Federation of Wireless Sensor Network Segments

Mohamed Younis

Department of Computer Science & Electrical Engineering
University of Maryland Baltimore County
TEL: (410) 455-3968
E-mail: *younis@cs.umbc.edu*
http://www.cs.umbc.edu/~younis

Embedded Systems and Networks Lab.

Esnet.cs.umbc.edu

Faculty: Mohamed Younis

Students:

- Current: 10 Ph.D. (+2 more in foreign universities), 2 MS
- Graduated: 6 Ph.D. , 41 MS degrees

Collaborators:

- Johns Hopkins Advanced Physics Lab.
- Pacific Northwest National Laboratory
- King Fahd University for Petroleum and Minerals
- King Saud University
- Old Dominion University
- NASA Goddard Research Lab.
- Honeywell

Current Research Projects

1) Federating Disjoint Wireless Sensor Networks
2) Effective Design of Underwater Acoustic Networks
3) Sustaining Anonymity of Critical Nodes in Ad-hoc and Sensor Networks
4) Non-invasive and Efficient Leak Detection System for Long Pipelines
5) Vehicular Networking for Intelligent and Autonomous Traffic Management
6) Wearable Multimodal System for the Remote Monitoring of Epilepsy Patients

Sponsors

- National Science Foundation
- National Security Agency
- Industry (Honeywell, AetherSystems, etc.)

More information can be found on "esnet.cs.umbc.edu"

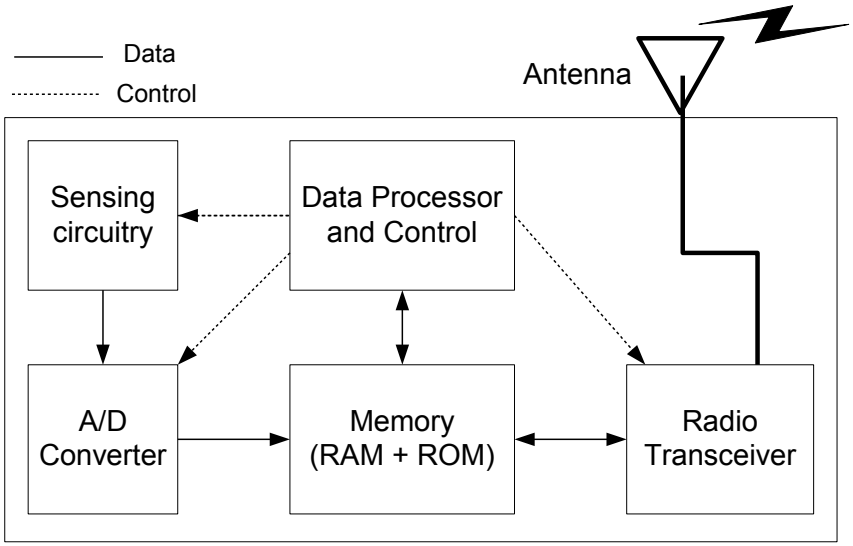
Other Technical Interest

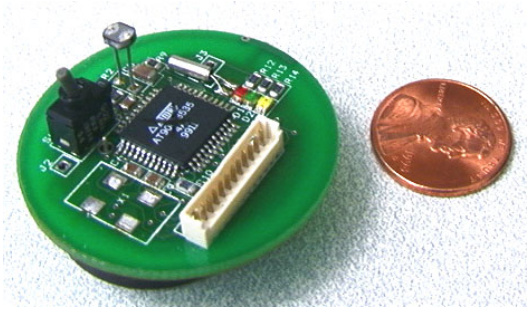
- Network Clustering and bootstrapping
  - Energy efficient sensor discovery protocol
  - Handling of time lag in nodes' deployment
  - Efficient assignment of sensors to gateways
  - Detection and tolerance of gateway failure
- Distributed location determination
  - Overcome the lack of a reference coordinate system, e.g. GPS
  - Accurate Anchor-free Localization
  - Scalable for large networks
- Energy-aware management for sensor network
  - Energy and context aware routing of sensor's data
  - Efficient Routing of QoS traffic
  - Gateway relocation for enhanced performance
  - Handling mobile gateways
  - MAC level energy and collision management
- Secure operation of sensor networks
  - Security architecture and protocols
  - Light weight key management

✓ ***Numerous publications***

Sensor Node

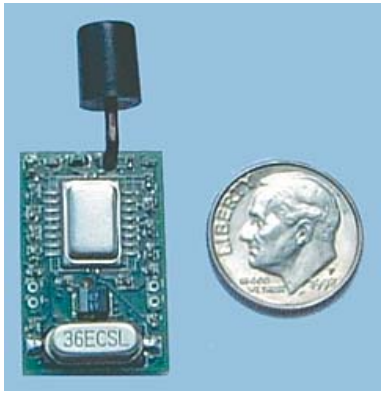
- The sensing circuitry measures ambient conditions and transforms them into an electric signal
- Signal processing reveals some properties about the vicinity of the sensor
- Collected reports are sent (broadcasted) via radio transmitter
- Very miniaturized and battery operated circuits





A prototype Mote sensor developed at Univ. of California at Berkley (RF 916.5 MHz 10kbps, 20 meter range)

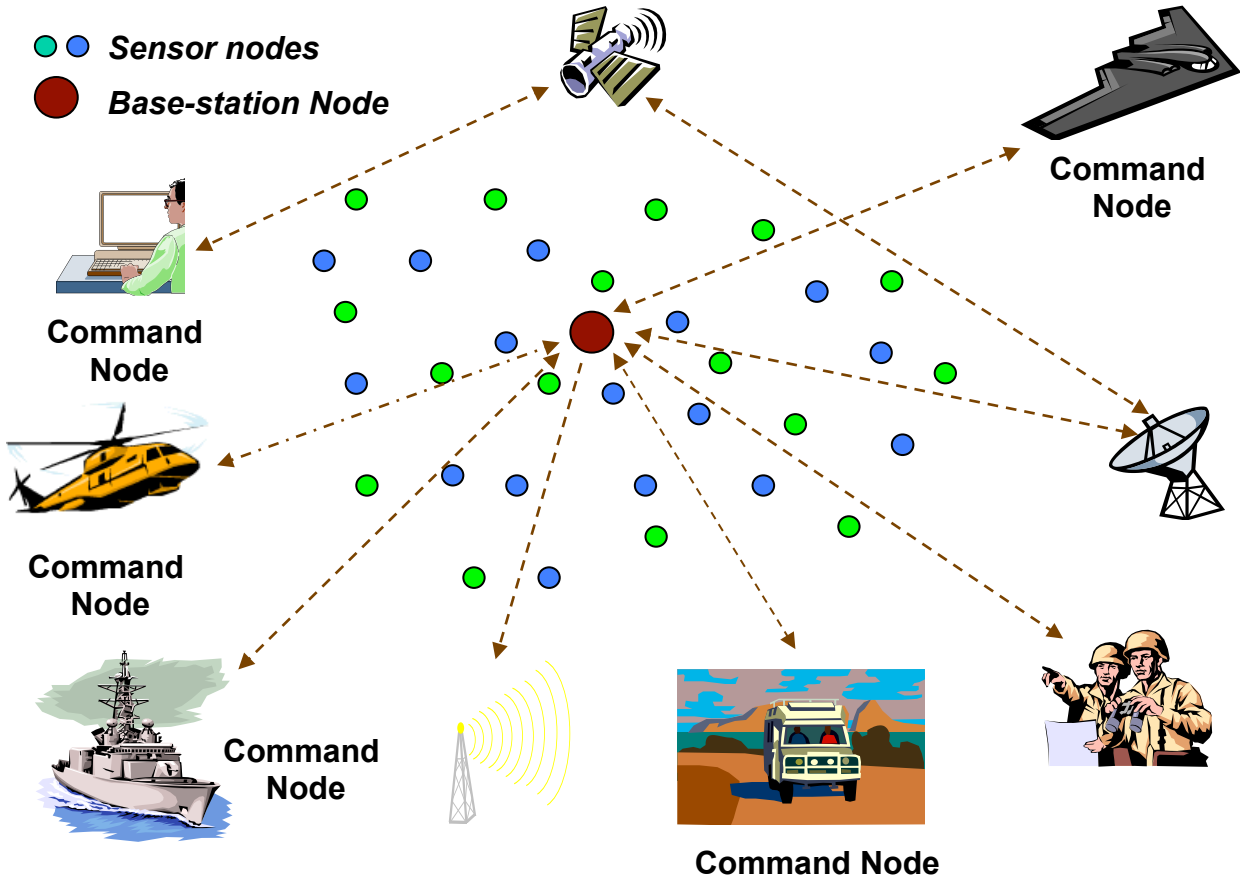
Current offerings in the market include, but not restricted to, Temperature, Humidity, Light, Barometric Pressure, Imaging, Motion detection, Acoustics, etc.



A wireless sensor node developed at Millennial Net Inc.

Wireless Sensor Networks (WSNs)

- A collection of sensors connected and dynamically configured in ad hoc topologies to provide support for detection and monitoring applications
- Aggregating readings of numerous sensors for increased data fidelity
- Mission-oriented deployment in unattended setup possibly with no node energy re-supply
- A large number of nodes with a dynamically changing membership



"Sensor networks are massive numbers of small, inexpensive devices pervasive throughout electrical and mechanical systems and ubiquitous throughout the environment that monitor and control most aspects of our physical world."

National Research Council

Dependability Requirements for WSNs

- Longevity
  - Network should fulfill its task as long as possible – definition depends on application
  - Lifetime of individual nodes relatively unimportant -- But often treated equivalently since it may partition the network and risk coverage
- Quality of service
  - Service of WSN must be “good”: Right answers at the right time
- Scalability
  - Support large number of nodes
- ✓ **Reliability**
  - Be robust against node failure (running out of energy, physical destruction, …)
  - Reliable data delivery (e.g. limiting packet losses
- Security
  - Secure data collection and dissemination (resilience to attacks)
  - Protect the valuable network assets, e.g. sink node, from physical damage
- Maintainability
  - WSN has to adapt to changes, self-monitoring, adapt operation
  - Incorporate possible additional resources, e.g., newly deployed nodes

Recovery from Node Failure

- Failure of a node causes
  - Loss of connectivity
  - Reduction or holes in coverage
- Failure recovery should be
  - Rapid, to maintain QoS
  - Self-healing in risky areas
  - With minimal overhead

Problem Statement

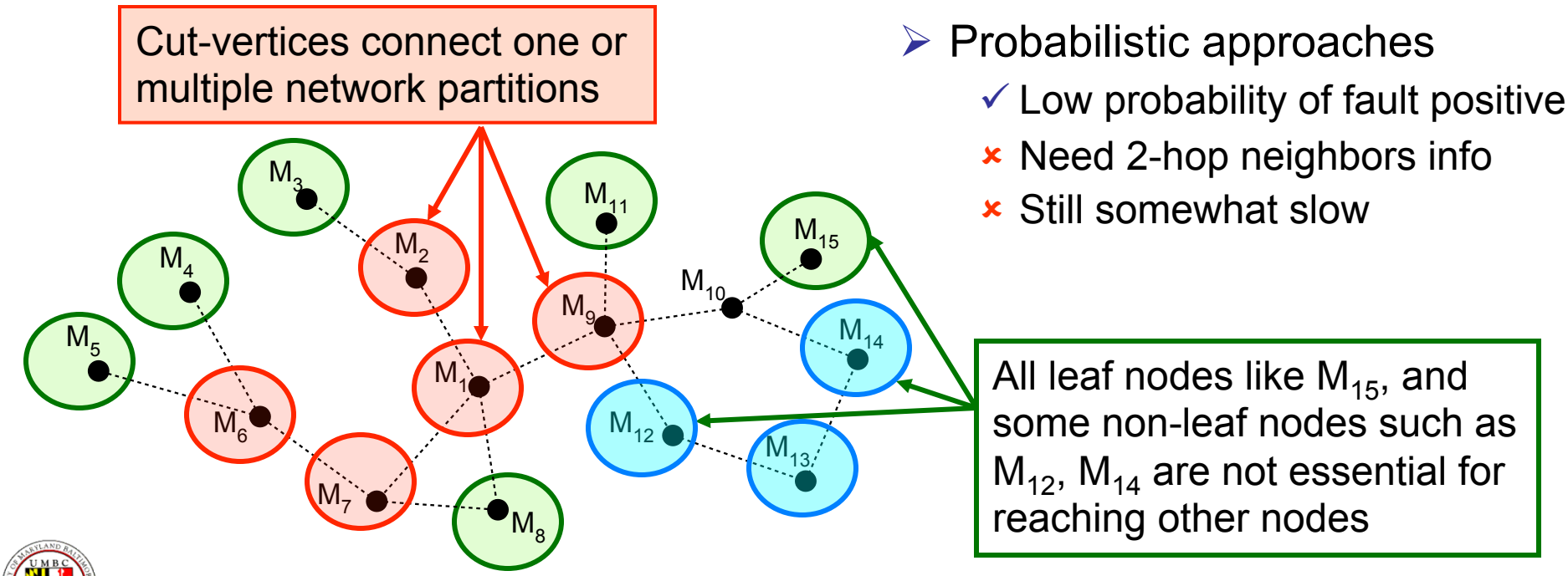
- Recover from connectivity loss
- Sustain pre-failure coverage
- Unsupervised recovery
- Spare nodes may not be available

- Sensing and Communication Ranges
  - Sensing < Communication => Coverage problem
  - Communication < Sensing => Connectivity problem
  - Coverage may not be an issue (2nd Tier topology of multiple actors or base-stations)
- Although redundancy may alleviate the risk, surviving successive failures cannot be guaranteed
- Fixing coverage holes is often done using modeling the network using Voronoi diagram or inter-node forces (not discussed in this presentation)

Topological Impact of Node Failure

- **Effect on network connectivity**
  - Very limited
    - ✓ Lost of leaf or non-leaf will not affect other sensors connectivity
  - Very dramatic / Serious
    - ✓ Loss of a cut-vertex inflicts a serious damage to connectivity

- **Required Analysis:**
  - Finding cut-vertices requires depth first search at each node
    - ✓ Limited engagement of nodes
    - × Analysis is wide in scope
    - × Lots of state update
    - × Slow recovery
    - × does not scale
  - Probabilistic approaches
    - ✓ Low probability of fault positive
    - × Need 2-hop neighbors info
    - × Still somewhat slow



Connectivity Restoration Methodologies

Provisioned Tolerance

- Precautionary method
- Deploy redundant nodes at network setup time
- Establish a k-connected topology
  - Every node can reach others over at least *k* independent paths

Issues

- Require the deployment of a large number of nodes, high cost
- With node mobility the topology changes and the provisioned solution may be void

Reactive Recovery

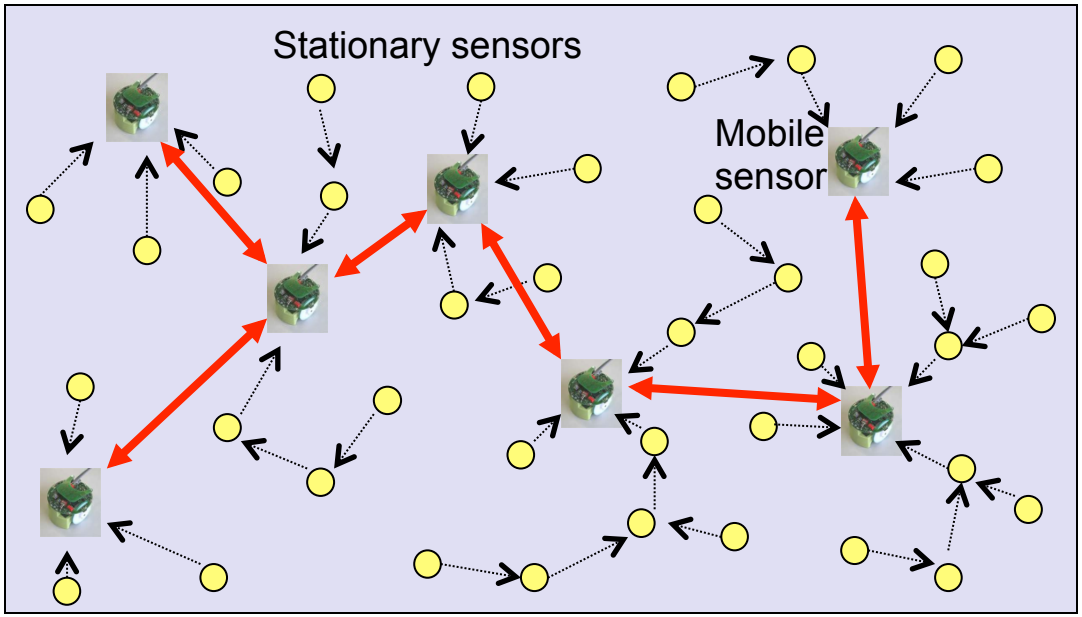
- Real-time connectivity restoration
- Activate only when a failure is detected
- Asynchronous and reactive in nature
- Can be optimized based on the scope of the failure
- May be the only option

Issues

- Compromises may be needed due to the limited resources
- Coordination and recovery overhead is usually high

For inter-base-station (actor) network, the cost of redundant resources makes reactive schemes more appropriate

Reactive Solution Categorization



Environment and Mission

- ❑ Unattended Setup: Border protection, search and rescue, space exploration, etc.
- ❑ Nodes: Mobile battery-operated sensor nodes or a mix of stationary and mobile nodes for which the mobiles form a second tier topology
- ❑ Operation: Ad-hoc networking, collaborative execution of tasks, e.g. search for trapped victims and find an escape path

Metrics

✓*Rapid* restoration of connectivity and normal operation

✓*lightweight self-healing* process

Main Idea

- Successive relocation of some nodes until connectivity is restored

Problem Variants

✓*Centralized (known network state)*

✓*2-hop information (probabilistic determination of cut-vertices)*

✓*Direct neighbors only known*

✓*Preserving pre-failure path lengths*

✓*Avoiding the position of failed node*

✓*Intermittent connectivity restoration*

✓*Spare node designation*

✓*Heterogeneous node capabilities*

DARA: Distributed Actor Recovery Algorithm

Overall Methodology

- Determine the scope of the failure prior to re-acting
  - Real-time restoration, no designated spares in the network
  - Uses probabilistic schemes to detect cut-vertices
  - Requires 2-hop neighbors information
- Restore network by exploiting the mobility of actor nodes

Approach Overview

- An actor (mobile node) from the neighbors of a failed actor will replace it in order to restore connectivity
  - The neighboring actors vote for an actor to move
    - Best Candidate (BC) selection
- Cascaded relocation instead of block movement
  - The connectivity with the children should not be violated
  - Node signals its children so that they follow it in needed in order to keep the connectivity
- DARA is recursively applied to deal with the effect of node's repositioning

Failure Detection & Recovery Plan

Heartbeats & Neighbor List

- Each actor must maintain 2-hop neighbor list
  - Will be used for determining the scope of failure
  - The list also maintains neighbor's node degree, position, ID, etc.
- Neighbors exchange heartbeats to update information & ensure that they are functional
- Missing heartbeat messages are used to detect the failure of actors

Reacting to a node failure

- Recovery may be needed, based on the failed actor position in the network topology
  - Recovery will be initiated only if the Failed actor is a cut-vertex
    ➔ Strong inter-actor connectivity is lost (broken into disjoint partitions)
- The entire detection and recovery process is localized
  - Only the neighbors of a failed actor will detect the failure
  - The scope of recovery will be decided by these neighbors only
  - The restoration process will be initiated on these neighboring

Connectivity Restoration

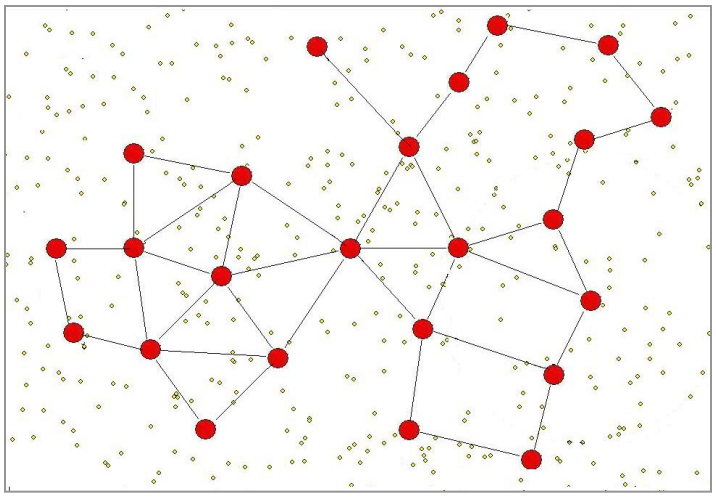
Which neighbor to replace the failed node: *Best Candidate (BC) Selection*

- Only the neighboring actor of a failed actor will participate in the process
  - No inter-actor communication is required
  - Each actor will use its 2-hop neighbor information table
- BC selection criteria are (in order):
  1. Least node degree: The neighbor least node degree limits the scope of the recovery and helps in reducing total distance to be traveled
  2. Least distance to the failed actor: In case of various actors with identical node degrees, least distance favored
  3. Greater actor ID: tie breaker for actors with identical node degrees that are also equidistant to the failed actor

