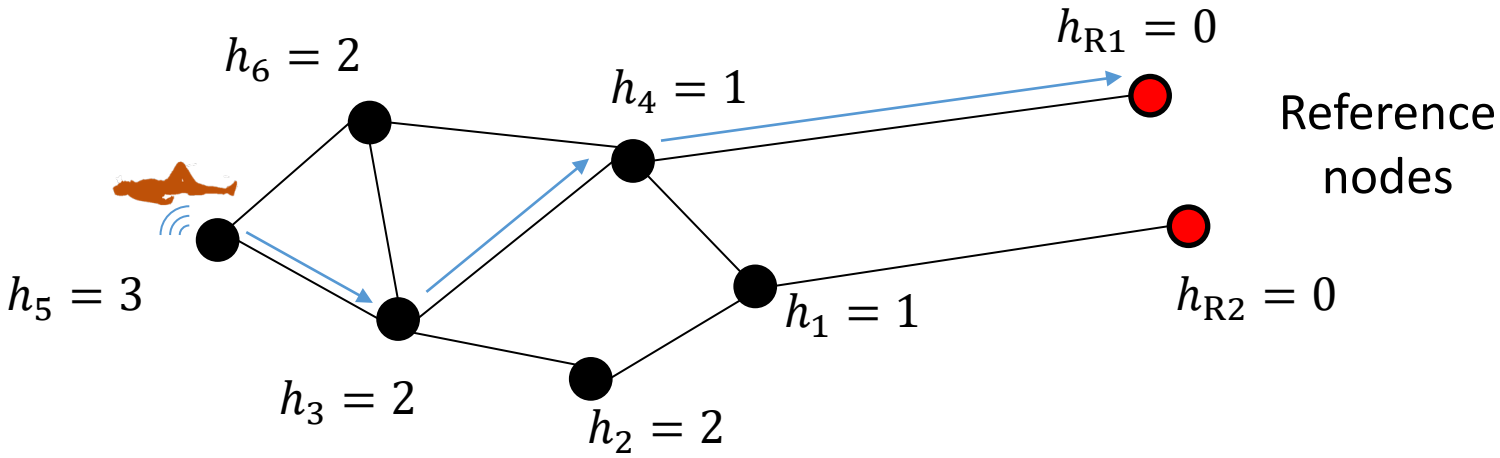
Transmitting Message Based on Hopping Distance

- Our approach is simple:
  - We define “hopping number” as # of hops to reach a reference node
    - The hoping number of a reference node is 0
  - The nth node determines its hopping number h_n based on its neighbors’ hopping numbers as $h_n \triangleq 1 + \min_{m \in \mathcal{N}_n} h_m$
  - If a node transmits a message, it forwards its message to a node that has the smallest hopping number in its neighbor list



Transmitting Message Based on Hopping Distance

- Our approach is simple:
  - We define “hopping number” as # of hops to reach a reference node
    - The hoping number of a reference node is 0
  - The nth node determines its hopping number h_n based on its neighbors’ hopping numbers as $h_n \triangleq 1 + \min_{m \in \mathcal{N}_n} h_m$
  - If a node transmits a message, it forwards its message to a node that has the smallest hopping number in its neighbor list