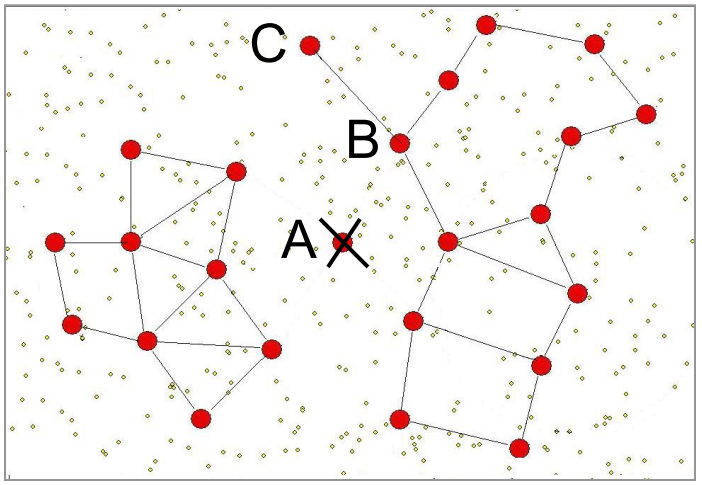
Cascaded Node Relocation

- Before moving: BC will inform its neighbors (children) about its new location
- Upon reaching its new location: BC broadcasts an announcement
- Children of BC will wait until hearing again from the BC:
  - If time-out, children will be assumed disconnected form the network
  - A new cycle of DARA will be started among children

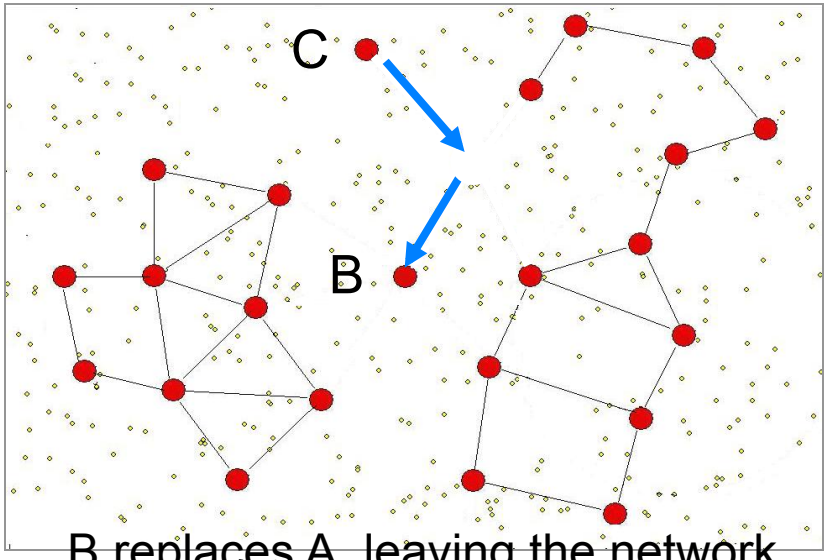
Sample DARA Execution



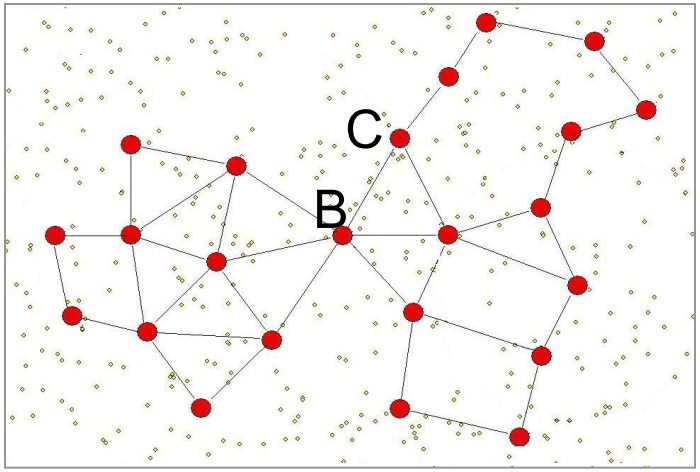
Initially connected network



A's failure partitions the network

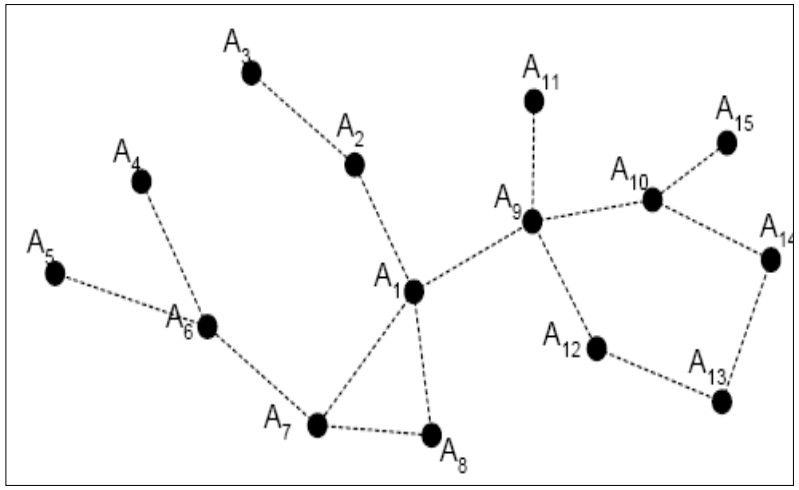


B replaces A, leaving the network partitioned again → apply DARA again

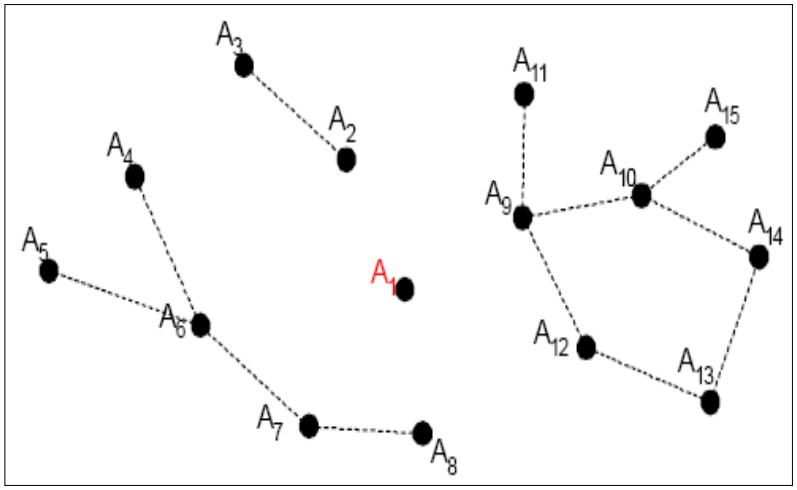


Restored network connectivity

Network Recovery - Example



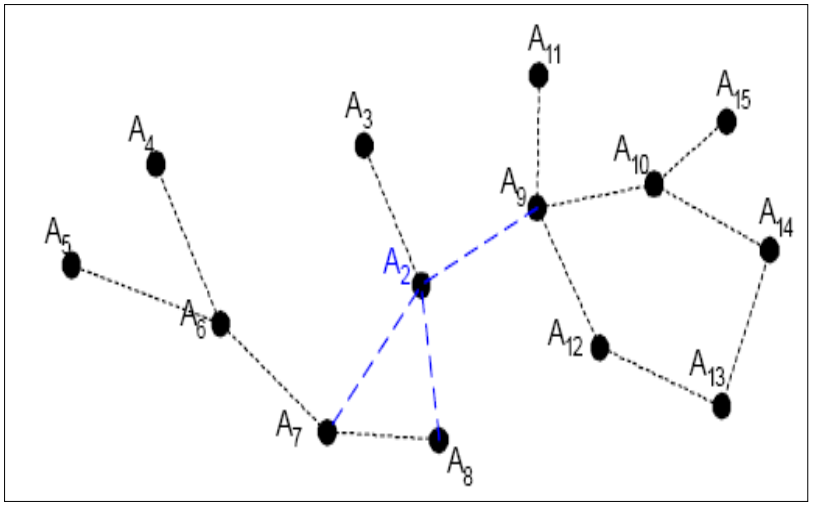
❶

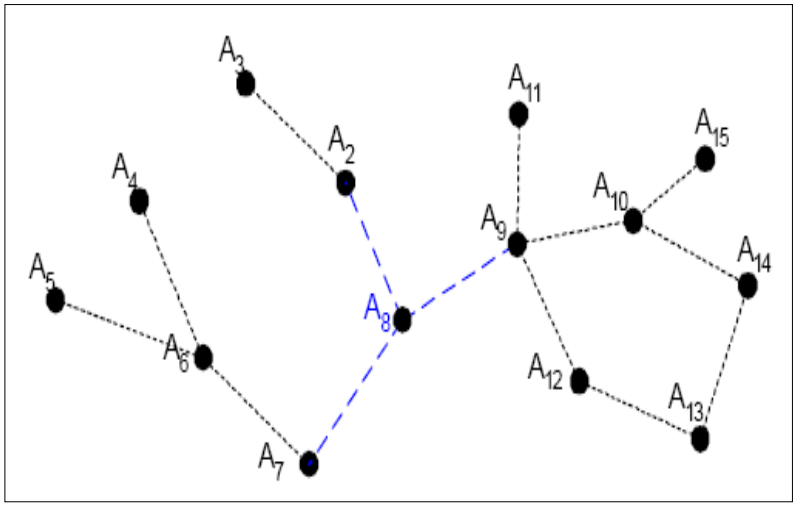


❷

Initial configuration (A_1 a cut-vertex)

A_1 fails causing 3 disjoint partitions





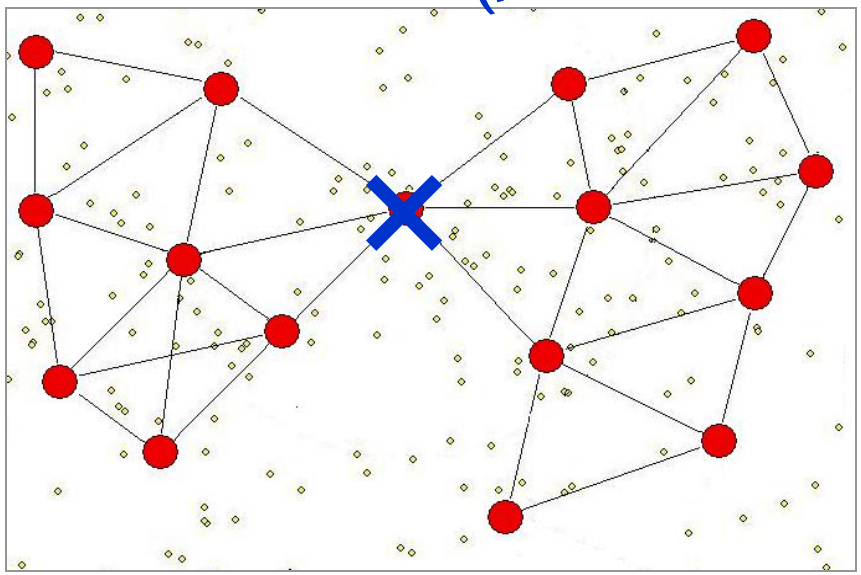
A_2 involves cascaded motion (the ID happened to be luckily favor A_8)

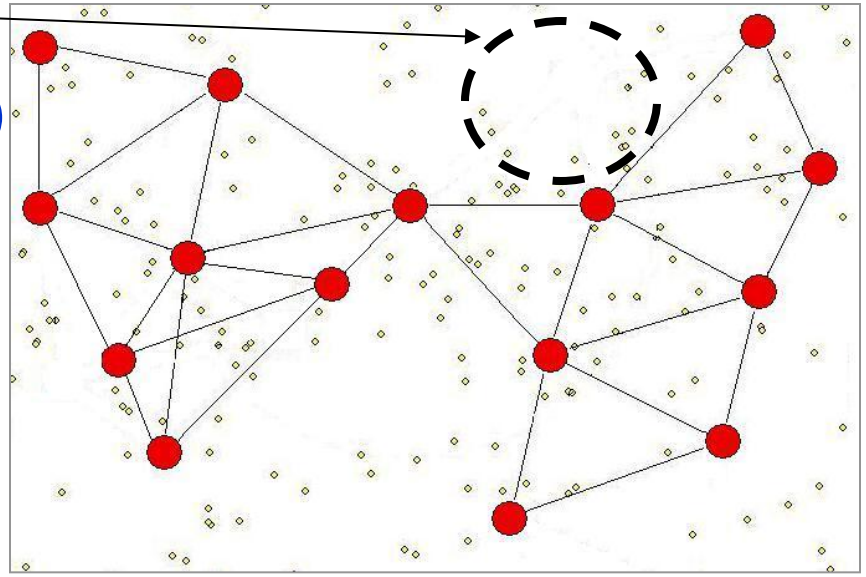
DARA: A_8 replaced the faulty actor (based on node degree and ID)

Comparison to Optimal Solution

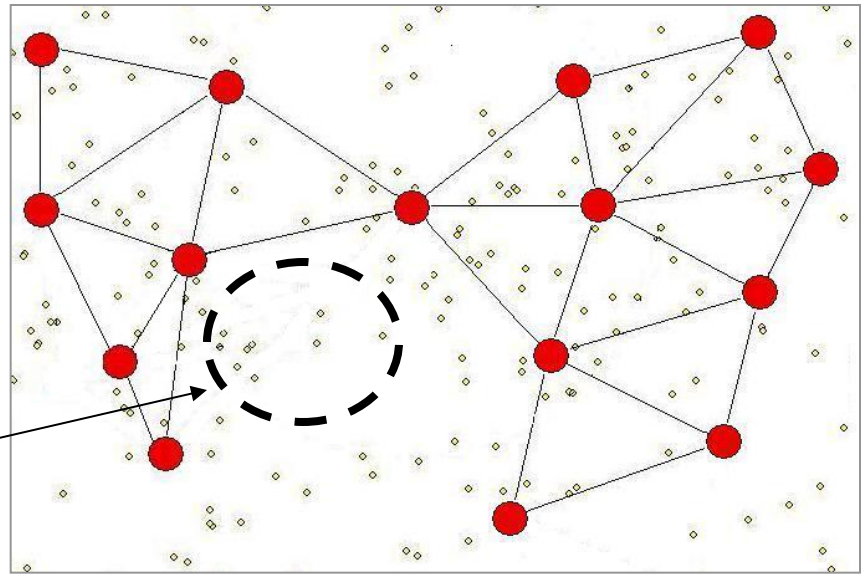
Favors least node degree to limit the scope of the recovery

Final Topology (Recovery through DARA)





Final Connectivity (Optimal recovery solution)

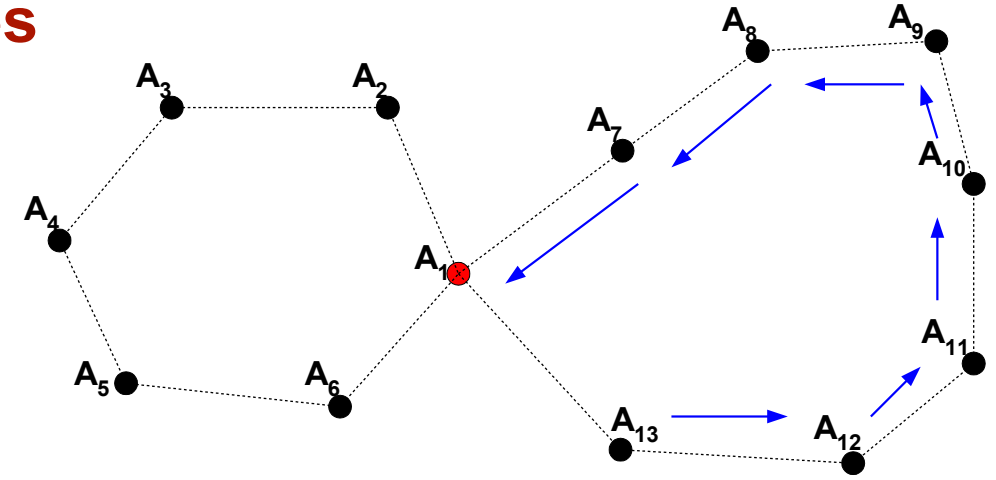


Least travel distance that happens to avoid cascaded relocation

Optimization for Special Topologies

❑ Challenge: cyclic topologies

- Ring like topologies are not uncommon (e.g., Perimeter detection applications)
- DARA may perform poorly:
  - Move all children on the cycle when a failure happens
  - Nodes move like train cars
  - Increases the travel distance
  - In fact this is not needed



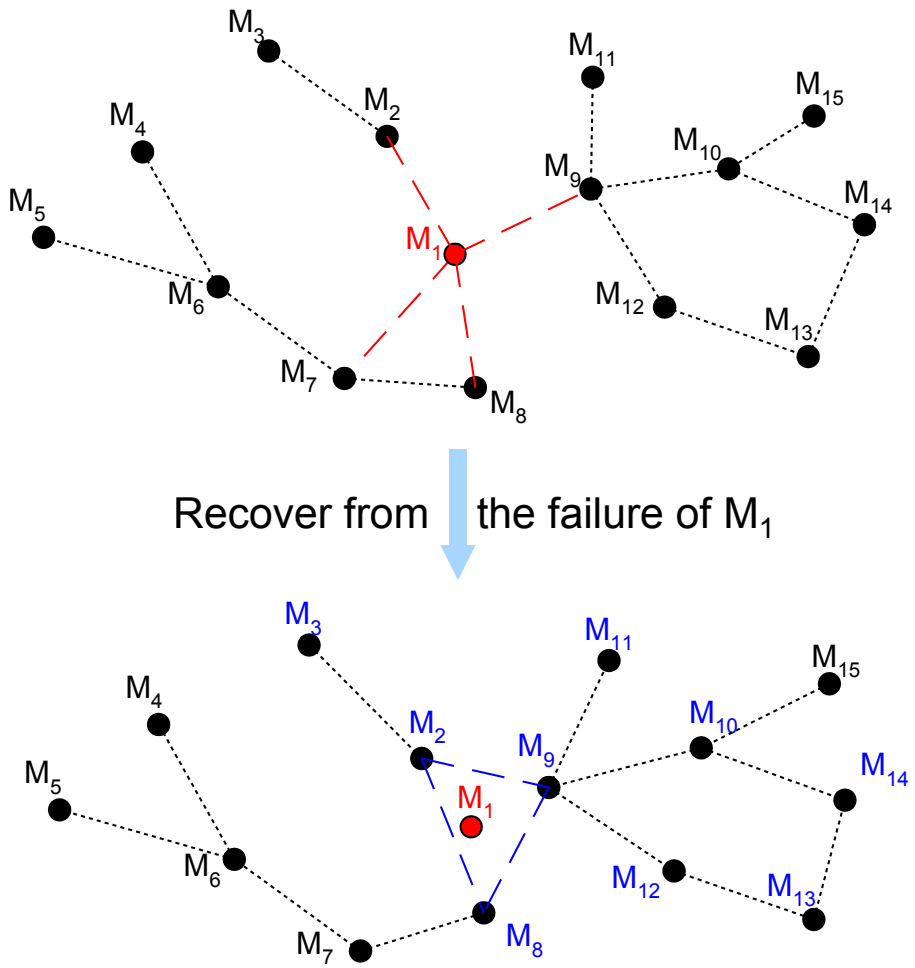
If A_1 fails, nodes A_8, .., A_{13} unnecessarily move

❑ Solution: Exploit Local Flooding (LF)

- Optimization to DARA
  - ✓To reduce the total travel distance
  - ✓At the cost of higher message complexity
- BC performs local flooding to check whether its child is connected or not
  - ✓A message is forwarded in the sub-network
  - ✓A response from one of the siblings is awaited
  - ✓If such a message is received, no further relocation is performed

Example: Localized Self-healing

- *RIM: Recovery through Inward Motion*
  - 1-hop neighbors of a failed node S_f move towards the position of S_f such that they are all reconnected
  - 1-hop neighbors of the neighbors (S_f) follow their parents towards S_f
  - Cascaded relocation repeated until all pre-failure links are restored
- *Network connectivity is fully restored to pre-failure status after recovery*
- *Only one-hop neighbor Information* required for recovery
- *Simplicity and effectiveness*
  - No check for cut vertices
- *Self-healing* recovery process through a *Distributed* algorithm
- *Practical* approach especially in dynamic environments

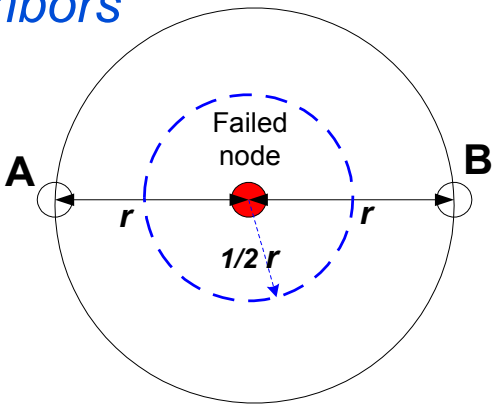


Collective effect as if the network is stretched inward around failed node

Detecting a Failure and Initiating the Recovery Process

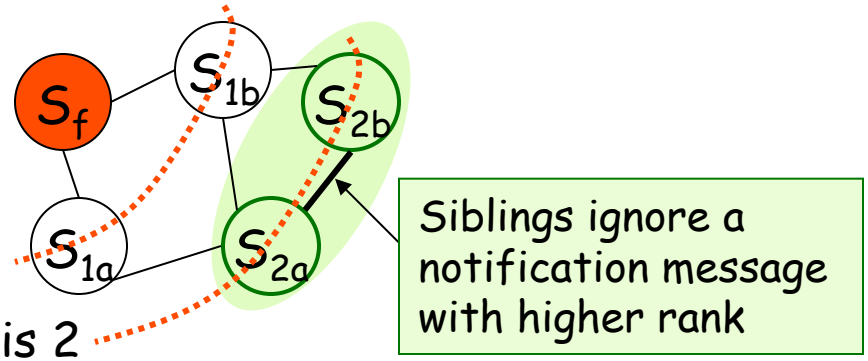
- Neighbor heartbeats are used to ensure that they are functional
- *Missing subsequent heartbeat messages from S_f determines a failure of S_f.*
- Without checking if S_f is a cut-vertex or not
- *1-hop neighbors of S_f, Neighbors(S_f), send a notification message to their neighbors*
- *Neighbors(S_f) move towards S_f until they become r/2 away from the position of S_f.* * *r* is a radio range of a sensor
  - *Maximum distance between the relocated neighbors is "r"*
  - *RIM yields higher connectivity among 1-hop neighbors*