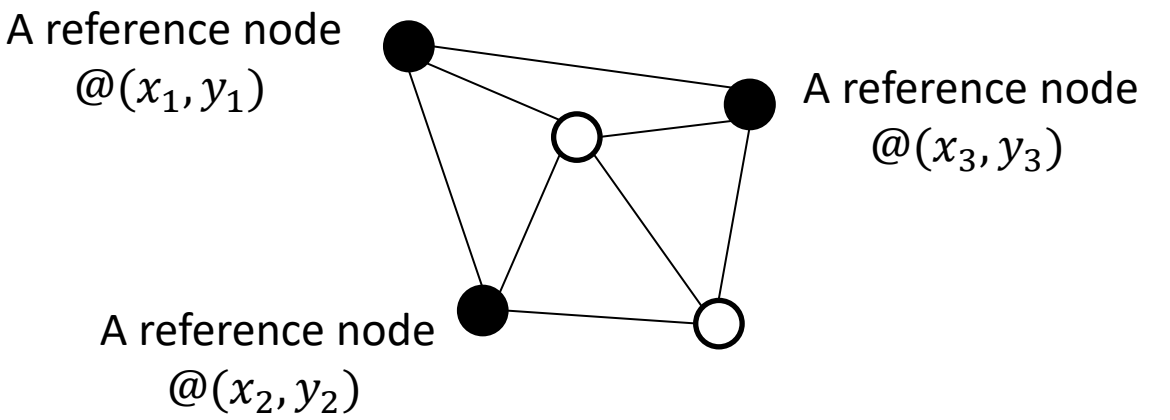


Solutions

Goal 3: Finding the positions of the nodes to find the trapped people

Iterative Localization in Mesh

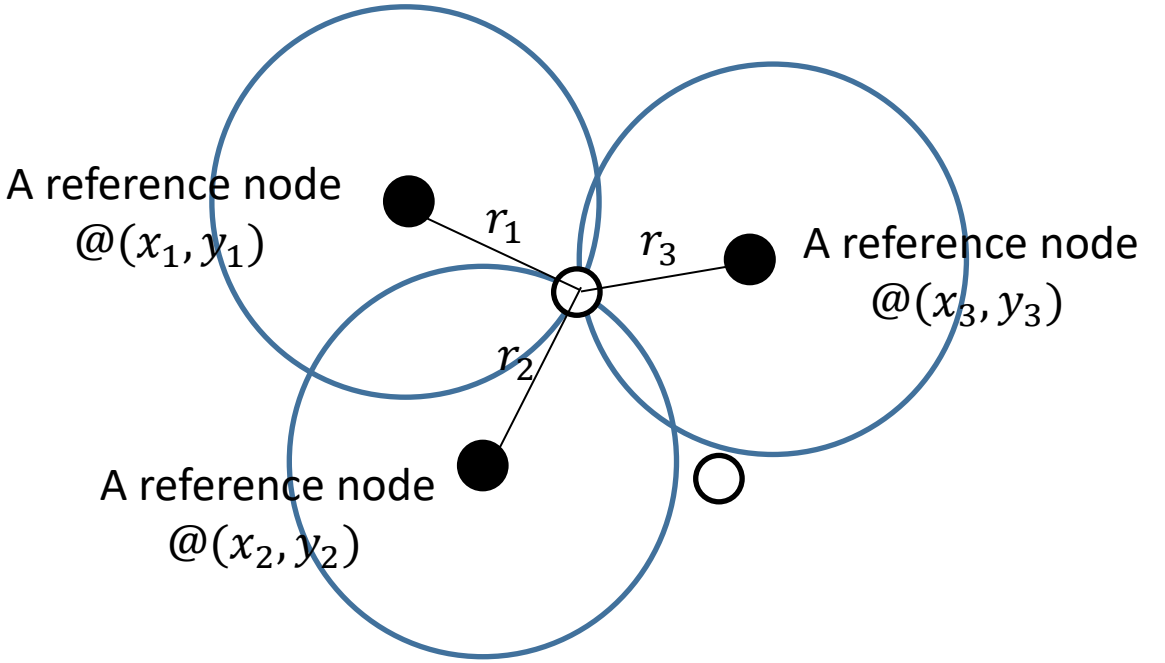
- If a node has at least four nodes knowing their positions, it can estimate its position in 3D with the state-of-the-art methods (e.g., two-way ranging)
- Our approach relies on discovering the position of the nodes iteratively
  - We assume that all the reference node knows their position information
- Example:



(assume that three node is enough for localization)

Iterative Localization in Mesh

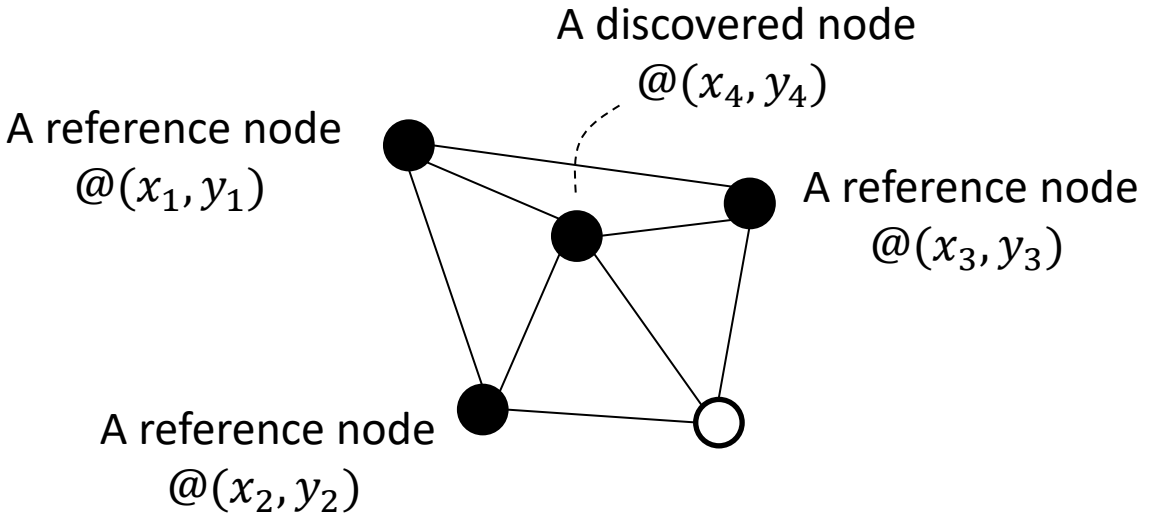
- If a node has at least four nodes knowing their positions, it can estimate its position in 3D with the state-of-the-art methods (e.g., two-way ranging)
- Our approach relies on discovering the position of the nodes iteratively
  - We assume that all the reference node knows their position information
- Example:



(assume that three node is enough for localization)

Iterative Localization in Mesh

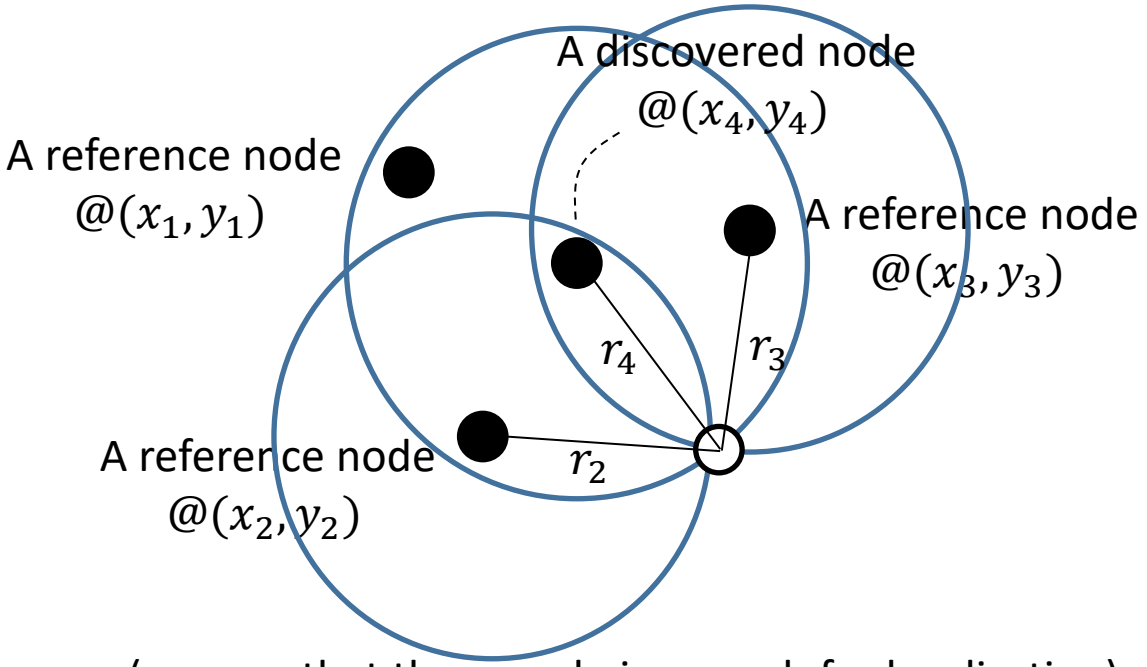
- If a node has at least four nodes knowing their positions, it can estimate its position in 3D with the state-of-the-art methods (e.g., two-way ranging)
- Our approach relies on discovering the position of the nodes iteratively
  - We assume that all the reference node knows their position information
- Example:



(assume that three node is enough for localization)

Iterative Localization in Mesh

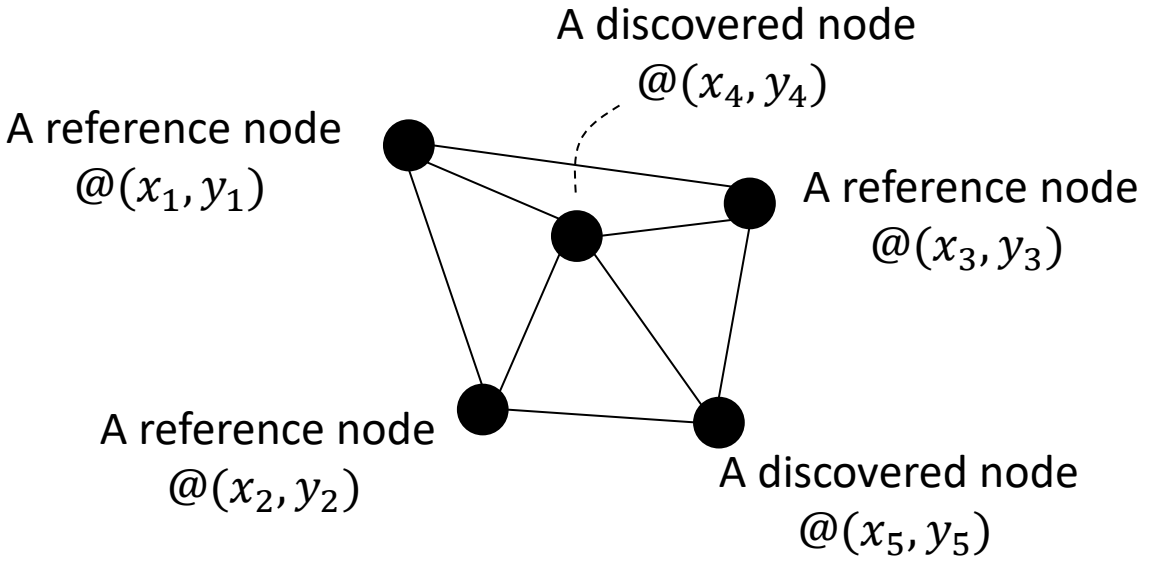
- If a node has at least four nodes knowing their positions, it can estimate its position in 3D with the state-of-the-art methods (e.g., two-way ranging)
- Our approach relies on discovering the position of the nodes iteratively
  - We assume that all the reference node knows their position information
- Example:



(assume that three node is enough for localization)