Theorem: *Moving the neighbors of S_f until they become r/2 away from S_f makes them all directly reachable to each other.*



Cascaded Node Relocation

- Let $S_{1r.}$ Be one of the 1-hop neighbors of S_f
- *Neighbors(S_{1r}) participate in the restoration process* when they receive a *notification message*, *{Node_ID, New position, Rank=1}* from S_{1r}
  - Rank denotes the minimum number of hops from S_f



- Neighbors(S_{1r}) assess whether or not the connection to S_{1r} will be lost after S_{1r} moves to its new position
- *If a loss of the connection to S_{1r} is detected, Neighbors(S_{1r}) also send a notification message to its children (S_{2i}'s) and move to the **new location***, which is ultimately towards S_f
  - *The **new location** of S_{2i} that is a neighbor of S_{1r} is computed based on the new positions of S_{1r} and other nodes that have sent a notification message to S_{2i} with the lowest rank (i.e., rank =1).*

Repositioning Nodes with Rank > 1

A new location of neighbors with rank more than 1 is computed based on:

(a) If C lost only 1 connection to B

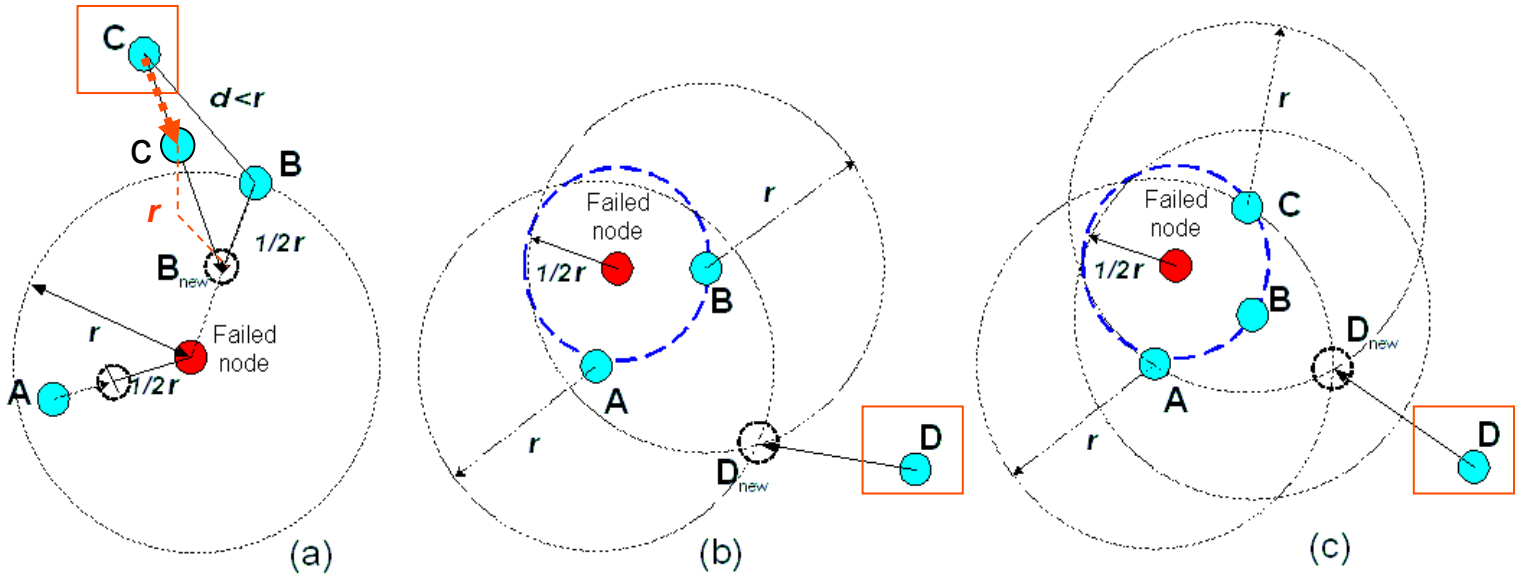
➔ C moves towards the new position of B *until it is reconnected to B*

(b) If D lost 2 connections to A and B

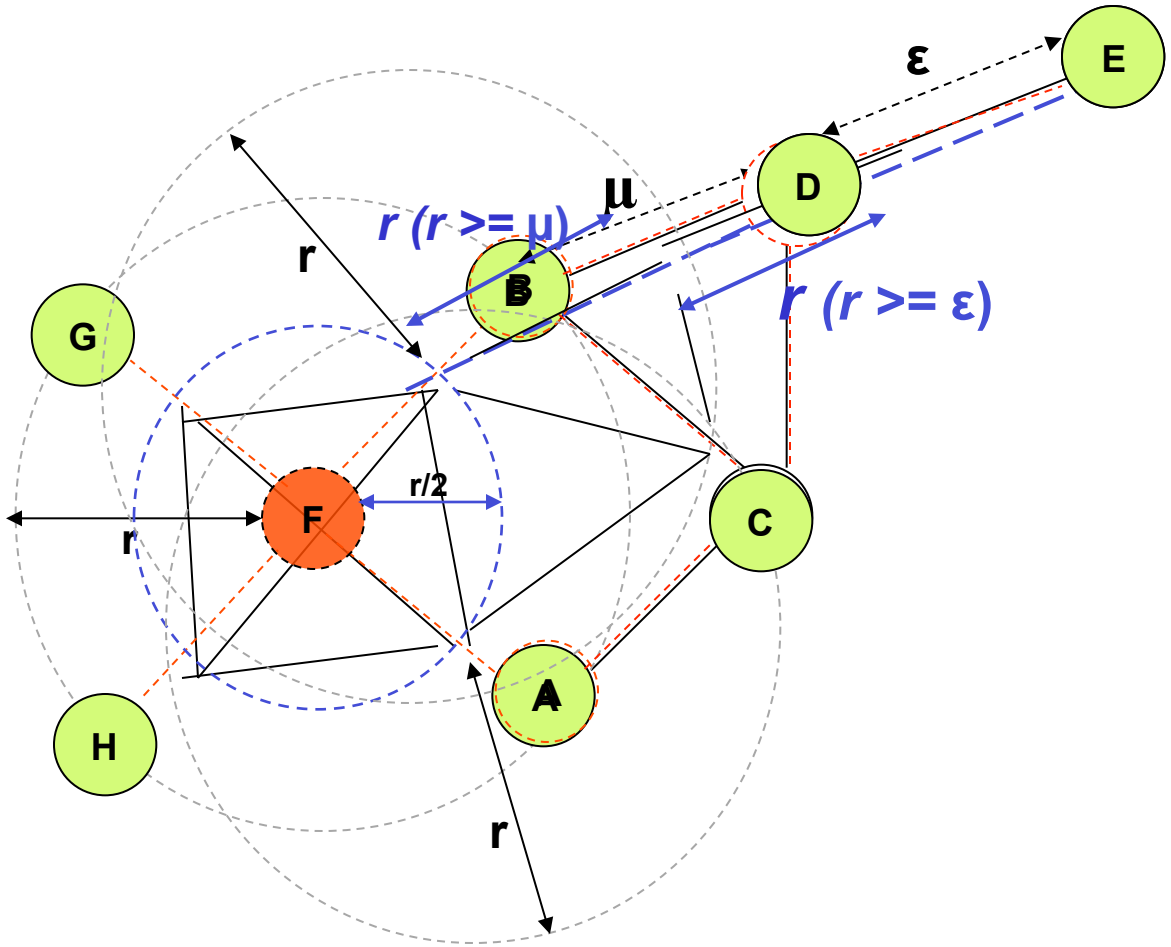
➔ D moves to *the closest point that lies within the range "r" of A and B*

(c) If D lost 3 connections to A, B, and C

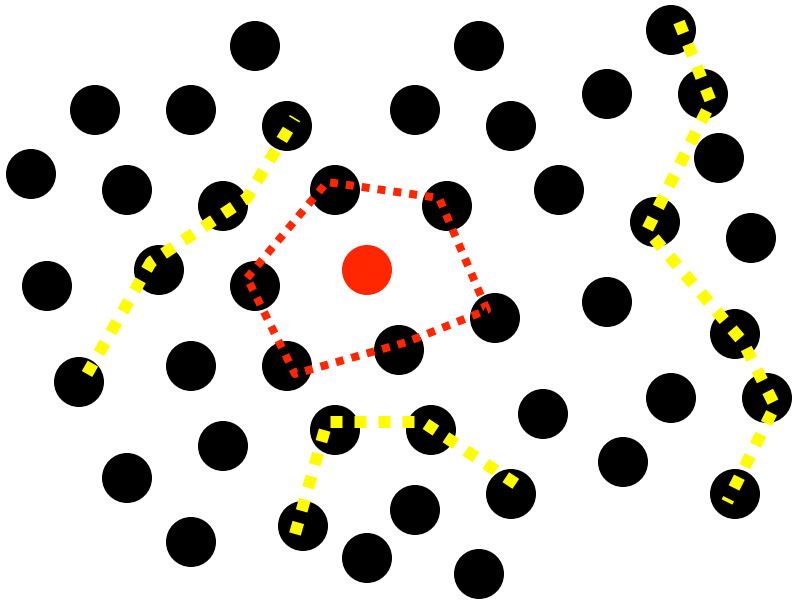
➔ D moves to *the closest intersection point that lies within the third (non-intersecting) circle*



Illustrative Example



Single vs. Multiple Node Failure

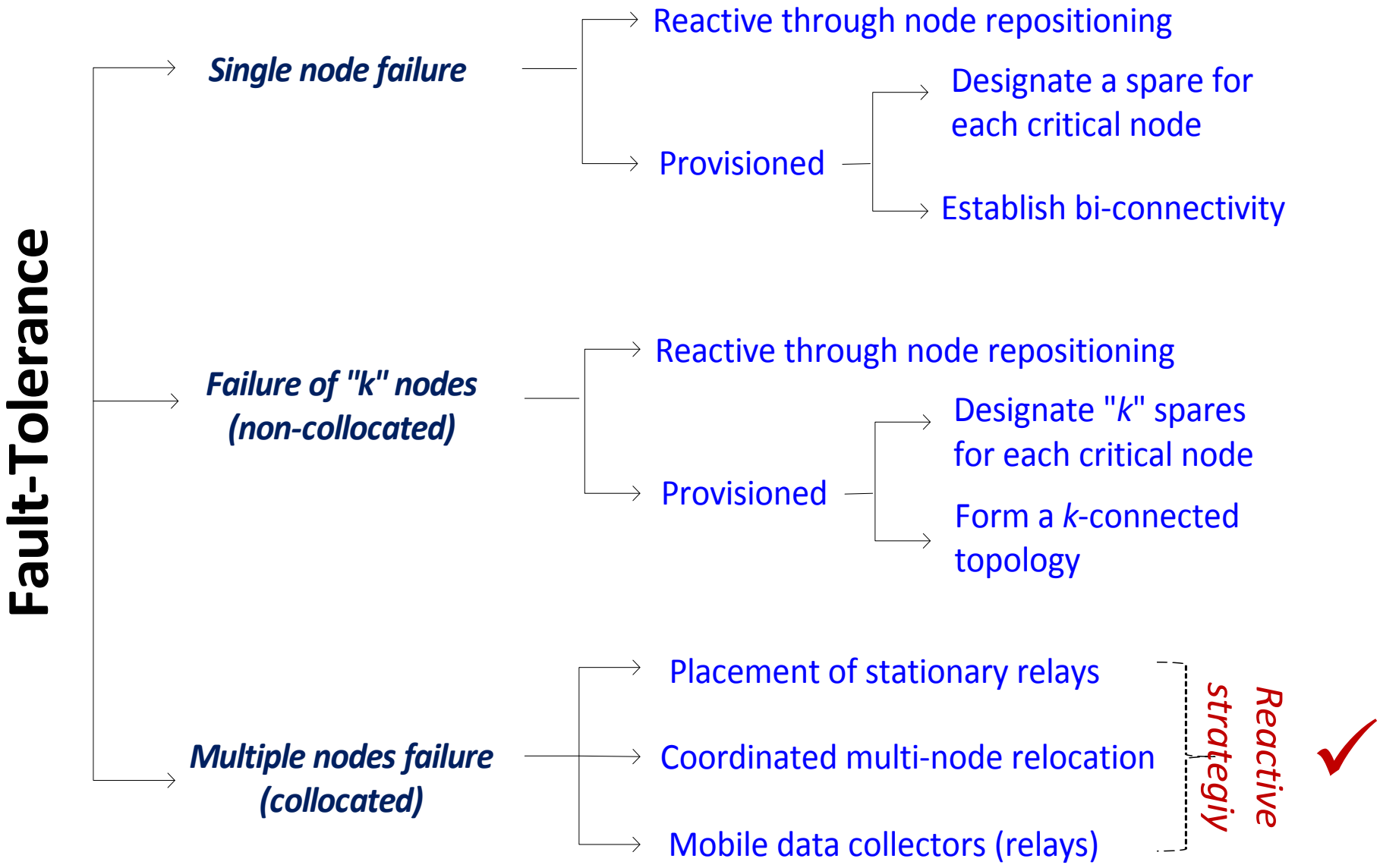


Federating the disjoint segment is a significantly harder problem:

- How to determine the scope
- The border nodes do not know the scale of the damage
- The network may be able to self-heal only if sufficient resources are available

- *A single node failure* indicates that only one node fails at a time and can be simply detected using 1 or 2-hop heartbeat message
- In harsh environments such as *border protection, search-and-rescue and battlefield reconnaissance*, nodes are susceptible to damage which causes the *Wireless Sensor Network (WSN) to get partitioned into disjoint segments*

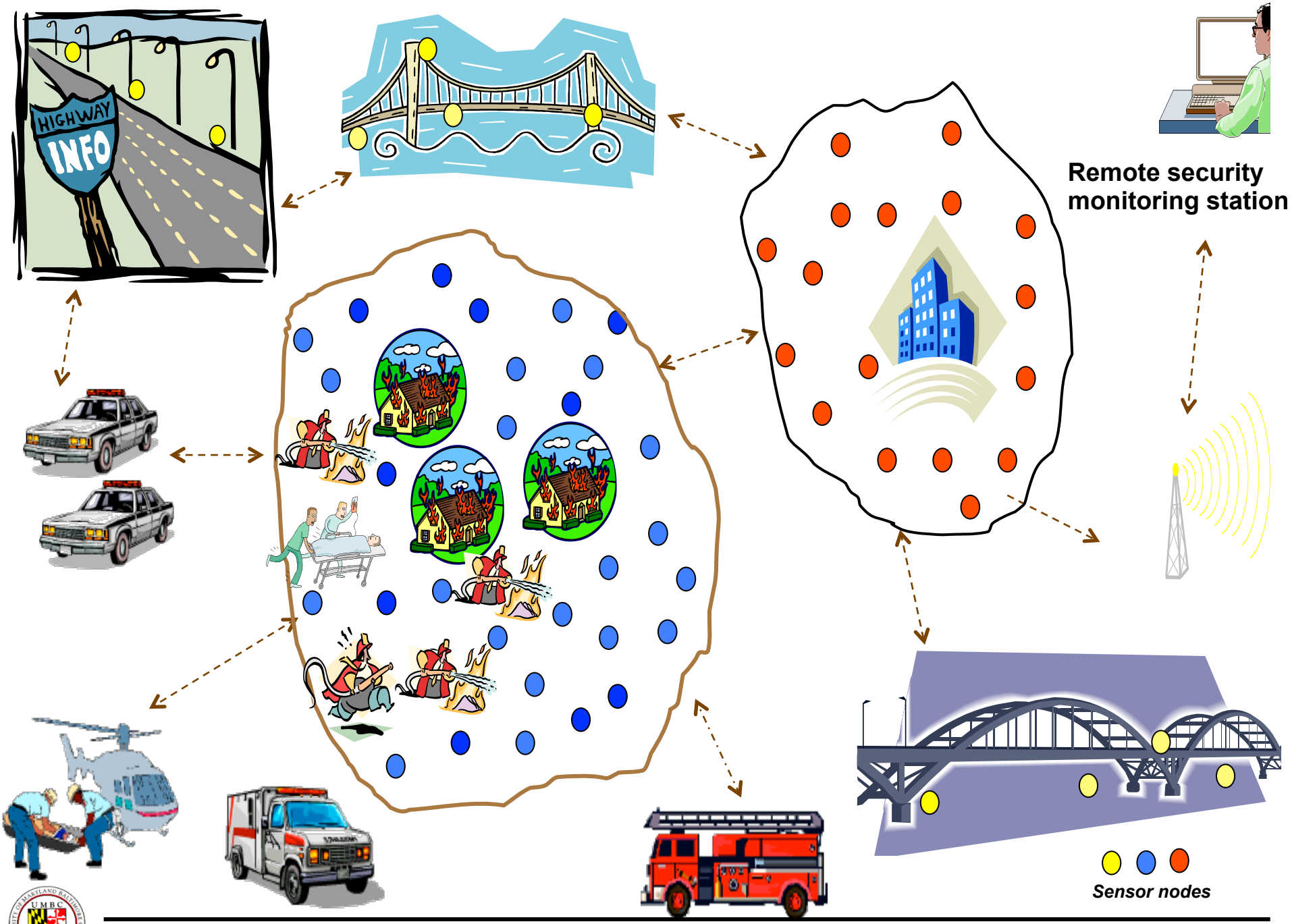
Taxonomy Federation Strategies



Other Federation Scenarios

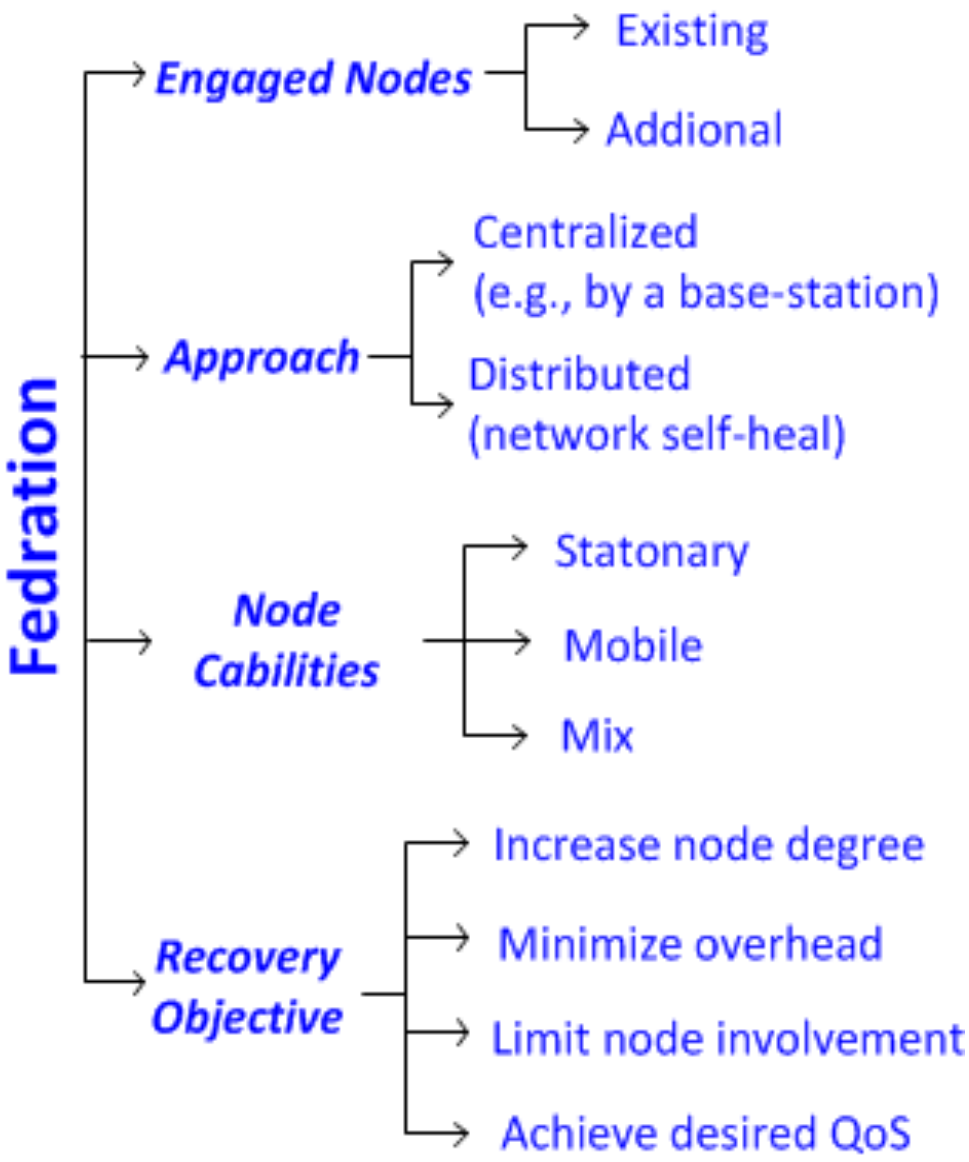
Many application scenarios would need or benefit from interconnecting multiple networks in order to better serve a particular mission:

- Multiple surveillance networks can be leveraged during a hunt for a criminal.
  - Interconnecting these networks will enable law-enforcement officers to do real-time tracking and localization of wanted individuals and expedite the arrest
- When data is to be shared among multiple autonomous wireless sensor networks to deal with unforeseen event such as major disaster
  - Enable overall situational awareness on and away from the scene
  - Allow authorities to allocate resources efficiently and contain the impact
- It may be sometimes necessary to connect offshore oil exploration station and/or underwater sensor network to an inland command center in order to
  - assess human risk when a major problem such as the eruption of a fire takes place.
  - provide real-time technical support and assist with onside repair


HIGHWAY
INFO
Remote security monitoring station
Sensor nodes

Categorizing Federation Methodologies

- Distributed approaches require the availability of mobile nodes
- Segments need to have sufficient count of mobile nodes in order to self-manage the recovery process
- Mobility can take the form of having robots that carry the relays
- Most approaches like to minimize the engagement of nodes
  - To avoiding re-tasking existing nodes and further degrade the network service
  - To cut on the externally-provided resources
- Data delivery latency may impose additional requirements on the formed inter-segment topology and the federation methodology

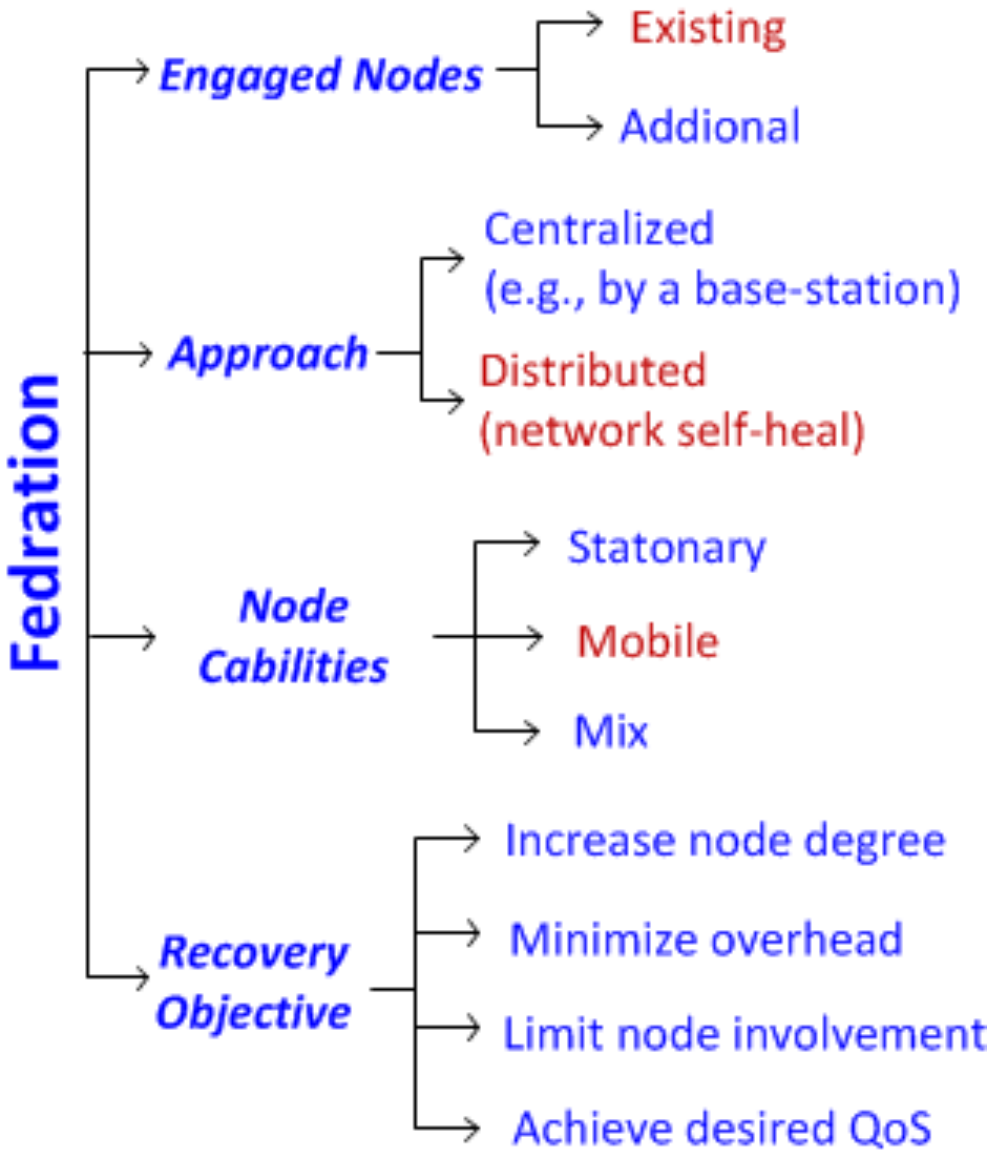


Sample Federation Approach

Two examples:

1. Connected dominating set based relocation (
   - ➢ limited to 2 segments)

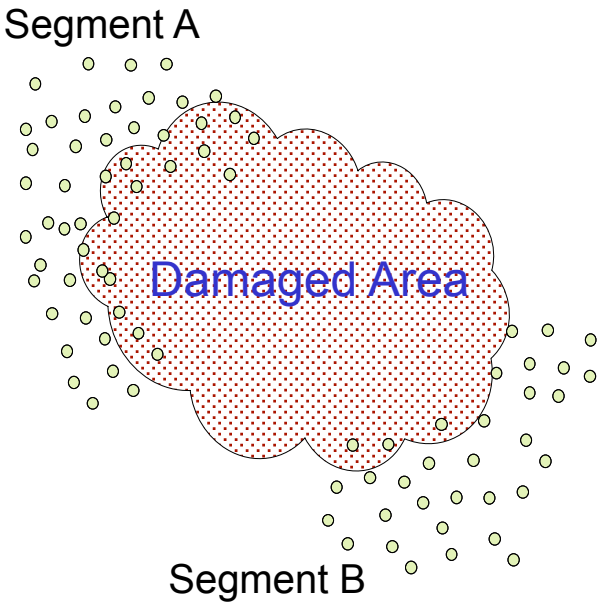
2. AUR- Autonomous Recovery
   - ➢ Any number of segments



Connecting Two Segments

Application Scenarios

- When multiple nodes fail resulting into ONLY two partitions that are far apart
- Two independent networks are to be combined to collaboratively perform a task, e.g. federating two boarder monitoring sensor networks to help in a rescue mission in a remote village that got hit by earthquake



Assumptions

- All nodes are mobile and know their locations

Objective

- Connect these sub-networks collaboratively in a distributed manner
- Minimize the overhead measured as the to total movement distance of all engaged noes

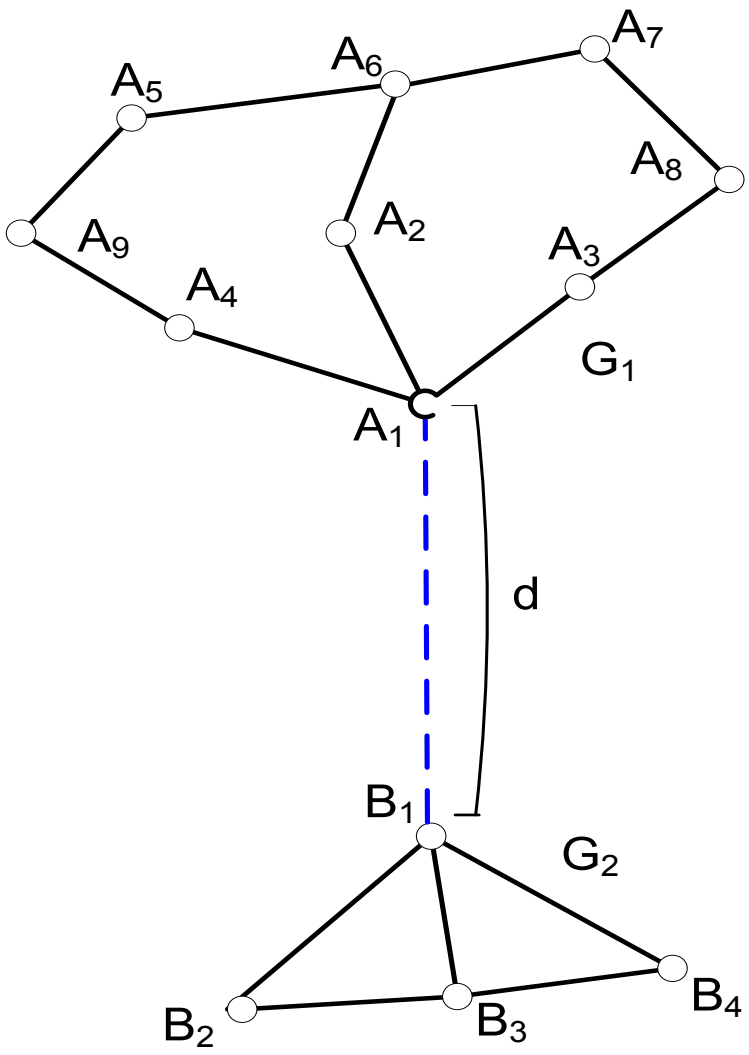
Starts with the case of 2 partitions case and present a generalization

Solution

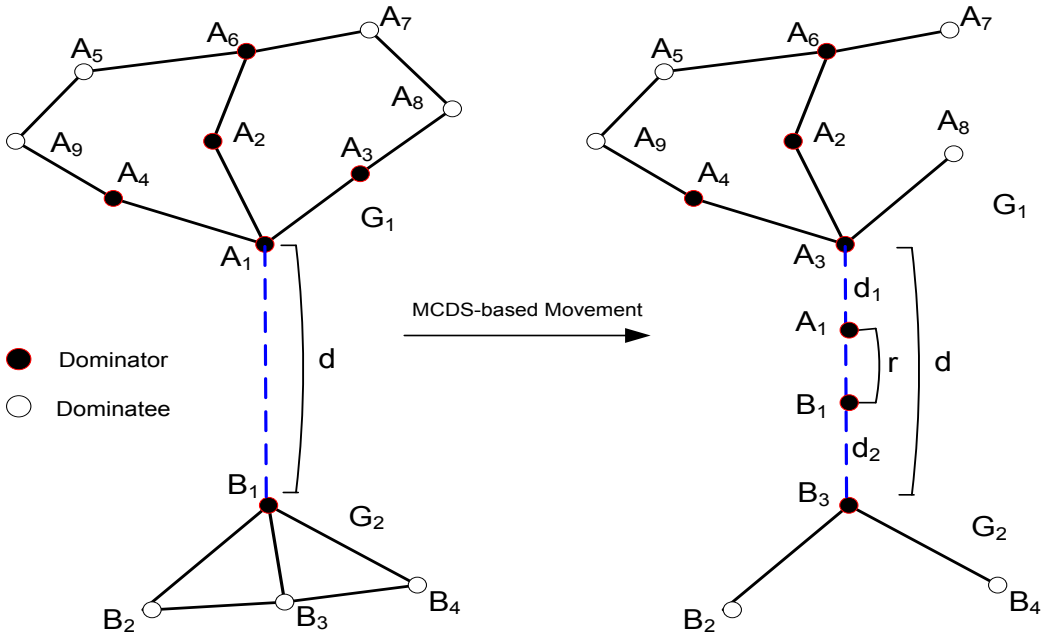
- Relocation of closest node until establishing connectivity
- Cascaded repositioning of nodes
- Effect resembles stretching the partitions

Establishing Connectivity

- **Assume G_1 and G_2 and their closest nodes A_1 and B_1**
- **The idea is to move A_1 and B_1 towards each other until they get into range of each other**
- **Appropriate nodes are picked to fill in the space between closest nodes A_1 and B_1 and corresponding sub-networks**
- **Goal: To share the movement load between G_1 and G_2**
  - Guarantee a max movement distance of $M_i = (d-r)/2$ for any node i
  - Minimize the total movement

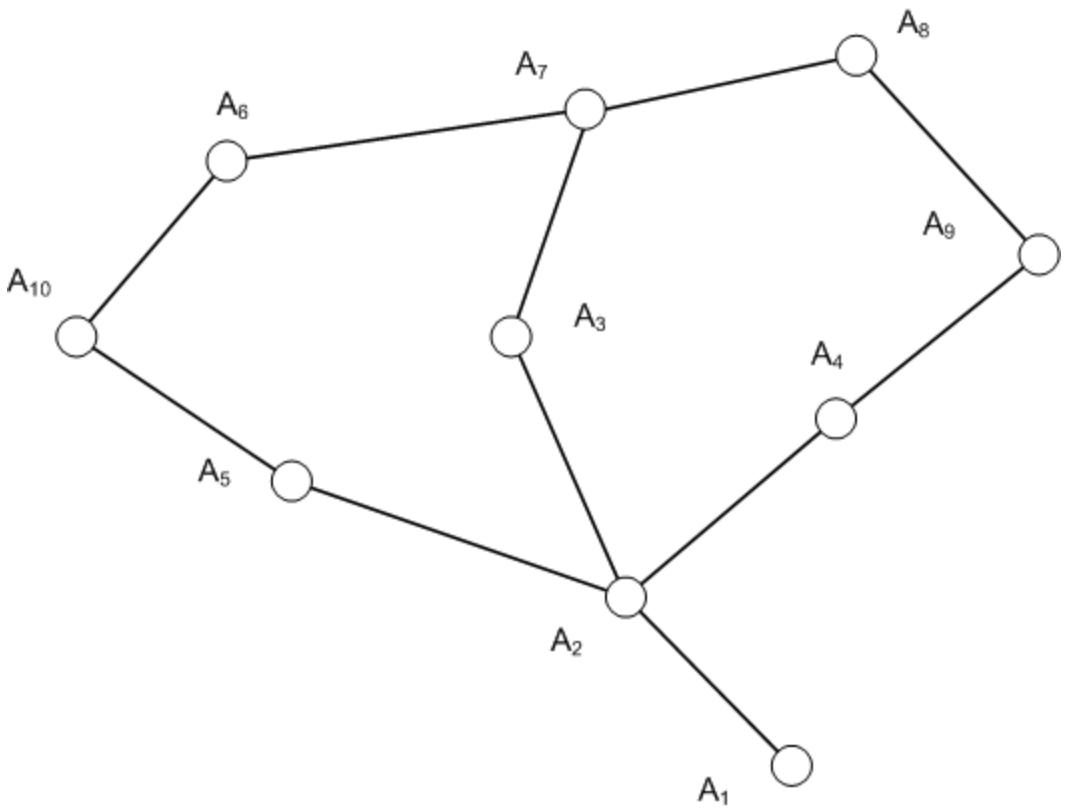


MCDS-based Movement

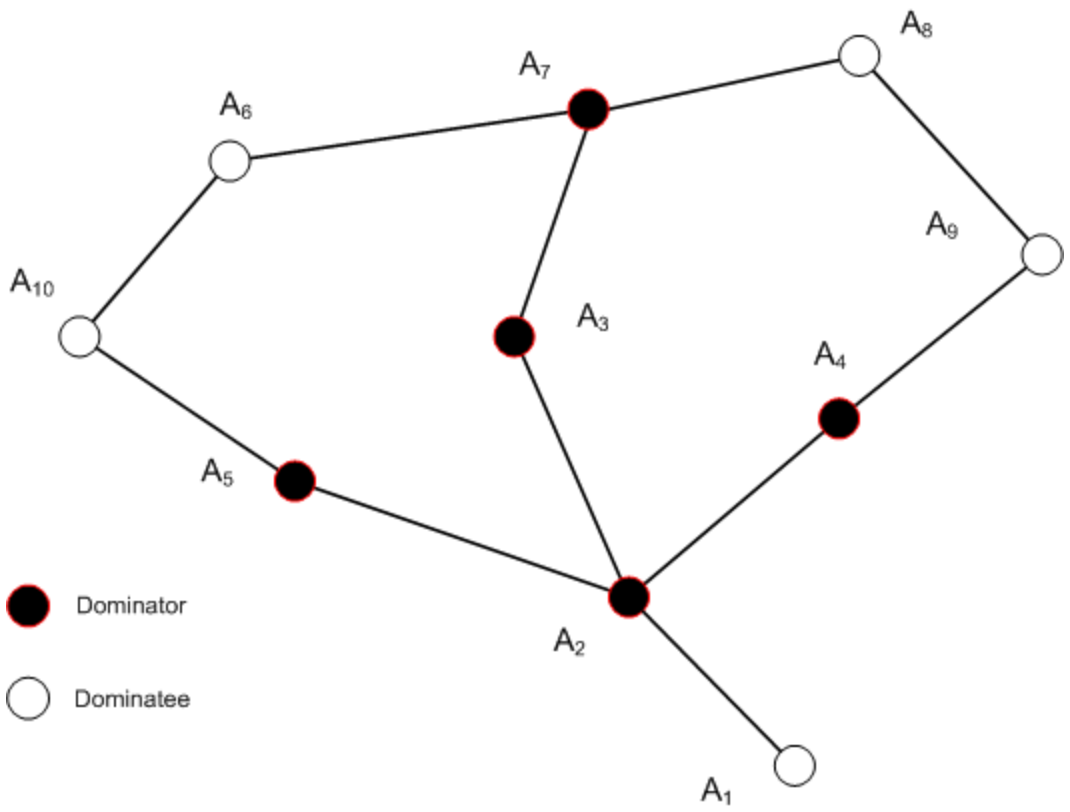


- Determine Minimum Connected Dominating Set (MCDS) of each partition
  - A node is either a dominator or dominatee
  - When A_1 and B_1 moves, they are replaced with a dominatee node
- The closest dominatee is picked
  - This may trigger cascaded movements in order to maintain connectivity of each sub-network
  - Step-by-step movement