Iterative Localization in Mesh

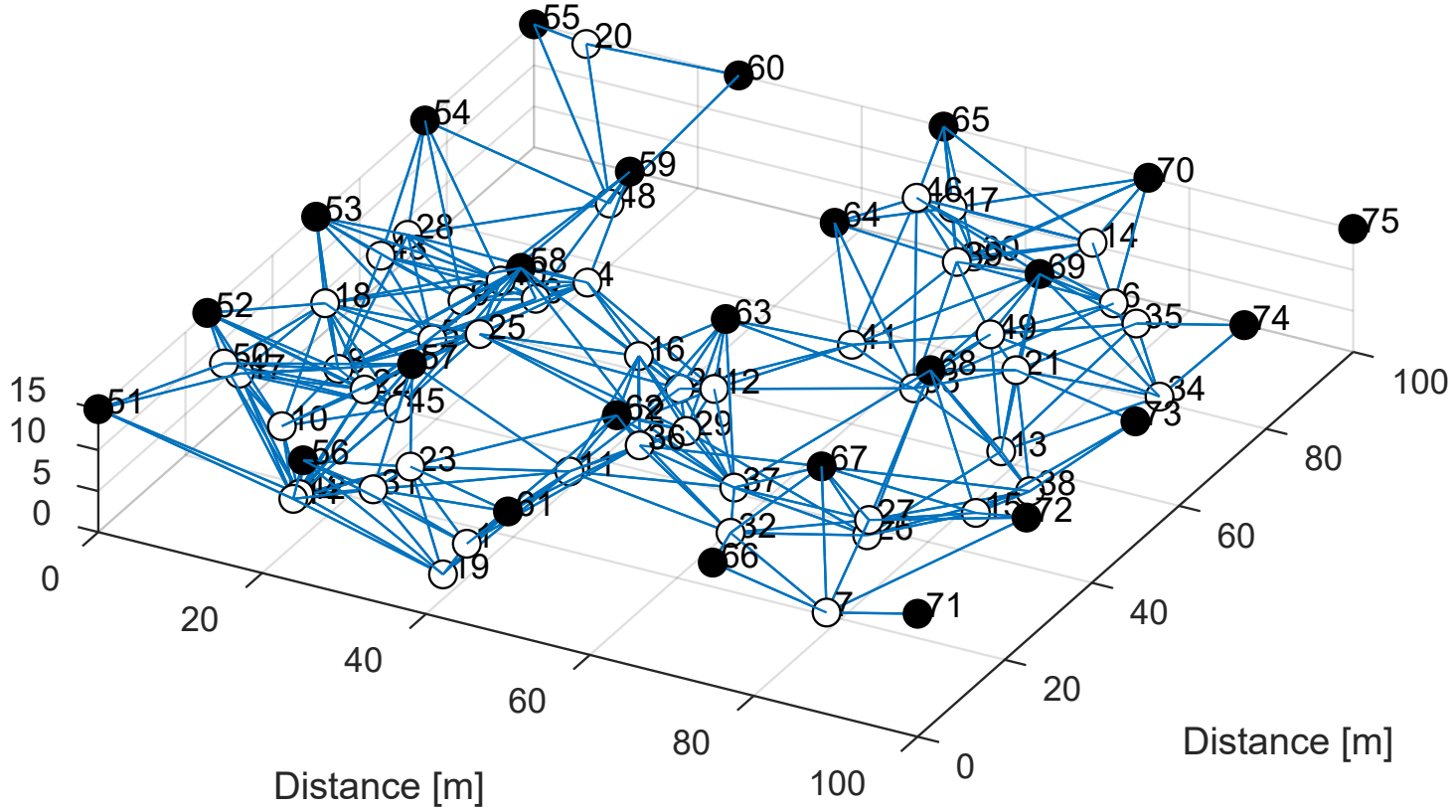
- If a node has at least four nodes knowing their positions, it can estimate its position in 3D with the state-of-the-art methods (e.g., two-way ranging)
- Our approach relies on discovering the position of the nodes iteratively
  - We assume that all the reference node knows their position information
- Example:



(assume that three node is enough for localization)