MCDS-based Movement Example

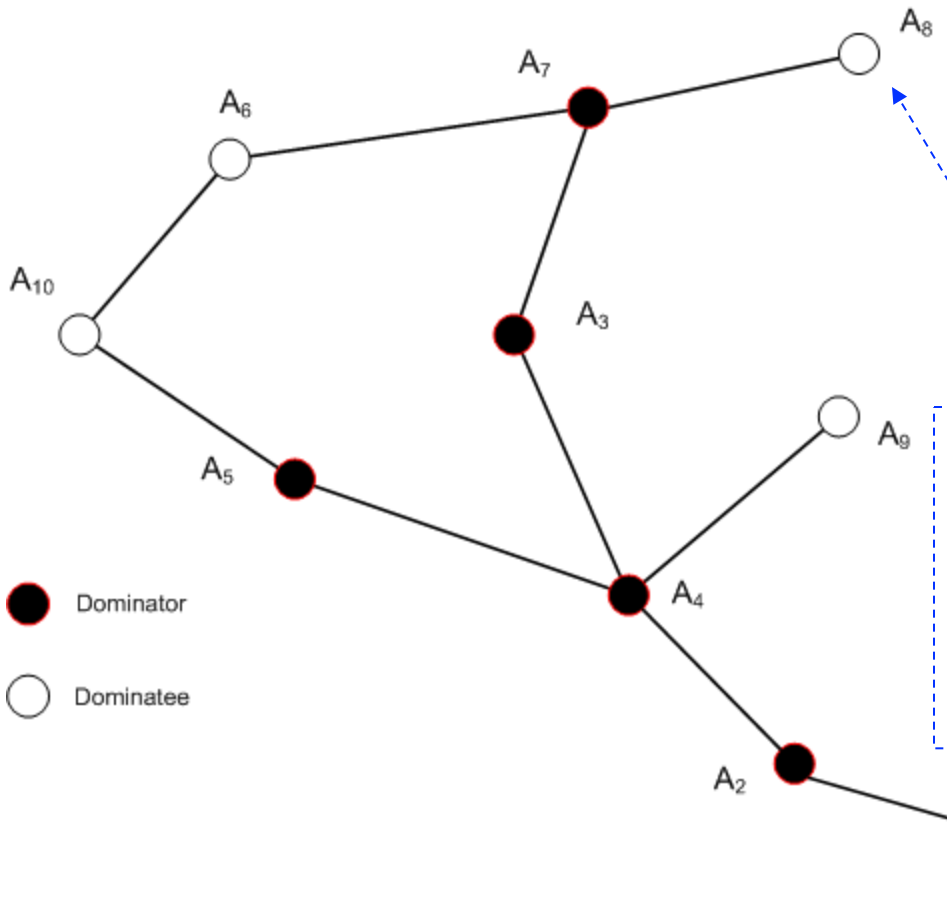


MCDS-based Movement Example



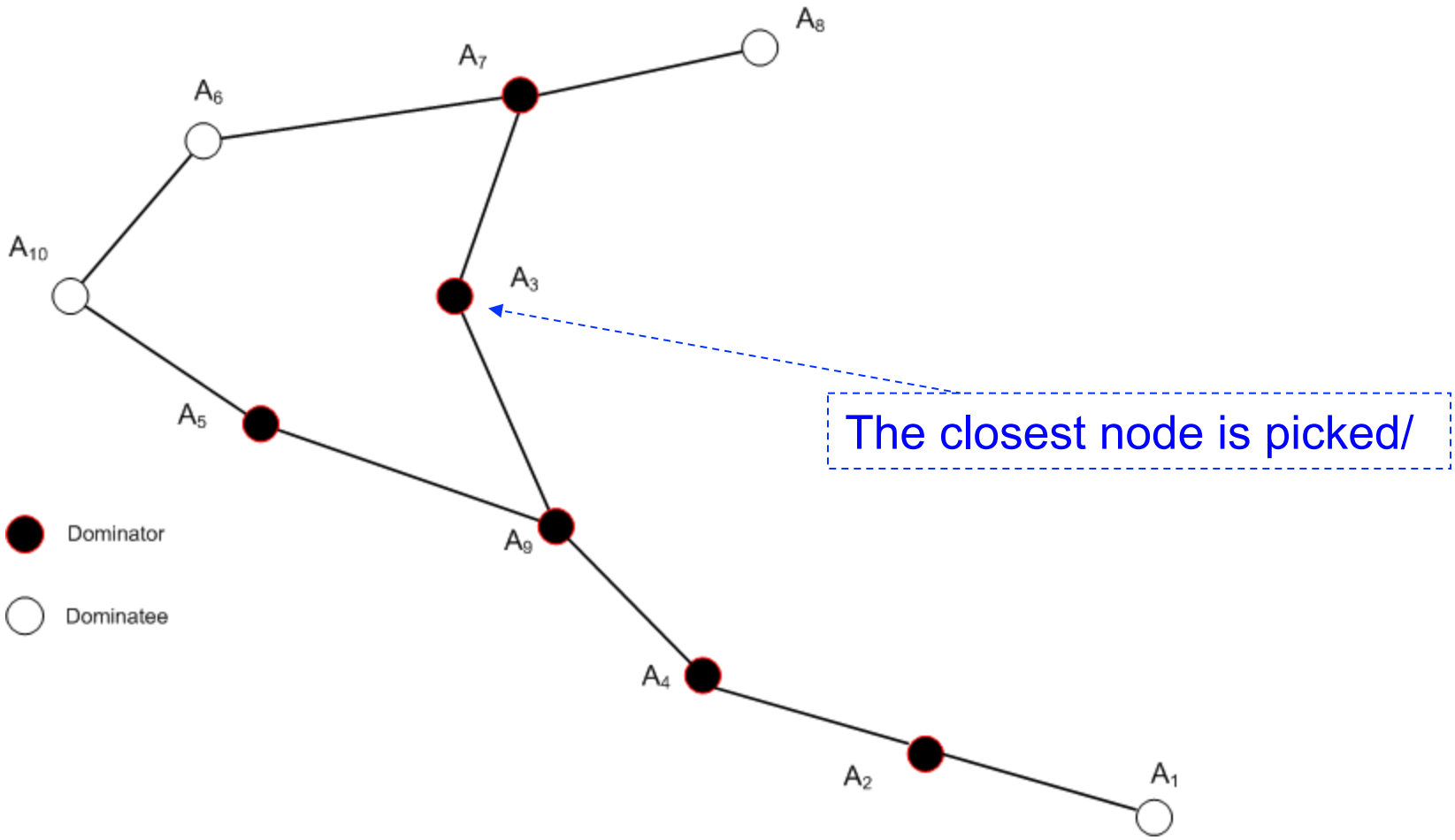
As long as the MCDS is maintained for a network, the effect of relocating a node will be limited to itself and would not risk the connectivity of the other nodes in the network

MCDS-based Movement Example

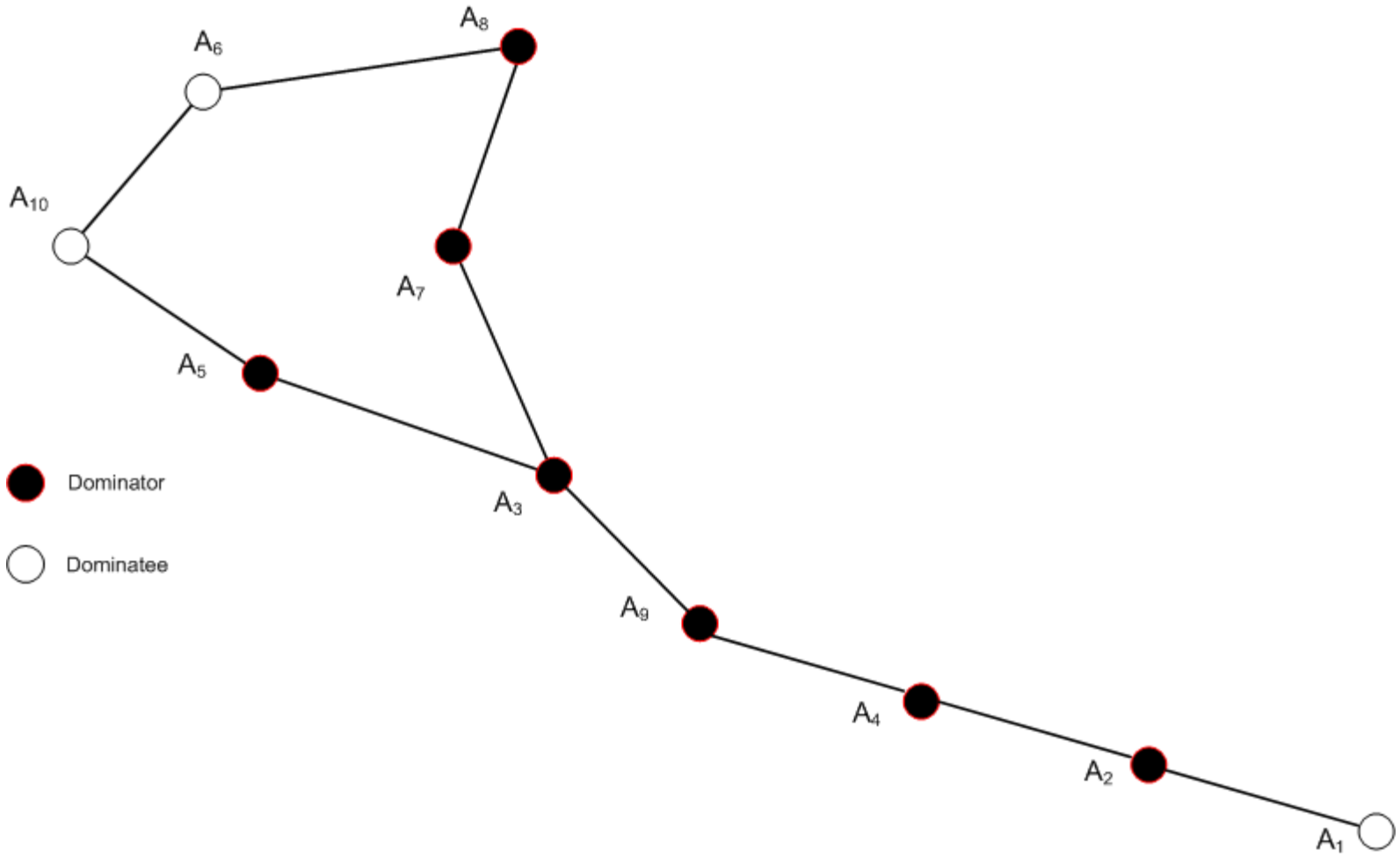


Neighbors are all dominatees and they are already connected to the network through other dominators
➔ There is no need to replace the moved node to maintain connectivity

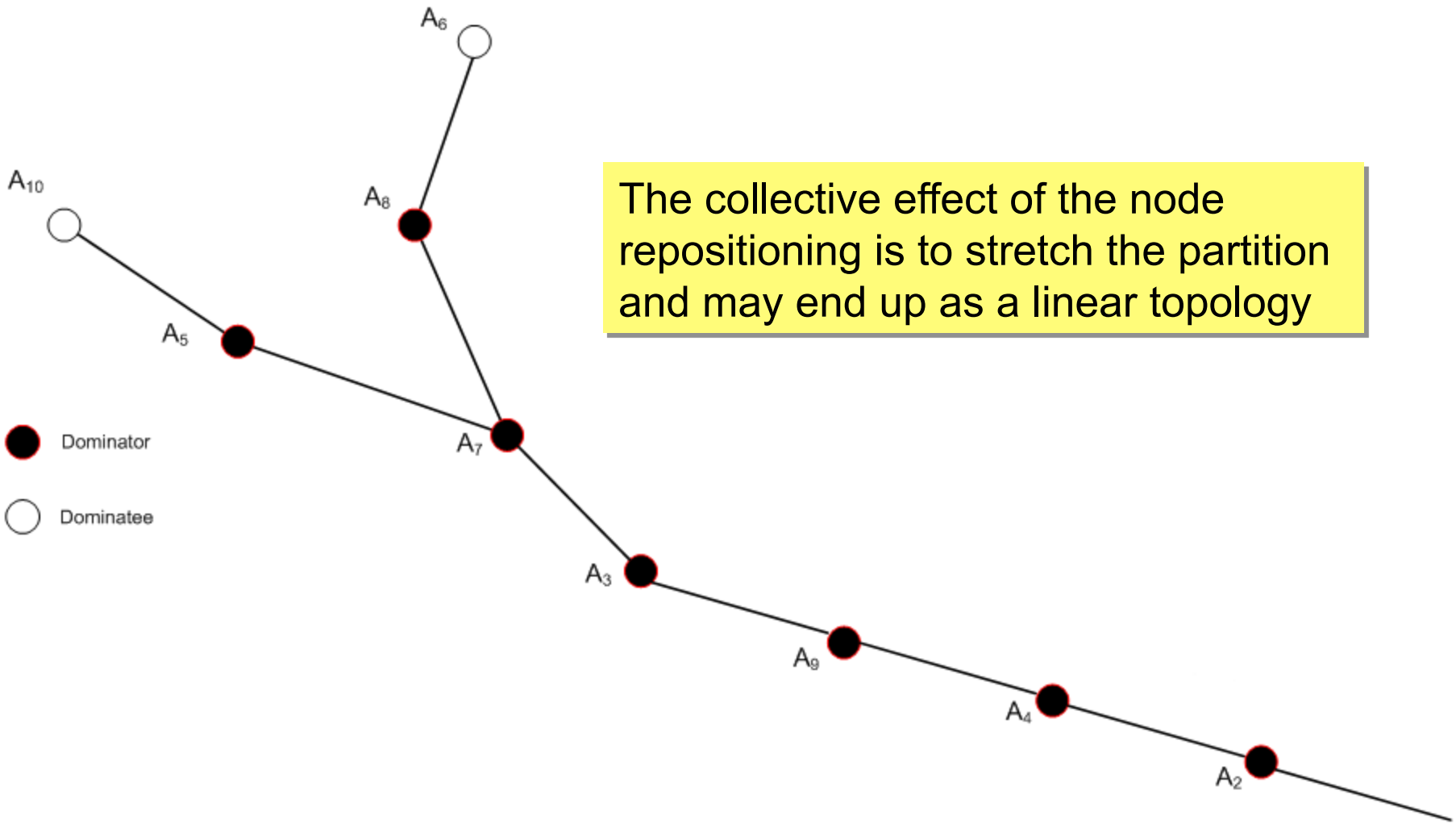
MCDS-based Movement Example



MCDS-based Movement Example



MCDS-based Movement Example



The collective effect of the node repositioning is to stretch the partition and may end up as a linear topology

AUR- Autonomous Recovery from Multi-Node Failures in WSN's

Aim: To re-establish lost connectivity while restoring lost coverage through self-spreading.

Features:

- Truly distributed approach.
- Reduced Complexity and easily scalable.
- Nodes interact with only their 1-hop neighbors to initiate recovery.
- Inspired by nature. Movements based on modified Coulombs' law of Electrostatic Attraction & Repulsion.

Phase II: Initial Self Spreading

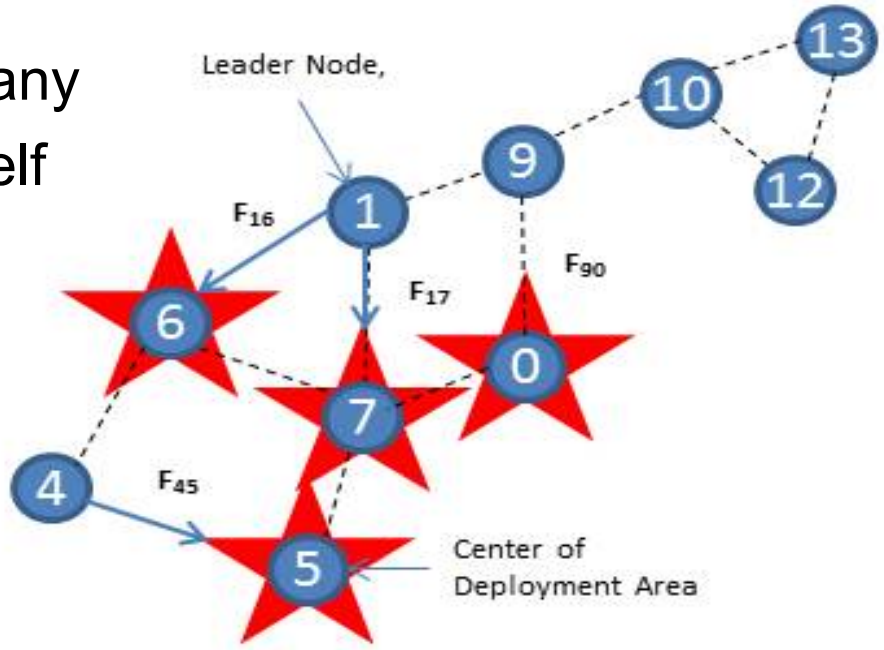
Aim is to spread in the direction of loss.

- Maximizes chance of connecting to other nodes.
- Recover some of lost coverage area.

Relocation Order based on:

- Lost Node Degree (P): Nodes with most lost neighbors move first.
- Current Node Degree (β): Drag many neighbors behind them increase self spreading
- Node ID: Used as tiebreaker.
  Node with lower ID wins tie.

Leader Node: has highest lost node priority in its 1-hop neighborhood and spearheads the relocation process.



AuR Motion - Force Equations

- Based on Coulomb's law of Inter-molecular interaction.
- Each node experiences forces only due to its 1-hop neighbors.
- Leader nodes experience only a force of attraction due to their lost neighbors:

$$\mathrm{F}af=r;$$

- Force of repulsion between live nodes "S_a" and "S_b" :

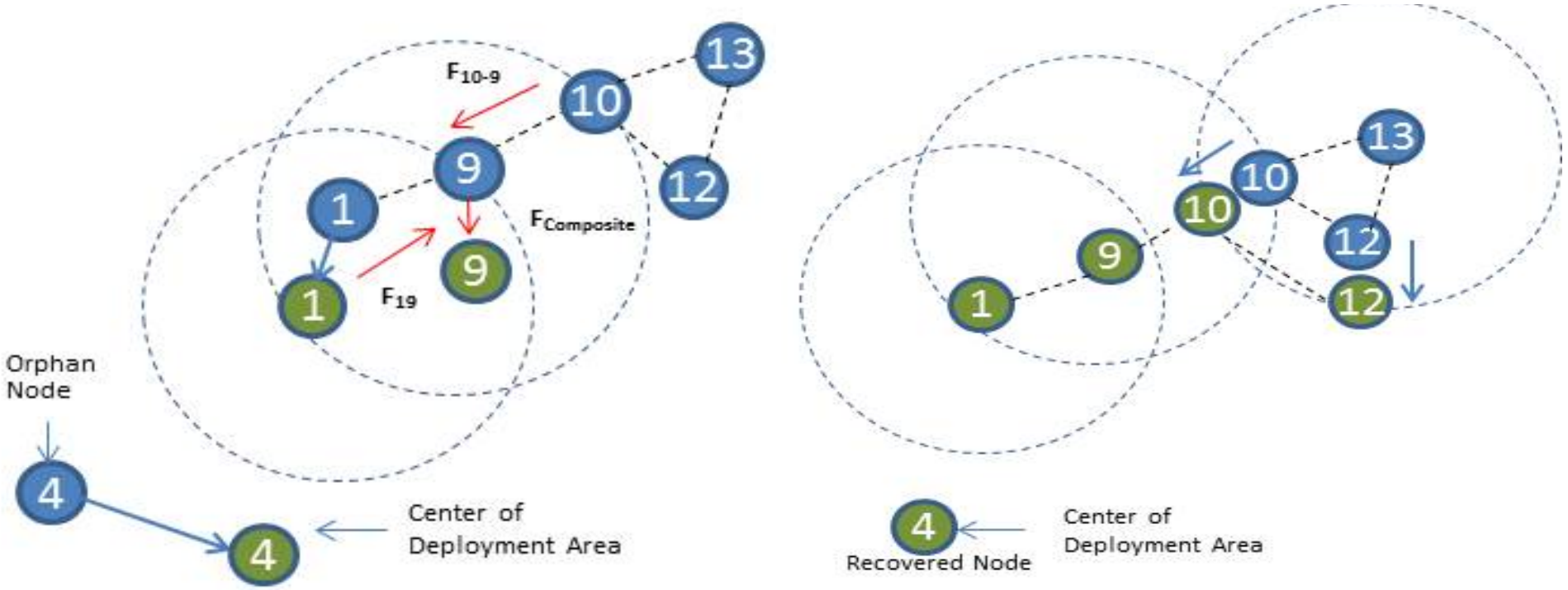
$$\mathrm{F}ab=\{ \blacksquare\blacksquare r-d\downarrow ab \ , \quad if\ r>d\downarrow ab\ 0 \quad , \quad if\ r\leq d\downarrow ab$$

All non-leader nodes move via repulsion.

'r' is communication range, dab is the distance between nodes.

Composite force acting on a node is

AuR Motion - Relocation



- Nodes spread out in direction of loss led by their respective leader nodes.
- Always maintain connectivity with all 1-hop neighbors.
- Nodes inform neighbors when relocating about their new destination.
- Nodes spread until the forces acting on them are balanced, i.e. Cannot spread any further. If no movement possible Center Force=SET.

Phase II: Motion Towards the Center

- Move disjoint spread-out partitions towards center of deployment area to ensure final convergence.

Relocation Order:

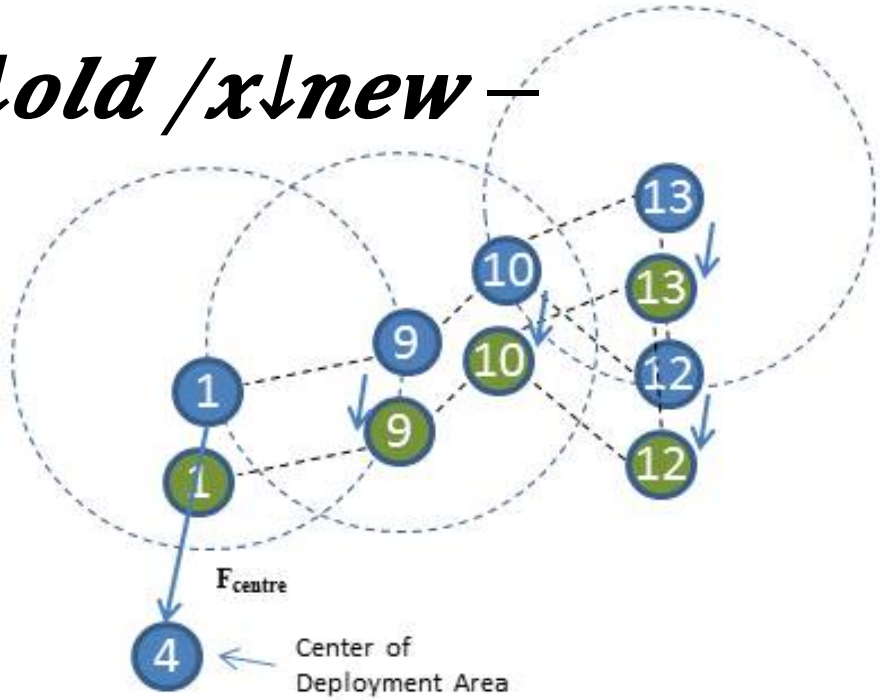
- Based on distance from center of deployment area - closest nodes to center have highest priority and become leader nodes
- Each node decides its own priority based on its 1-hop neighbor information.
- Leader Node moves r/2 in direction of center and informs its 1-hop neighbors of its new location.
- Waits for all follower nodes to come back in range before relocating again
- Motion continues until the leader node reaches the center or any node of the 1-hop network comes into contact with a center connected partition
- Segments that get connected applies AuR as a single block

Motion Towards the Center (Non-leader nodes)

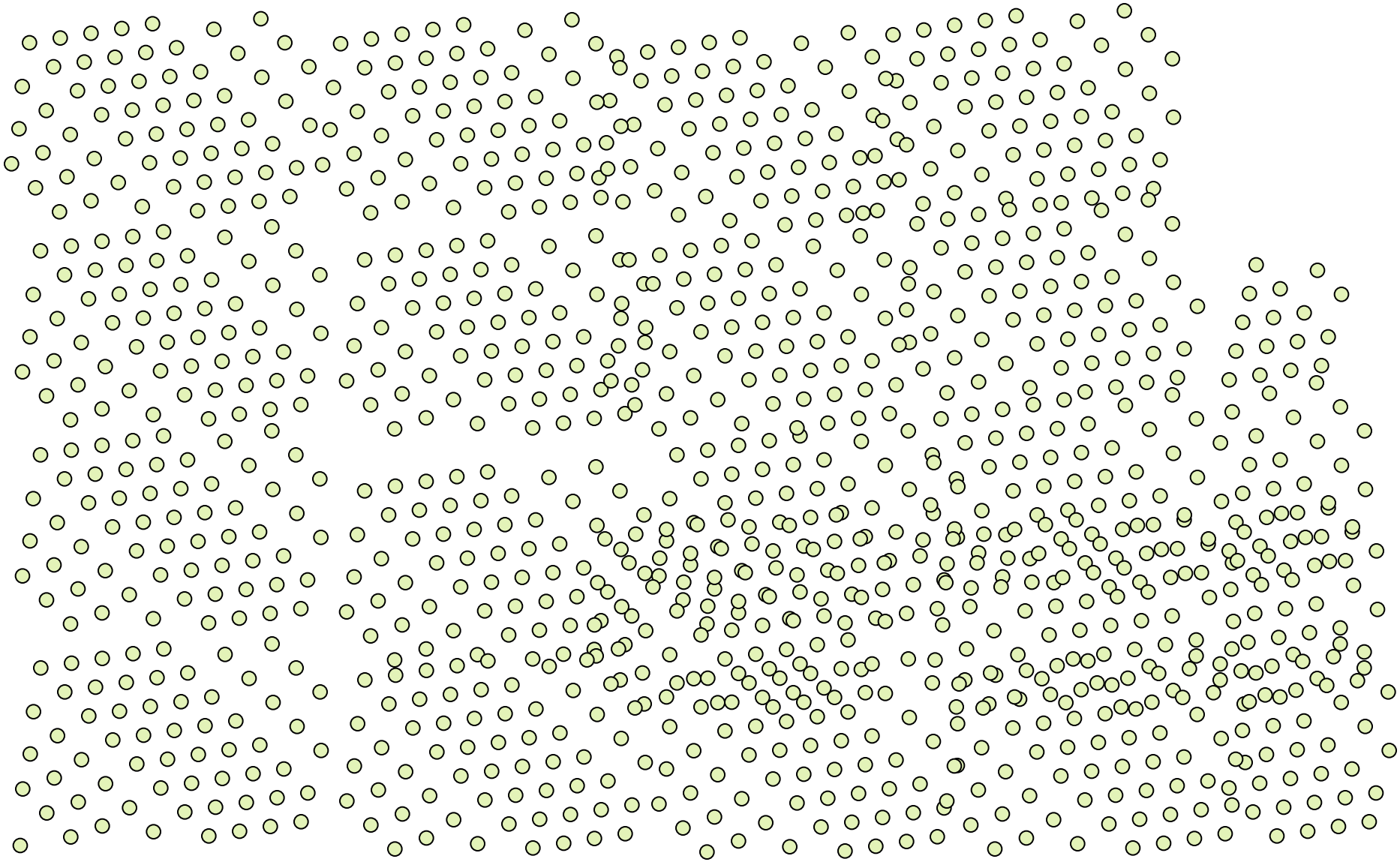
- Non Leader nodes wait until they have received messages from all nodes with higher priority before moving.
- Direction of motion based on composite slope of line to all higher- priority nodes.

$$Slope = y_{new} - y_{old} / x_{new} - x_{old}$$

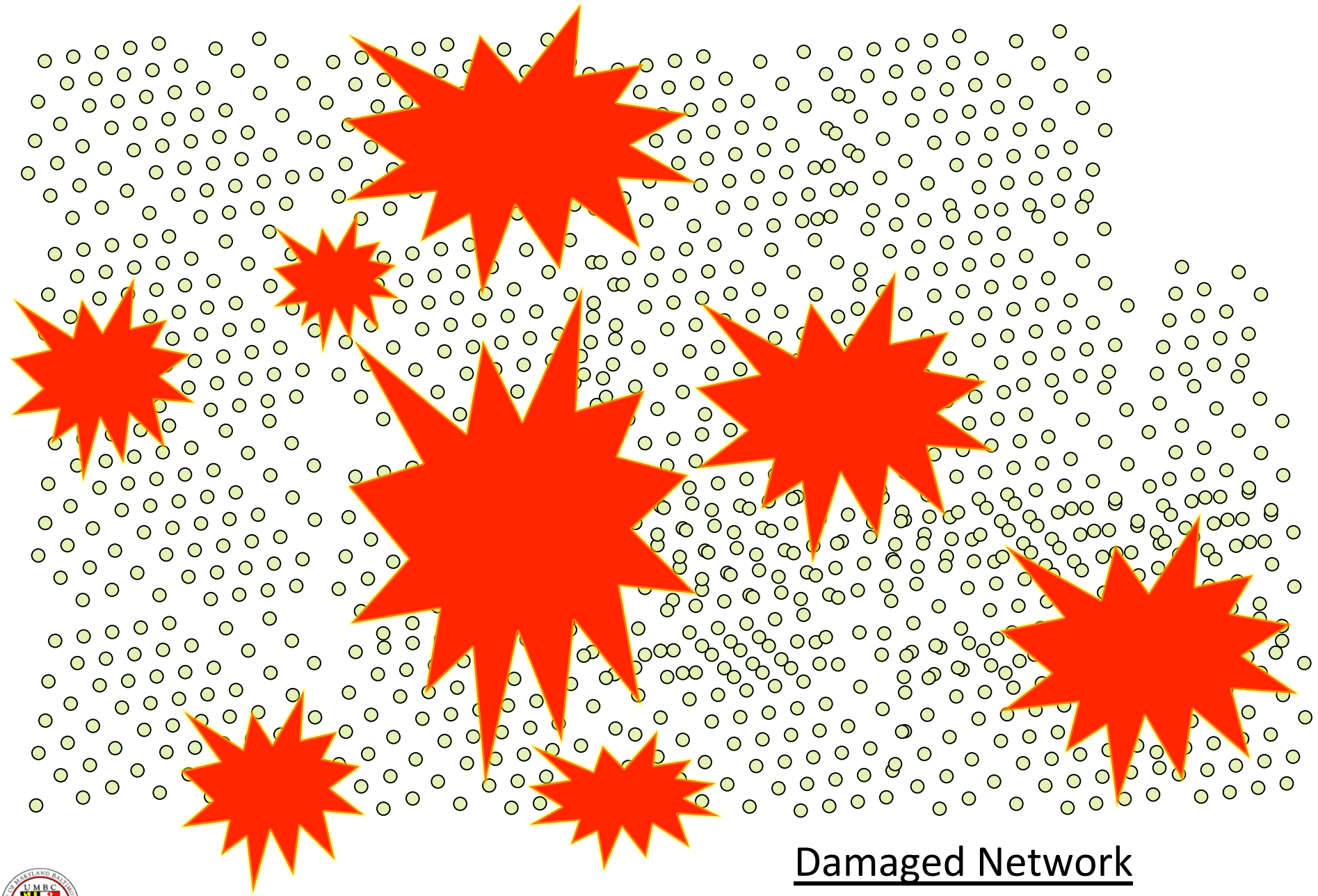
- Move in the calculated direction until they are in communication range with all higher priority neighbors