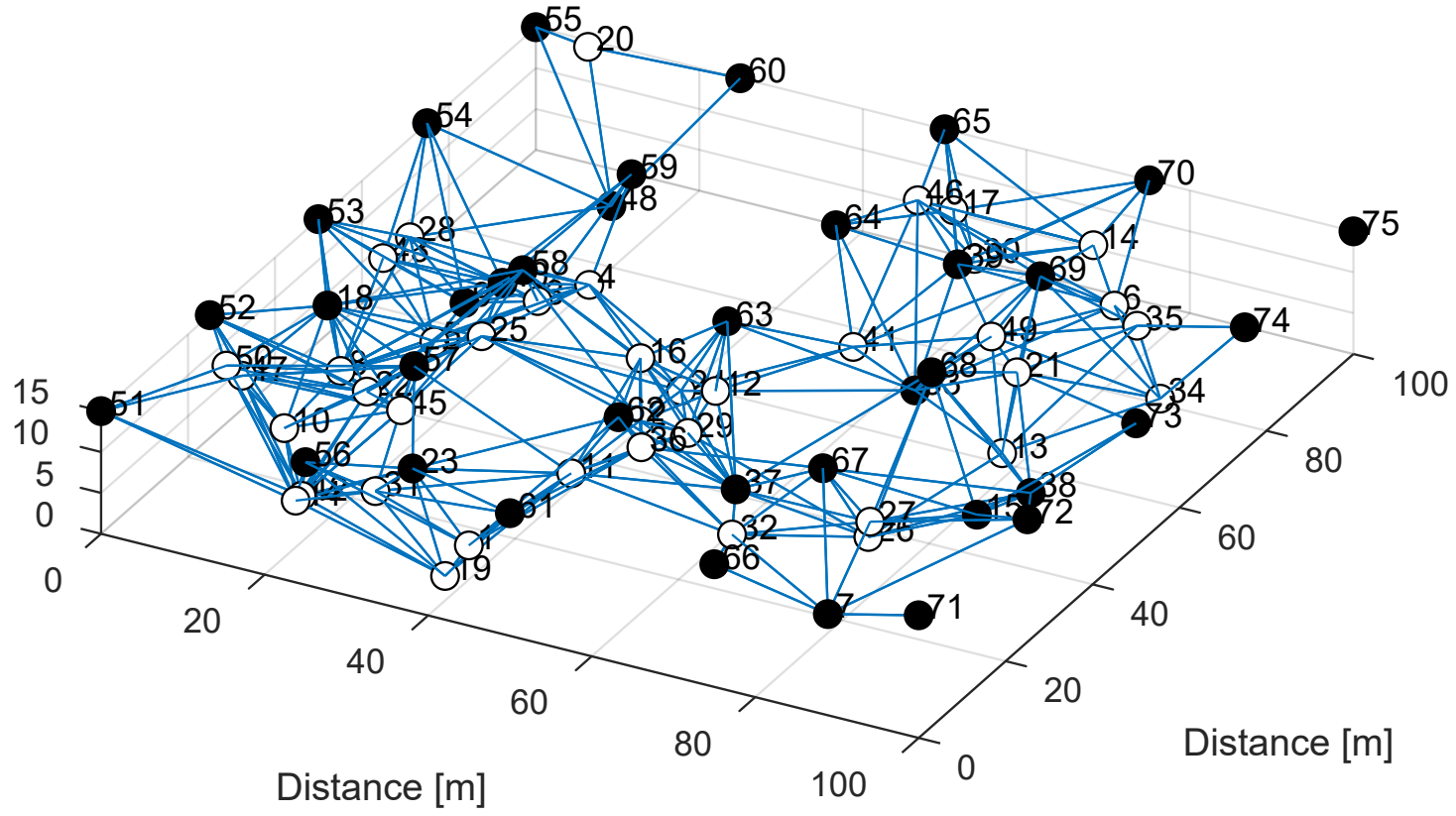
An Example in 3D Geometry

- 25 reference nodes (black ones), 50 nodes under the rubble (white ones)
  - At $t = 0$, none of the nodes know its position



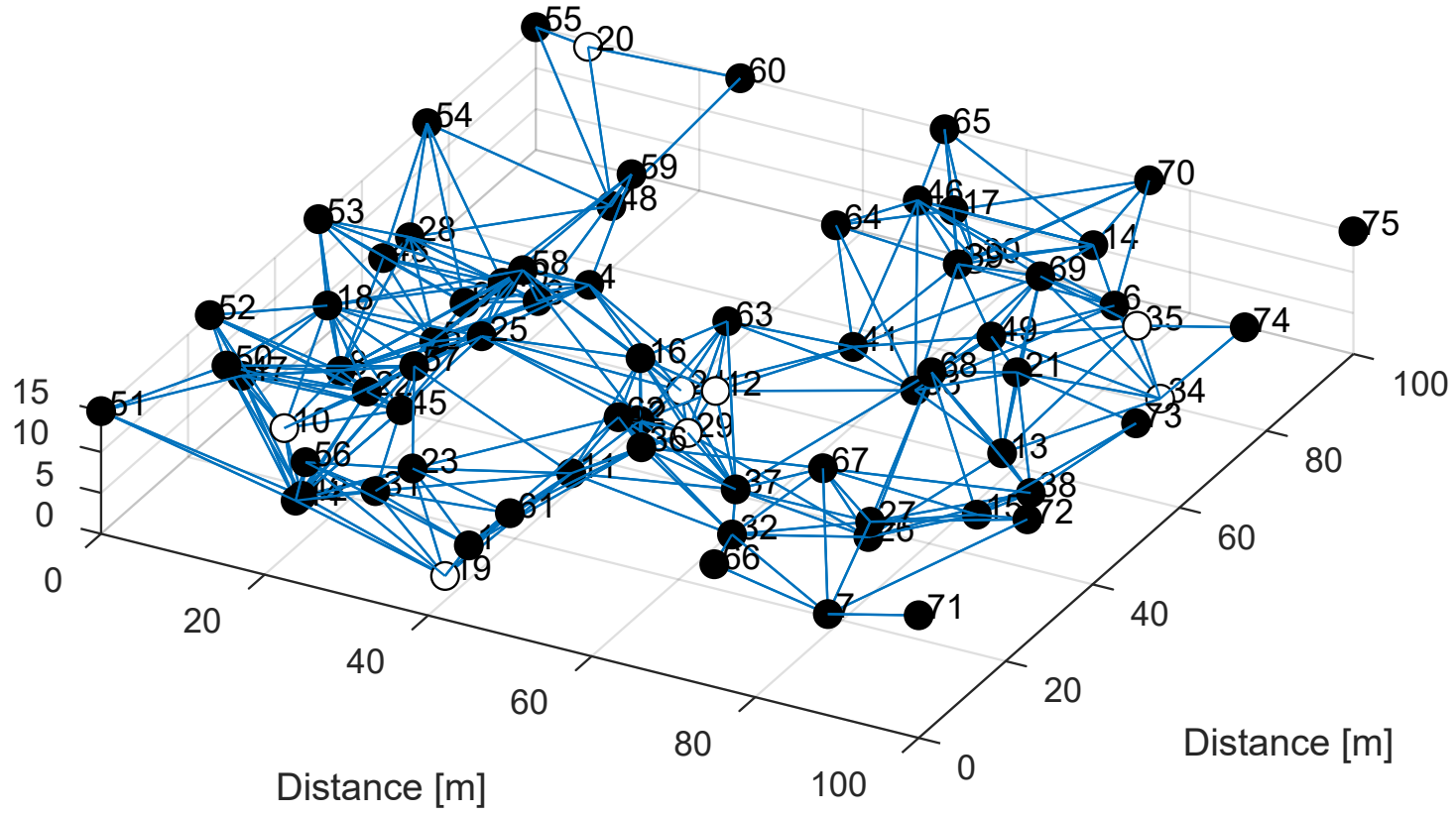
An Example in 3D Geometry

- 25 reference nodes (black ones), 50 nodes under the rubble (white ones)
  - At $t = 1$, 22% of the nodes learn their positions



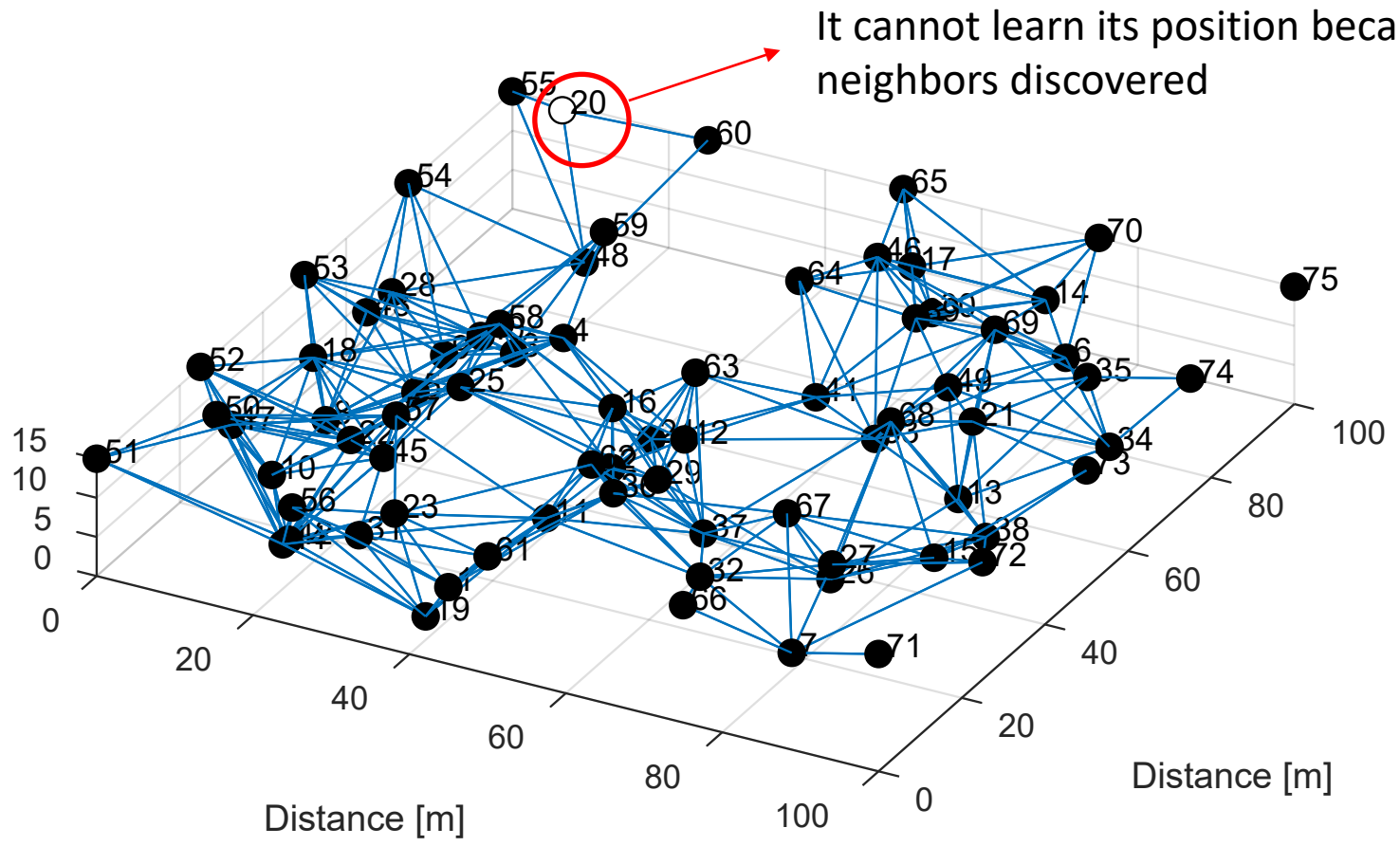
An Example in 3D Geometry

- 25 reference nodes (black ones), 50 nodes under the rubble (white ones)
  - At $t = 2$, 82% of the nodes learn their positions