AuR: Autonomous Recovery

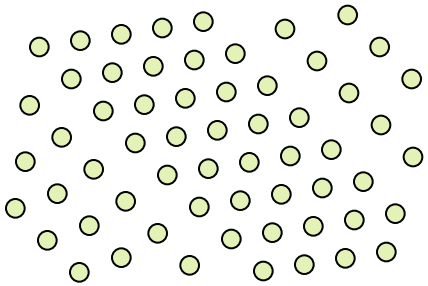


Initial State of WSN

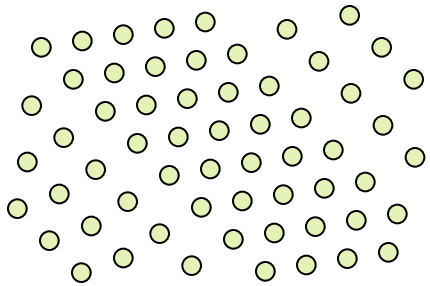


Damaged Network

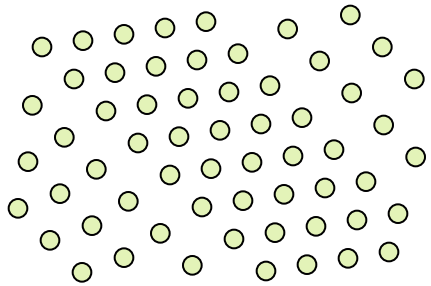
Partition #1

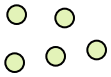


Partition #4



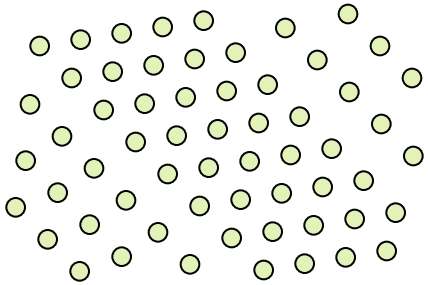
Partition #5

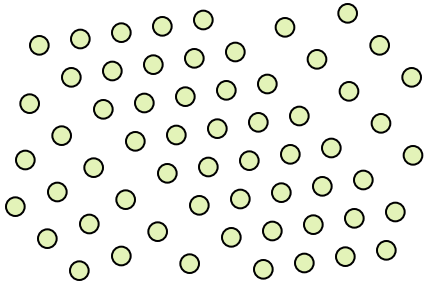




Partition #2

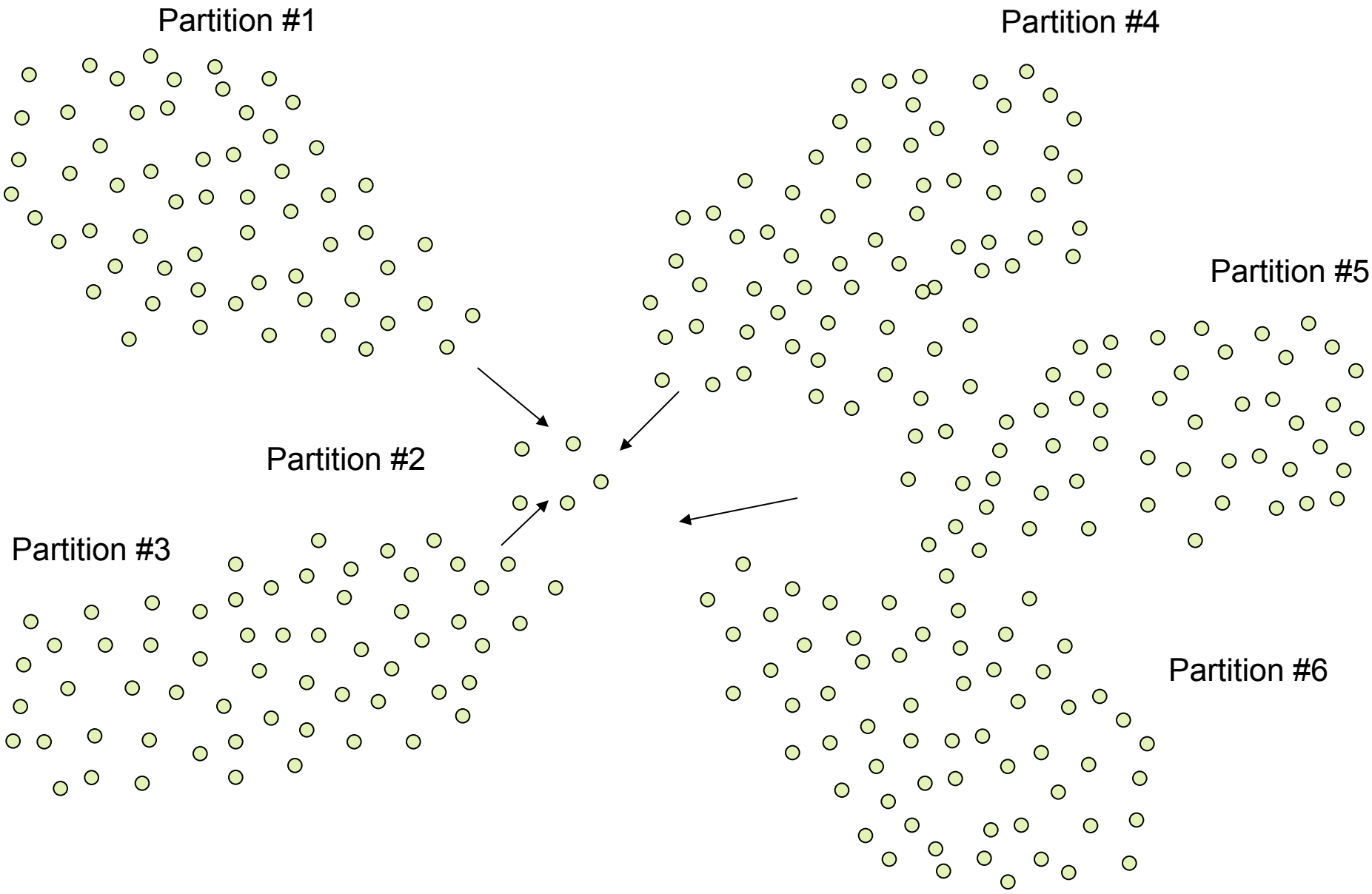
Partition #3



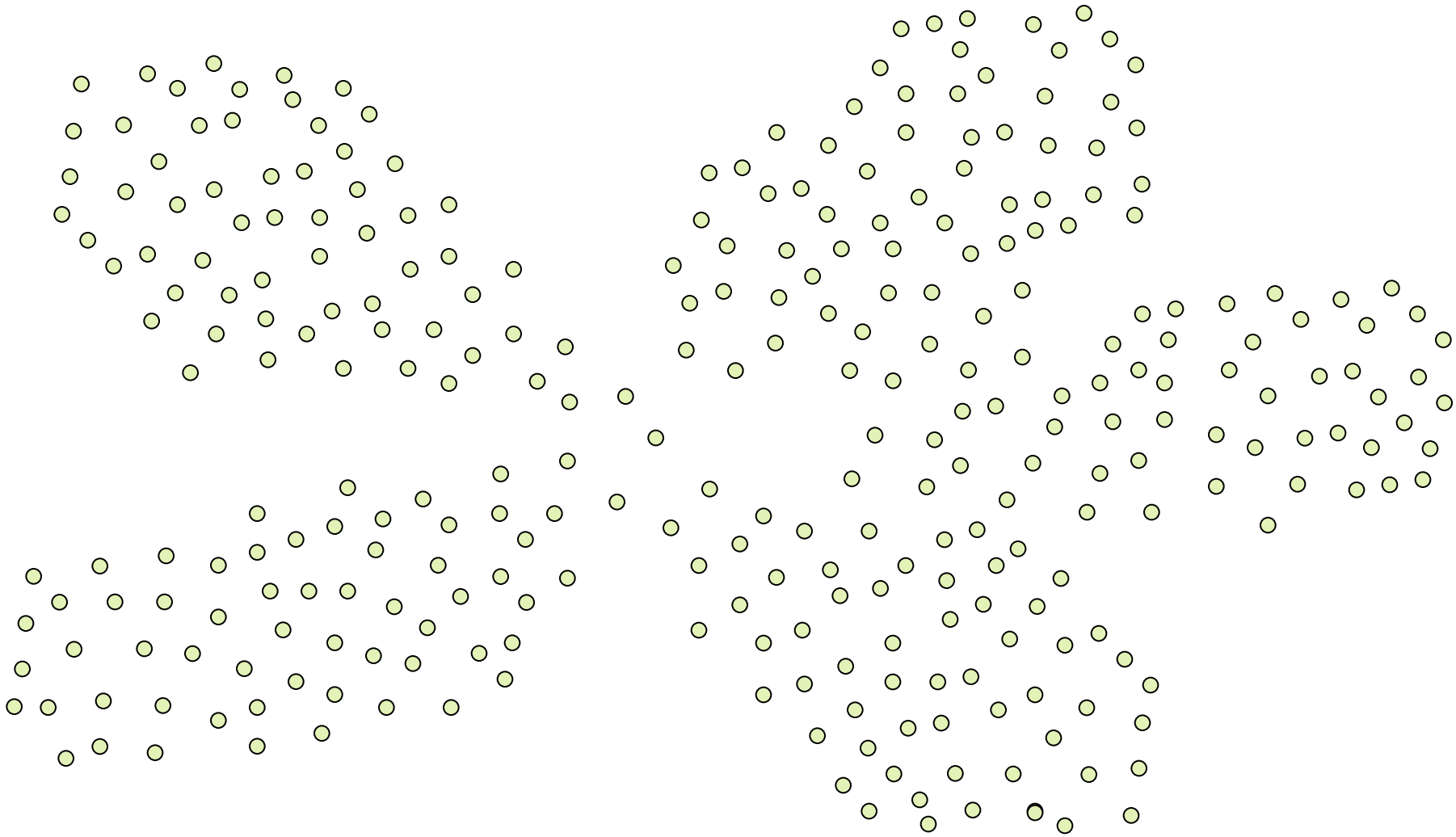


Partition #6

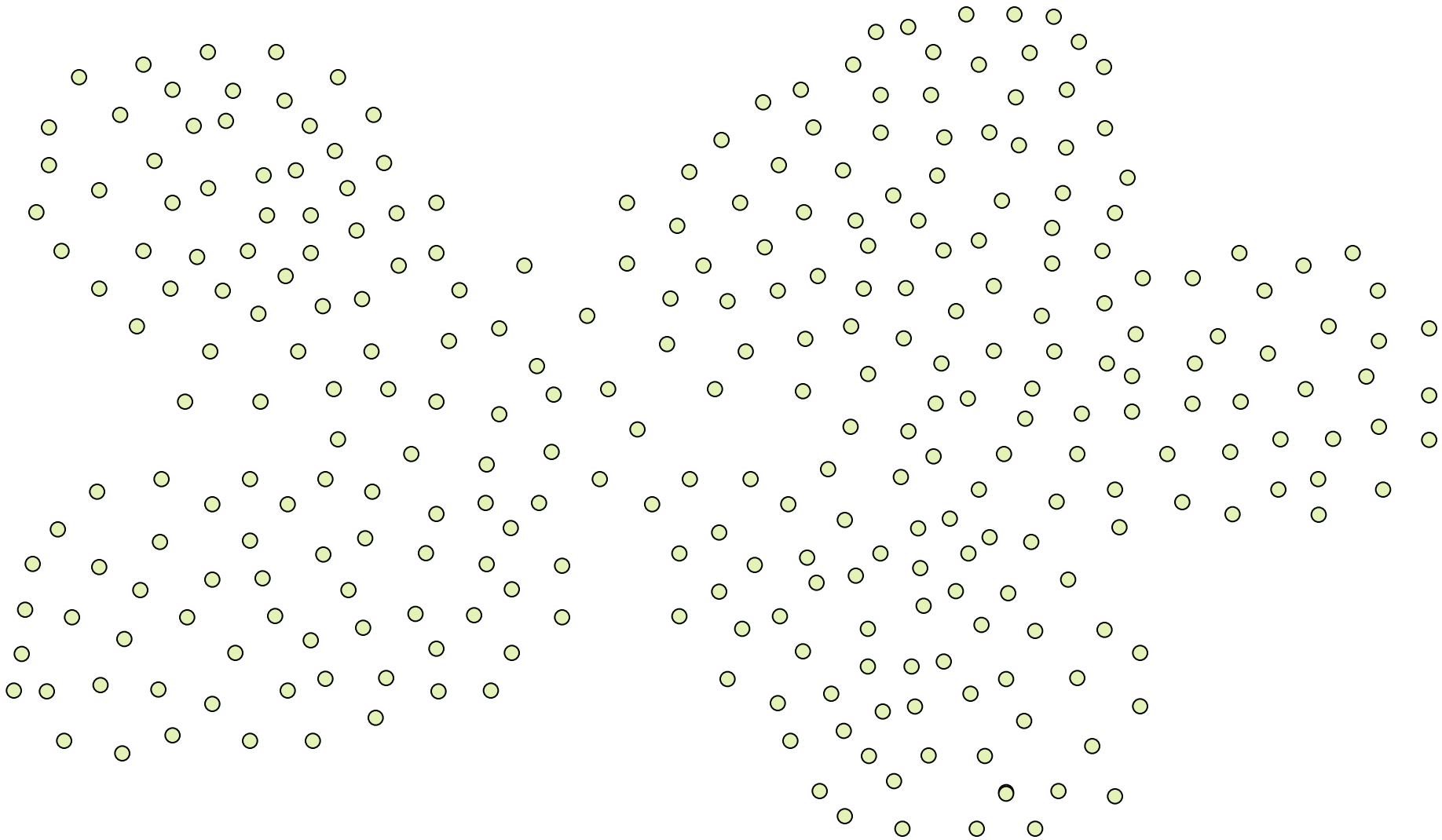
Damaged Network



Self Spreading & Motion Towards the Center



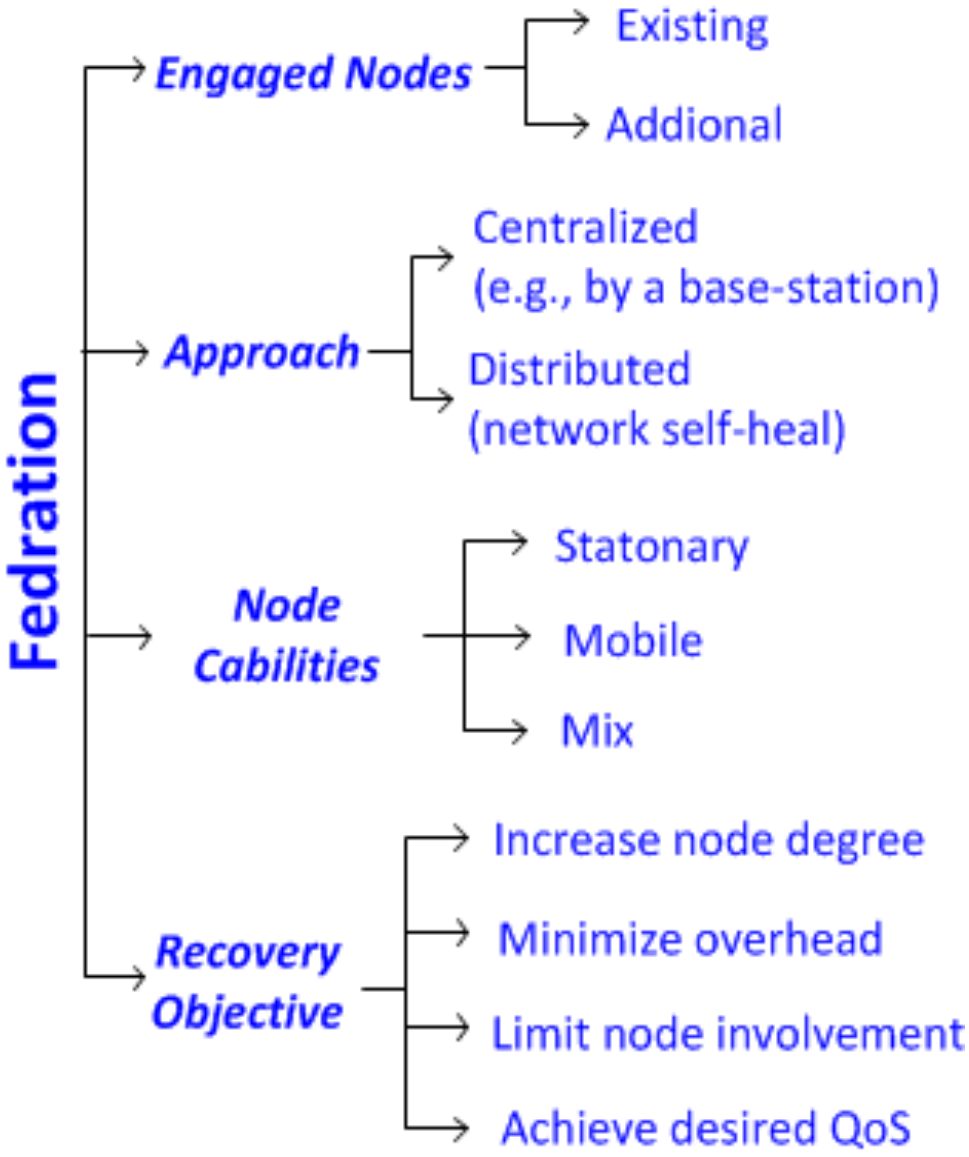
Final Self Spreading



Recovered Topology

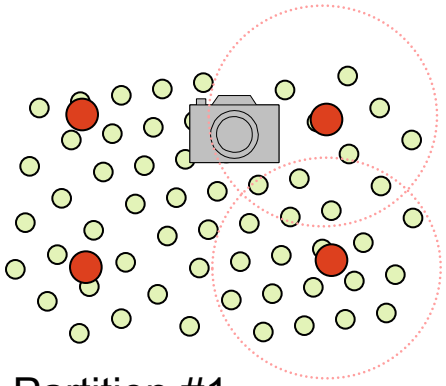
Sample Federation Approach

- What if the existing nodes (or at least most of them) cannot move?
- What if only stationary relays are available?

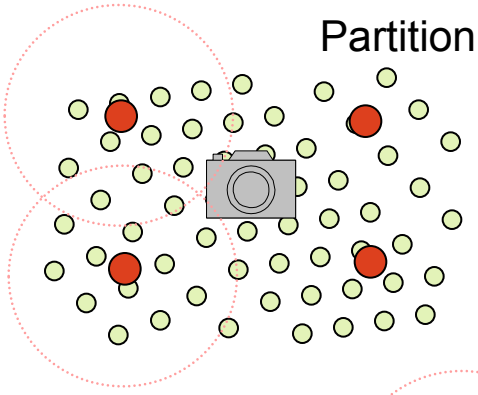


Problem Formulation

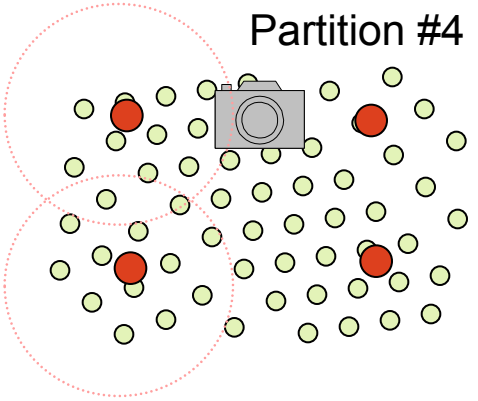
- **Objective:** To restore connectivity in a structurally damaged WSN or link multiple standalone WSN segments:
  - Proactive strategies: Not effective
  - Self-healing methodologies: Not feasible unless nodes can move
  - Solution: Employing additional resources (i.e. Relay Nodes)
- **Problem Statement:** Given *m* disjoint segments of sensors in an area of interest, determine the least count and position of relay nodes that are needed to connect all segments
  - Network segments may be represented with a single node (also called terminal)
  - This problem is equivalent to the Steiner Minimum Tree with Minimum Steiner Points and Bounded Edge Length (SMT-MSPBEL)
  - The problem is NP-Hard ➔ we pursued heuristics
  - Solution can serve general relay node placement applications
- **Variants of the Problem:**
  - Federate segments using least number of relays (possibly with additional goals)
  - Federate the segments using a fixed relay count (RC); All are mobile (prefer to keep them stationary)
  - Only a subset of the relays can move ($L_M + L_S = RC$, $L_M \geq 1$)



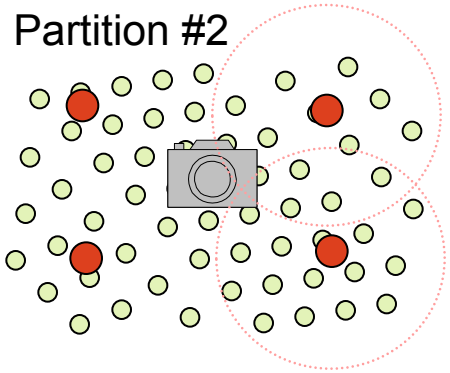
Partition #1



Partition #3



Partition #4



Partition #2

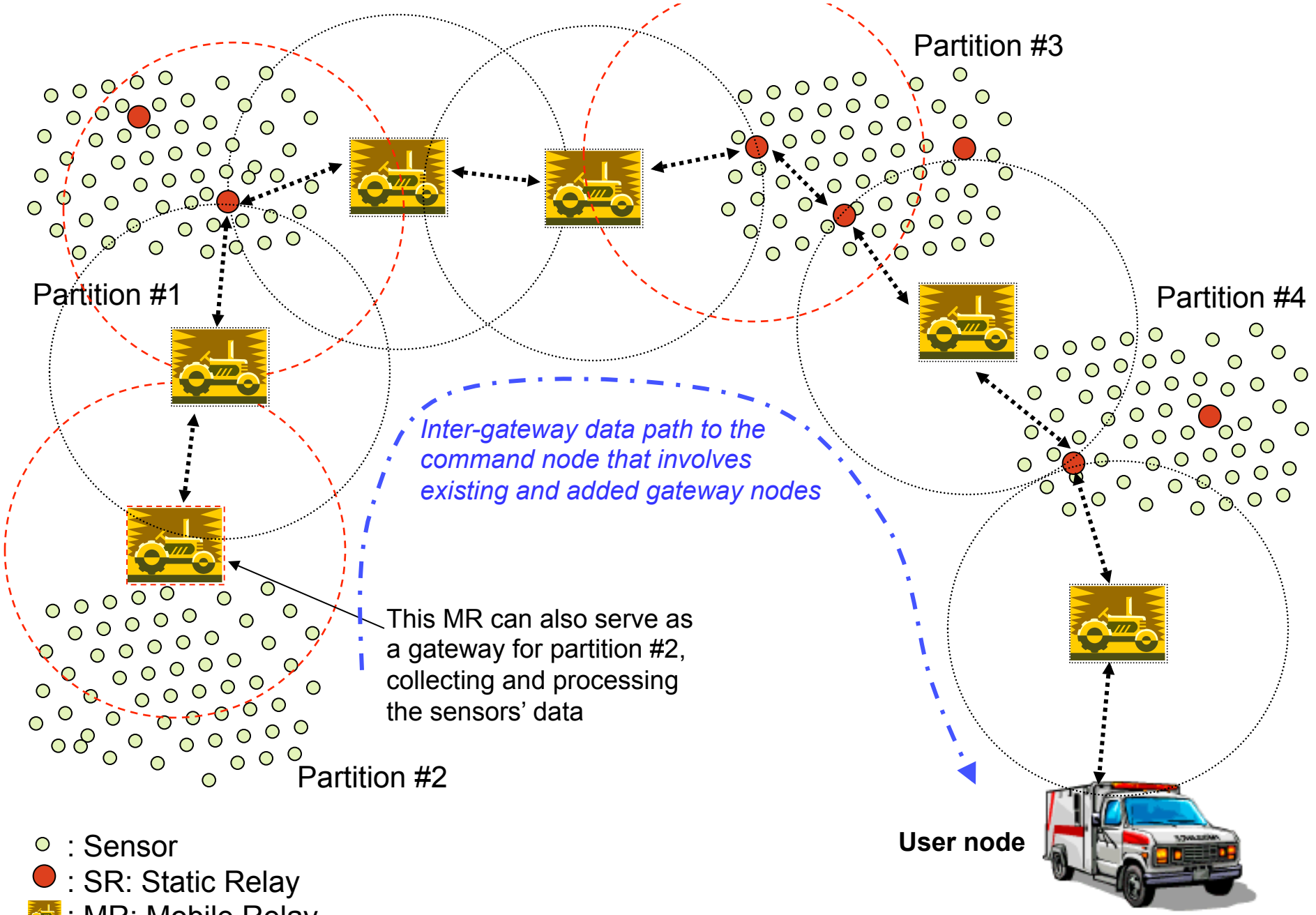
A disaster hits an area and response team arrives to the site. The local WSN suffers significant damage and gets partitioned. The rescue crew needs to quickly repair the network and federate few other private WSNs.

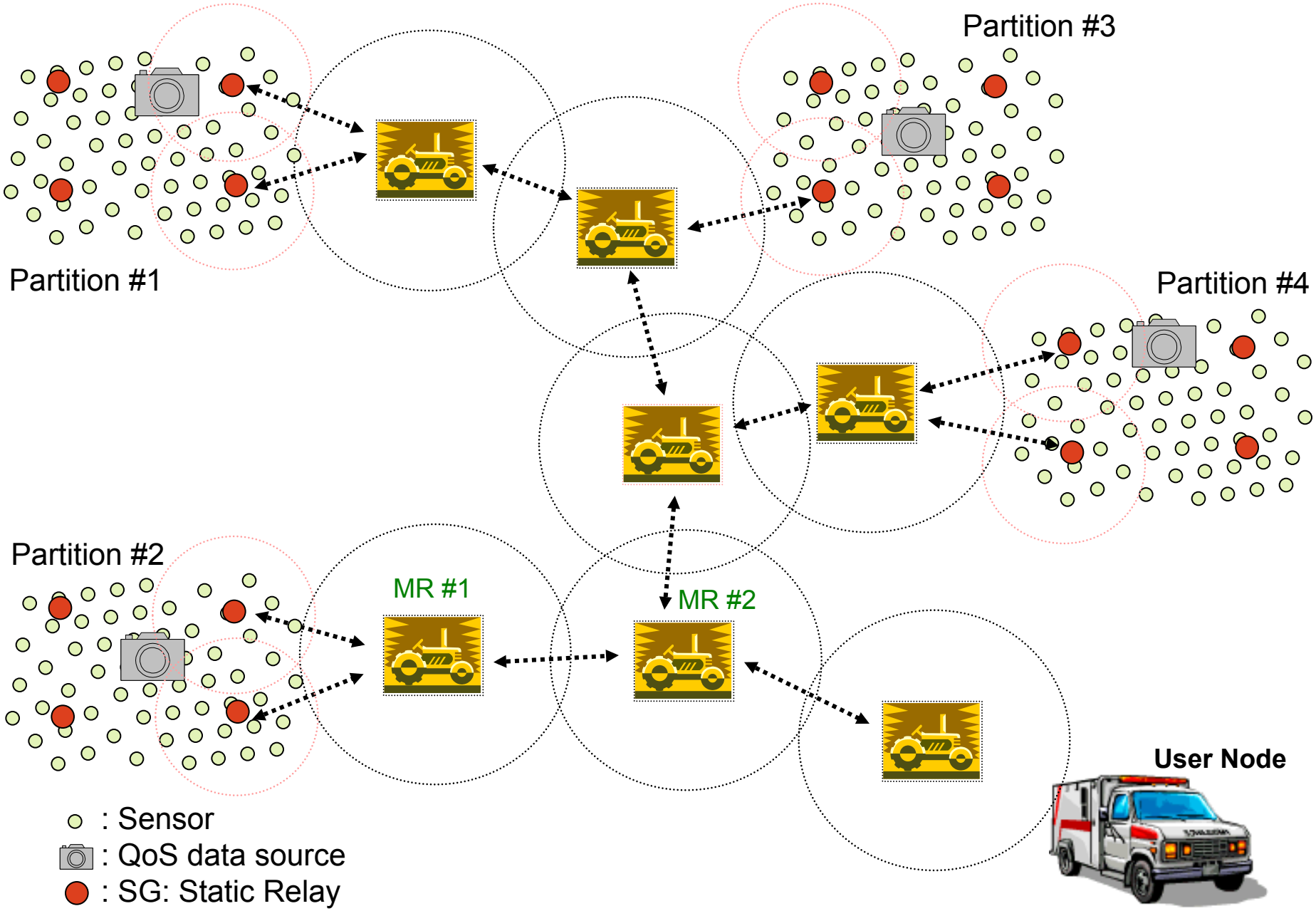
User Node

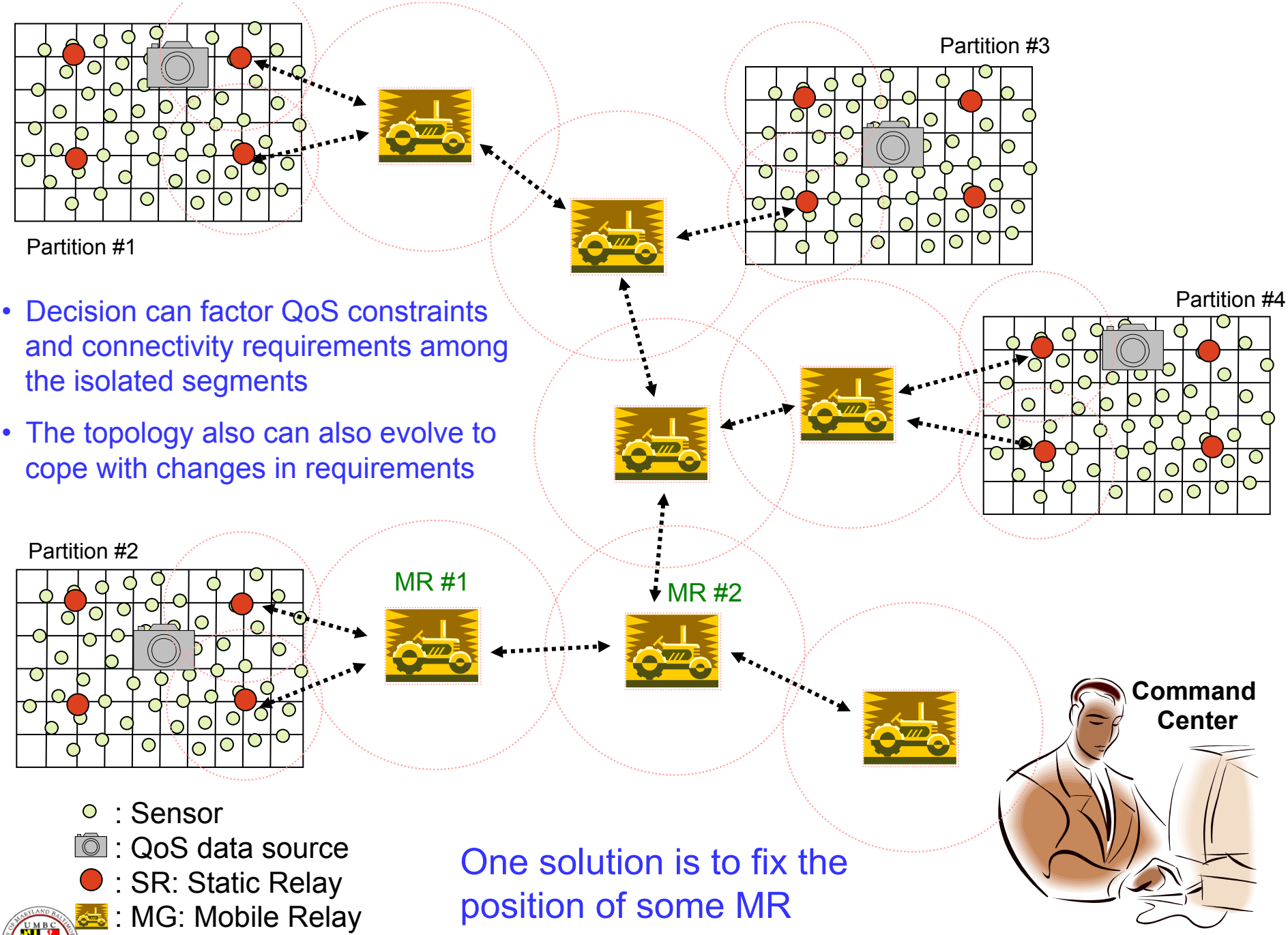
: Sensor

: QoS data source

: SG: Static Relay

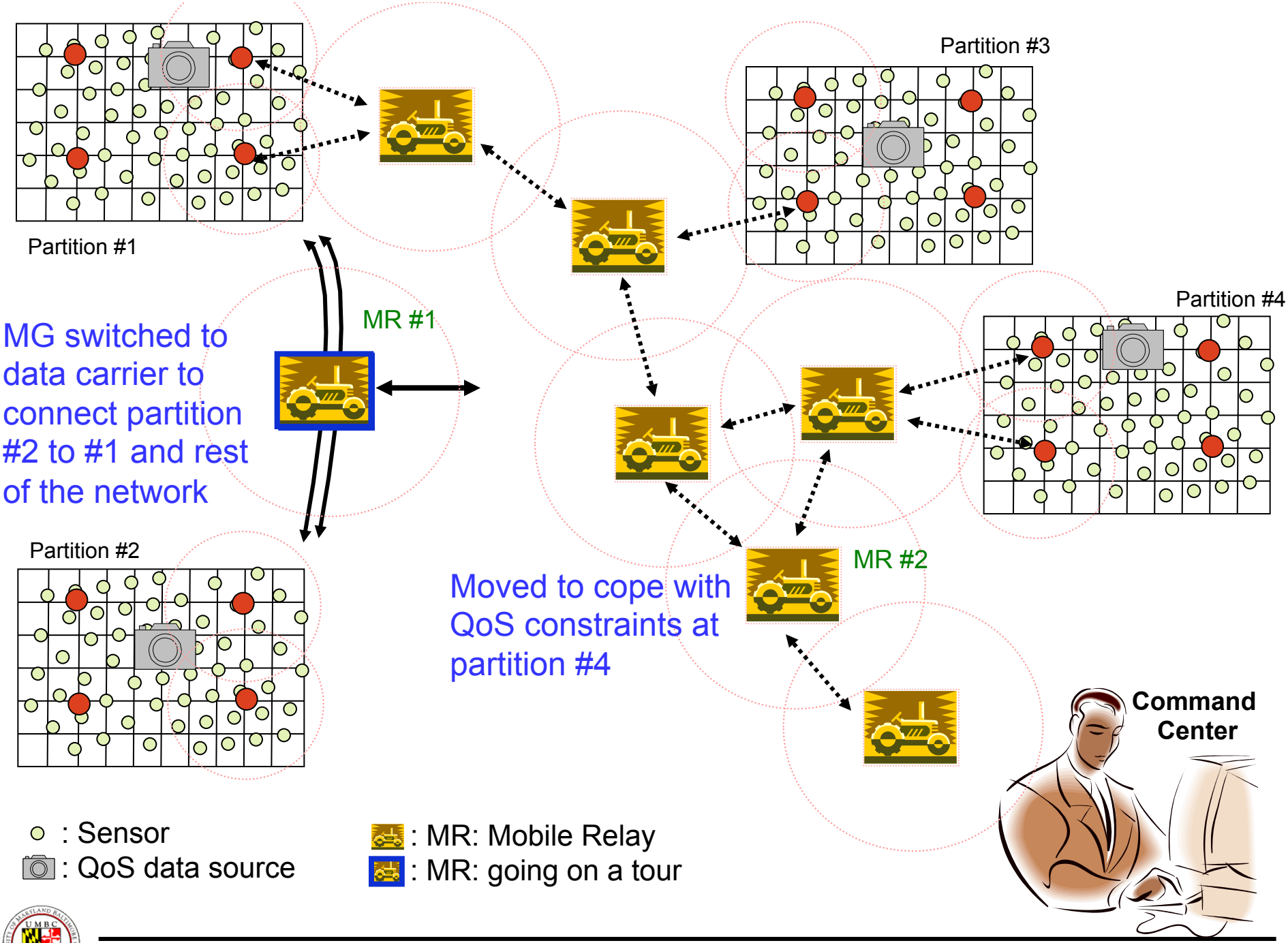

Partition #3
Partition #1
Partition #4
Inter-gateway data path to the command node that involves existing and added gateway nodes
This MR can also serve as a gateway for partition #2, collecting and processing the sensors' data
Partition #2
User node
: Sensor
: SR: Static Relay
: MR: Mobile Relay