An Example in 3D Geometry

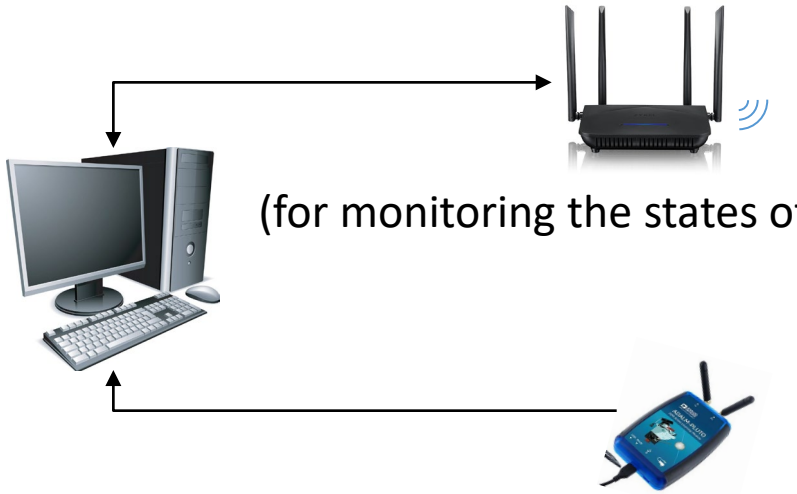
- 25 reference nodes (black ones), 50 nodes under the rubble (white ones)
  - At $t = 3$, 98% of the nodes learn their positions



Demonstration

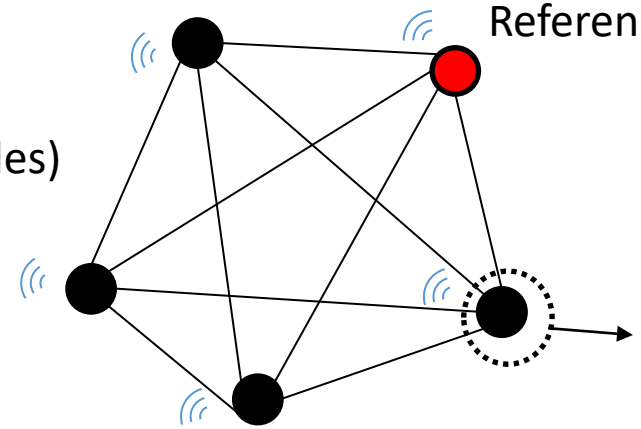
Setup

- A software defined radio is used for monitoring the signal in the air
- All nodes are connected to a router over Wi-Fi to monitor their states
- 5 nodes (SX1280 LoRa @ 2.5 GHz)
  - One of them is set as a reference node
  - $T_{\text{processing}} = 100$ ms, $T_{\text{communication}} = 200$ ms, $N_{\text{slot}} = 8$

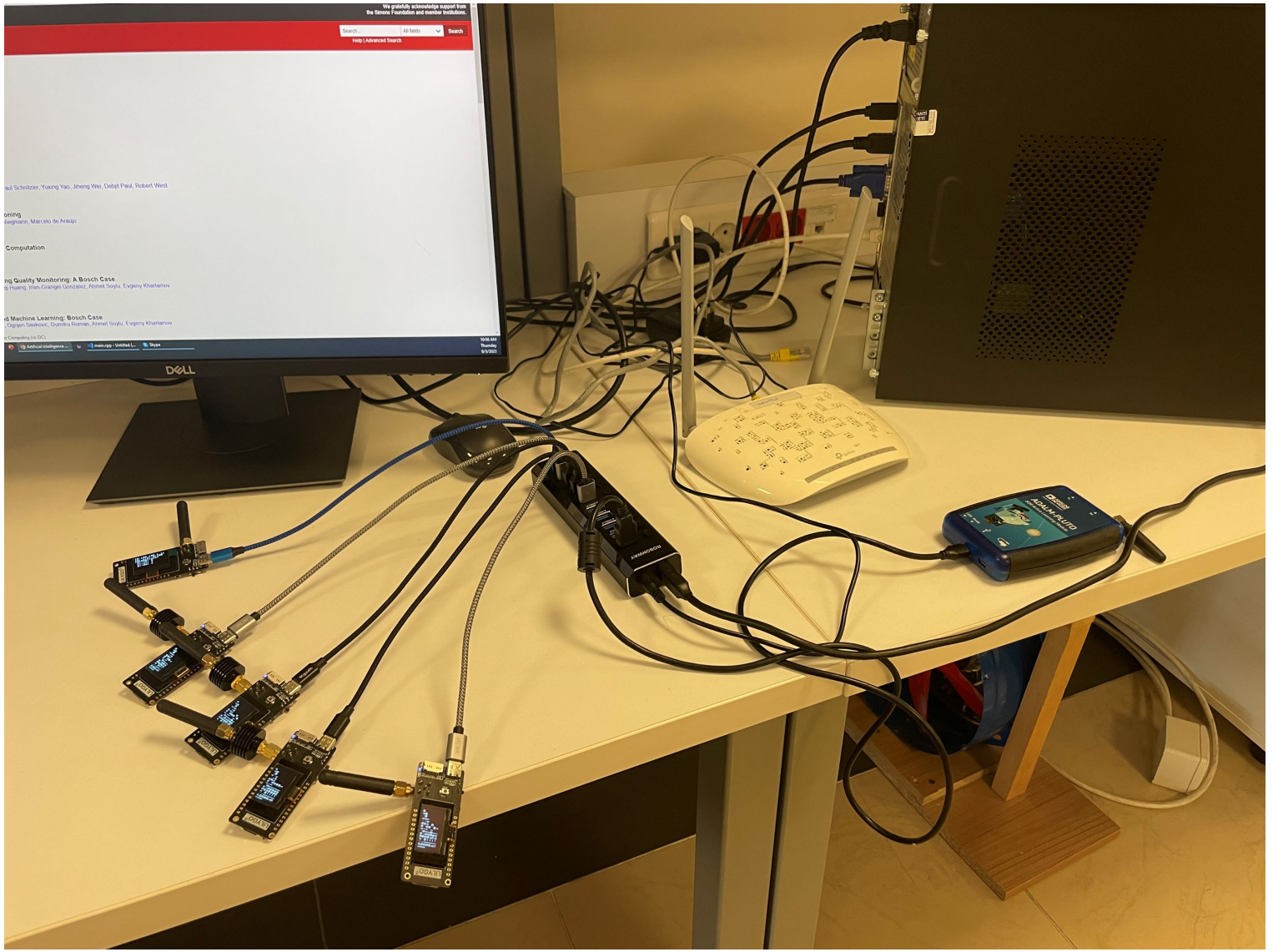


(for monitoring the states of nodes)

(for monitoring the signal in the air)

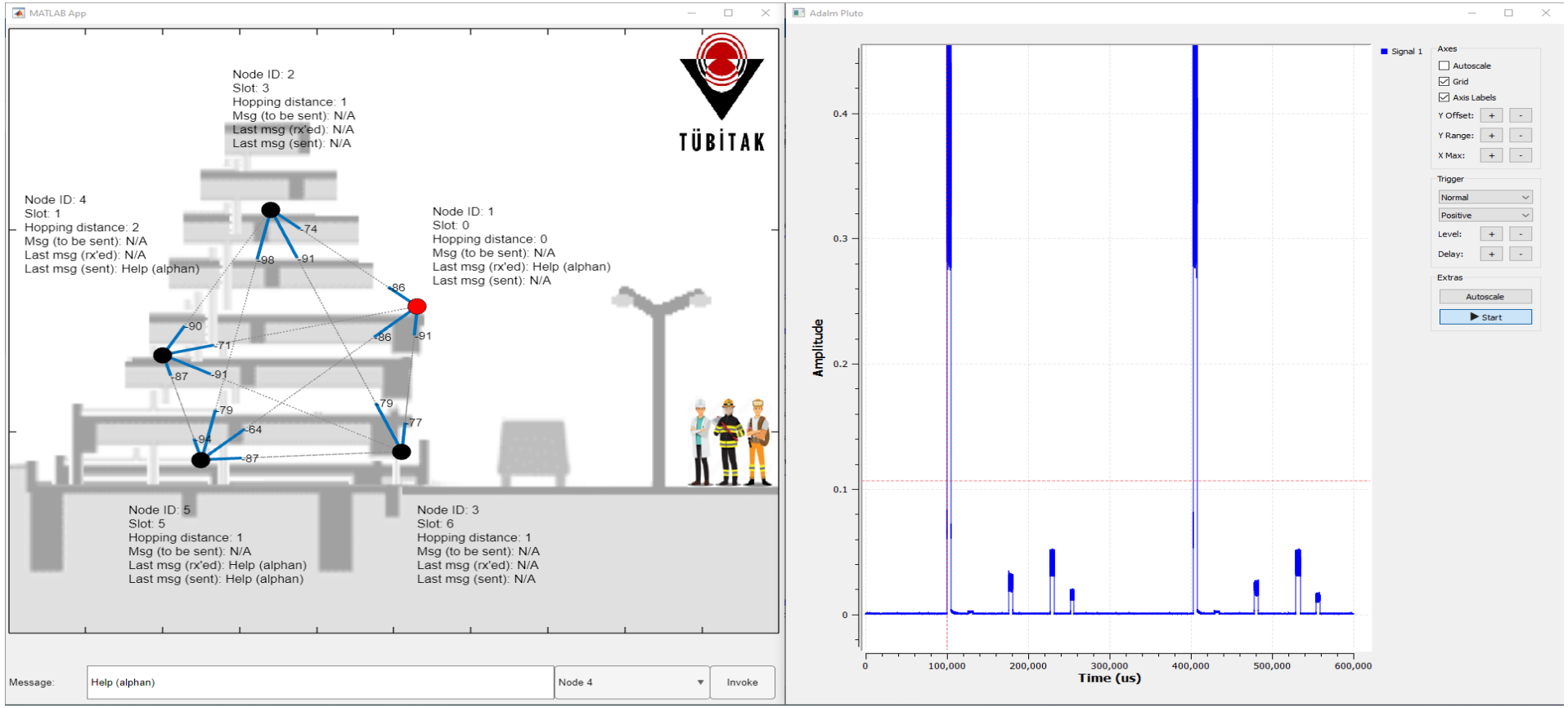


Setup



GUI and Signals Over the Air

- Result 1: All the nodes find their slots autonomously and adjust their timings
- Result 2: All nodes learn their hopping distance autonomously
- Result 3: The message from Node 4 reaches to the Node 1 over Node 5



Questions?