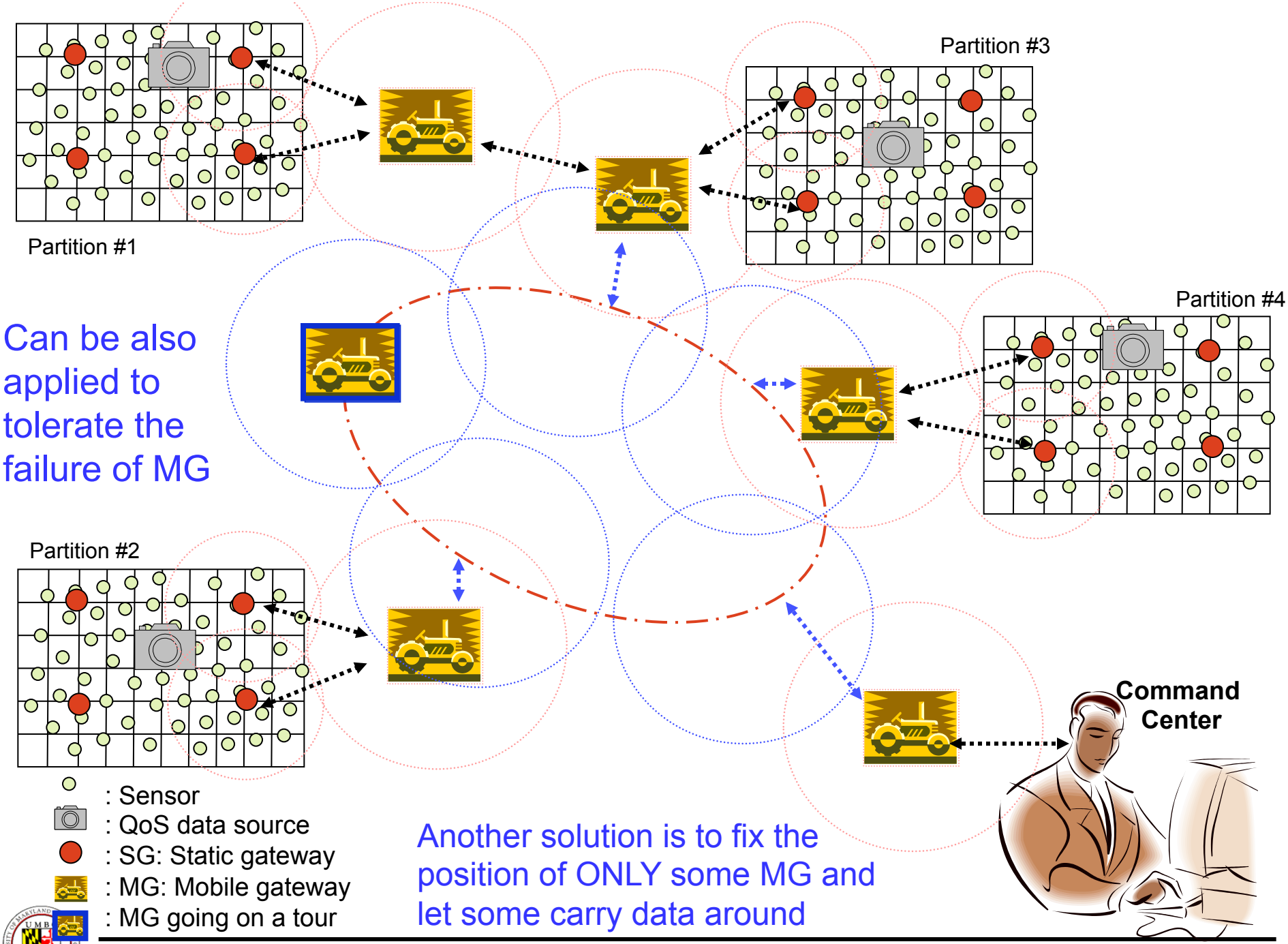

Partition #3
Partition #1
Partition #4
Partition #2
MR #1
MR #2
User Node
: Sensor
: QoS data source
: SG: Static Relay
: MG: Mobile Relay



- Decision can factor QoS constraints and connectivity requirements among the isolated segments
- The topology also can also evolve to cope with changes in requirements

One solution is to fix the position of some MR


Partition #1
Partition #3
Partition #4
Partition #2
MR #1
MR #2
MG switched to data carrier to connect partition #2 to #1 and rest of the network
Moved to cope with QoS constraints at partition #4
Command Center
: Sensor
: QoS data source
: MR: Mobile Relay
: MR: going on a tour



Can be also applied to tolerate the failure of MG

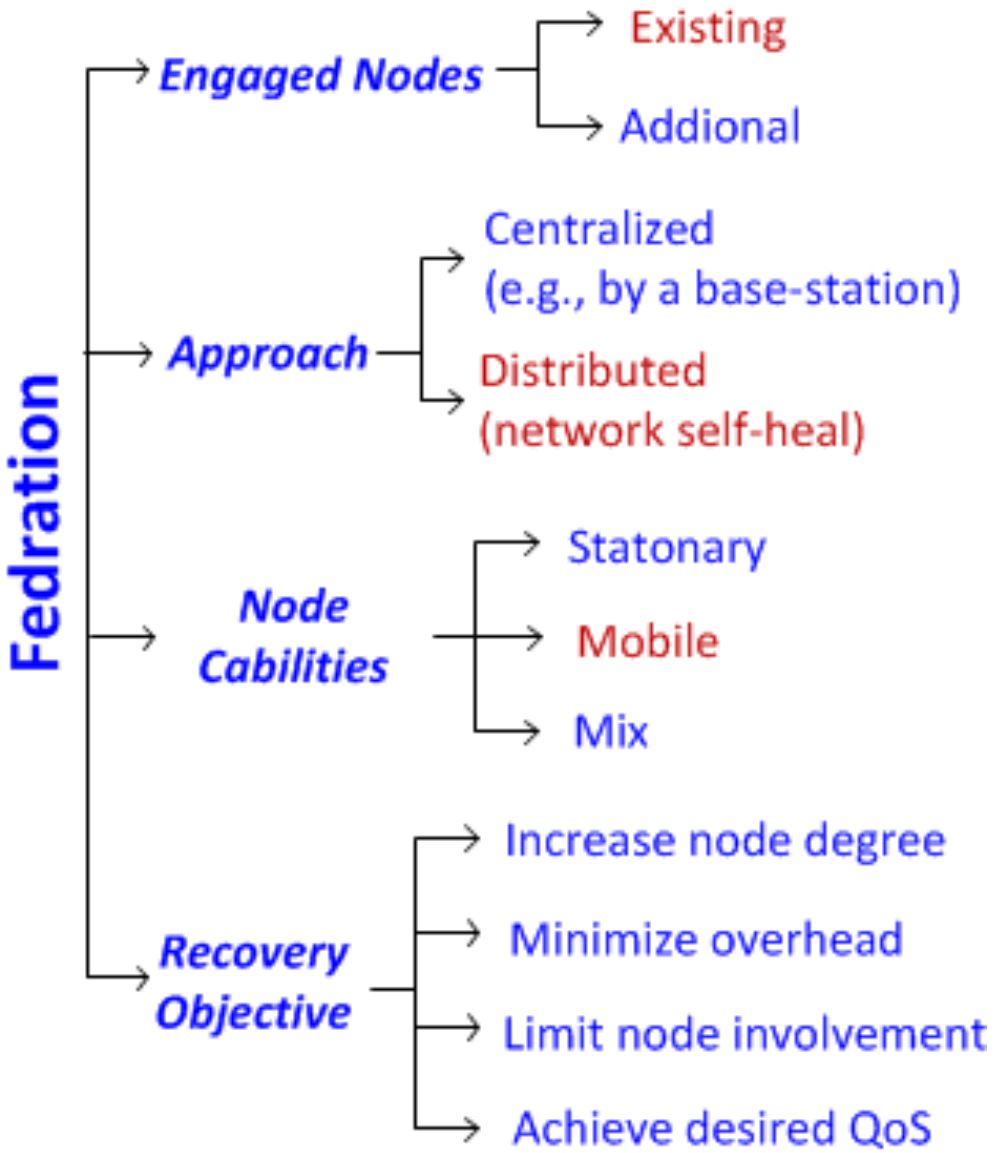
Another solution is to fix the position of ONLY some MG and let some carry data around

Sample Federation Approach

Two examples:

1. Incremental optimization of Triangles based on Delaunay Triangulation (IO-DT)

2. Distributed algorithm for Optimized Relay node placement using Minimum Steiner tree (DORMS)



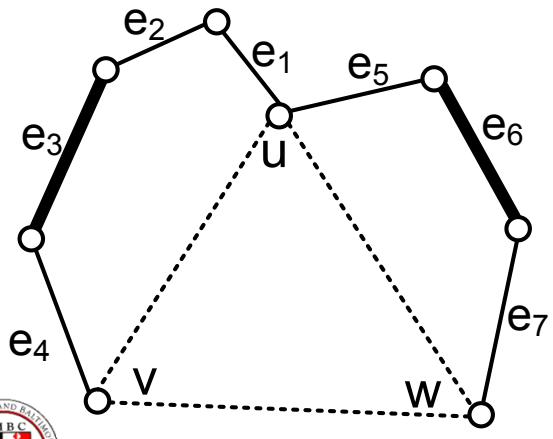
Optimal SMT-MSPBEL Solution For Three Terminals

- Definition: $W_{sp}(u, v)$ is the # of RNs to steinerize the edge (u,v)

$W_{sp}(u, v) = 2$

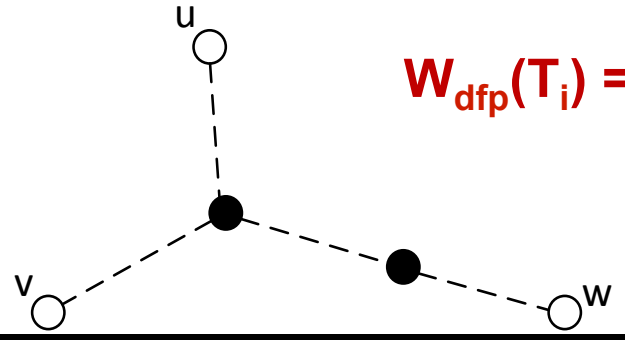
- Connecting a subset of 3 nodes u, v, w (a triangle) is a relatively easier problem
- Possible SMT-MSPBEL solutions for three points (i.e., triangle)

Case1: Steinerizing two mst edges which connects the corners of the triangle



$W_{mst}(T_i) = 3$

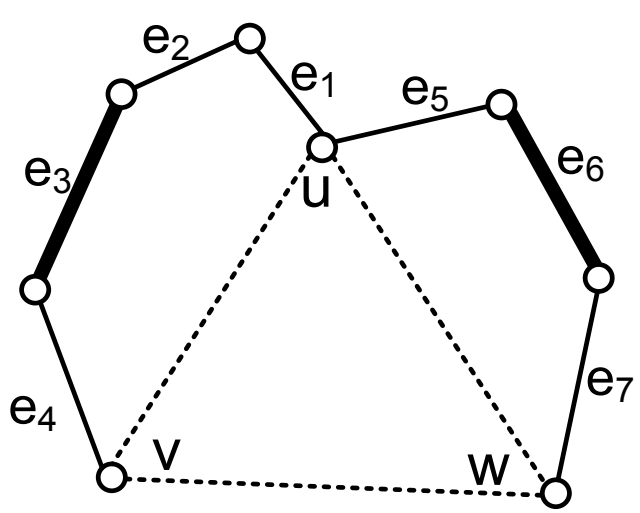
Case 2: Finding a point inside (which will be called Discrete Fermat Point (DFP)) the triangle and connecting corners of the triangle to the DFP

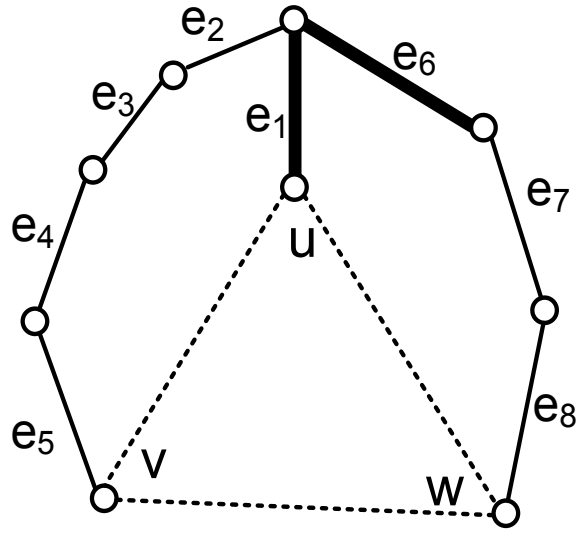


$W_{dfp}(T_i) = 2$

Case1: *mst-weight* of a Triangle

- mst-weight for a triangle $T(u,v,w)$
  1) Find two mst paths Υ_{uv} and Υ_{uw} from u to v and from u to w, respectively
  2) The edges with of the largest weights in Υ_{uv} and Υ_{uw} correspond the mst weight of $T(u,v,w)$





Case 2: Finding Discrete Fermat Point (DFP)

- **Discrete Fermat Point** of triangle $T(u,v,w)$ is such a point $\varphi\downarrow i$ inside the triangle that minimizes

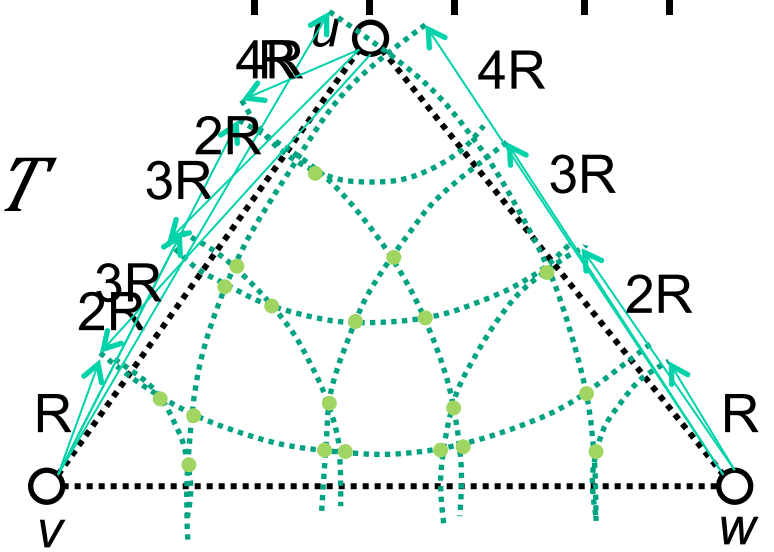
$$\varphi\downarrow i = \min\nolimits_{\forall q} \{\lceil |uq|/R \rceil + \lceil |vq|/R \rceil + \lceil |wq|/R \rceil\}$$

- How to find: Infinite number of possible choices!!!!
- **Theorem:** The number of points, which can possibly be DFP for a $T(u,v,w)$, does not exceed

$$\lceil 4s(s-|uv|)(s-|uw|)(s-|vw|)/R\uparrow 2 \times |uv|+|uw|+|vu|/|vw|\times|uw|\times|uv| \rceil$$

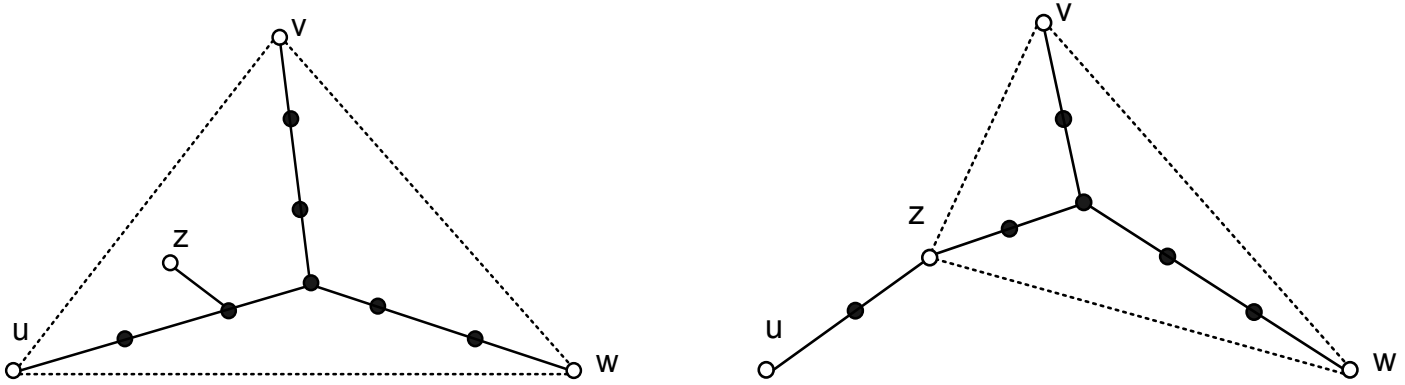
where s is the semi-perimeter of triangle T

Candidate DFPs!!



Incremental optimization of Triangles based on Delaunay Triangulation (IO-DT)

- Goal: leverage the DFP based steinerization as much as possible
- Question: how to identify a subset of triangles which minimizes total number of relays -- Number of all possible triangles is $O(n^3)$
- Considering big triangles having multiple terminal inside may require redundant RN deployment

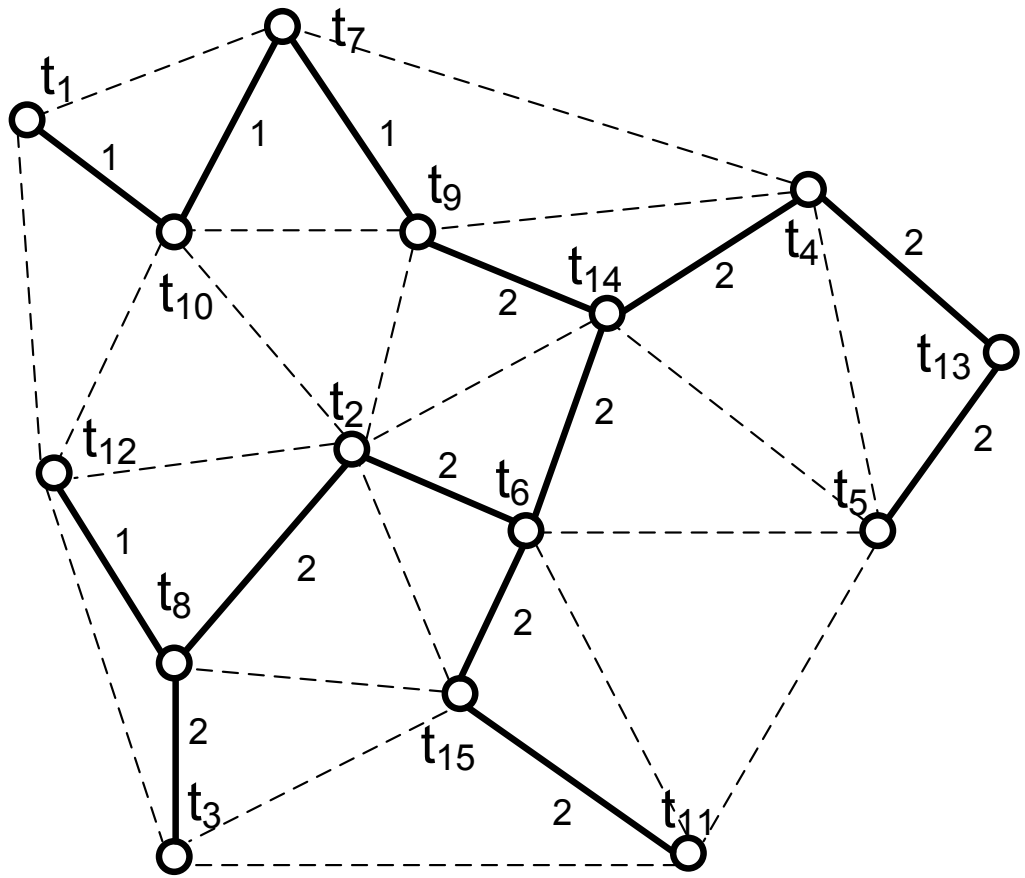


- To avoid such redundancy we need a triangulation such that no terminal is located inside a triangle
  - **Solution**: **Delaunay triangulation**

IO-DT Heuristic

- There are 4 main steps of the algorithm
  1. Calculate Delaunay Triangulation (DT) of the terminals
  2. Calculate *mst* of the terminals
  3. Sort the triangles in DT according to their dfp-weights in ascending order
  4. Iterate over the sorted list of triangles
     - In each iteration, if the dfp-weight of the triangle is less than mst-weight, then place relay at the dfp of the triangle and update mst on-the fly.
- Run Time Complexity of IO-DT is $O(n^2)$

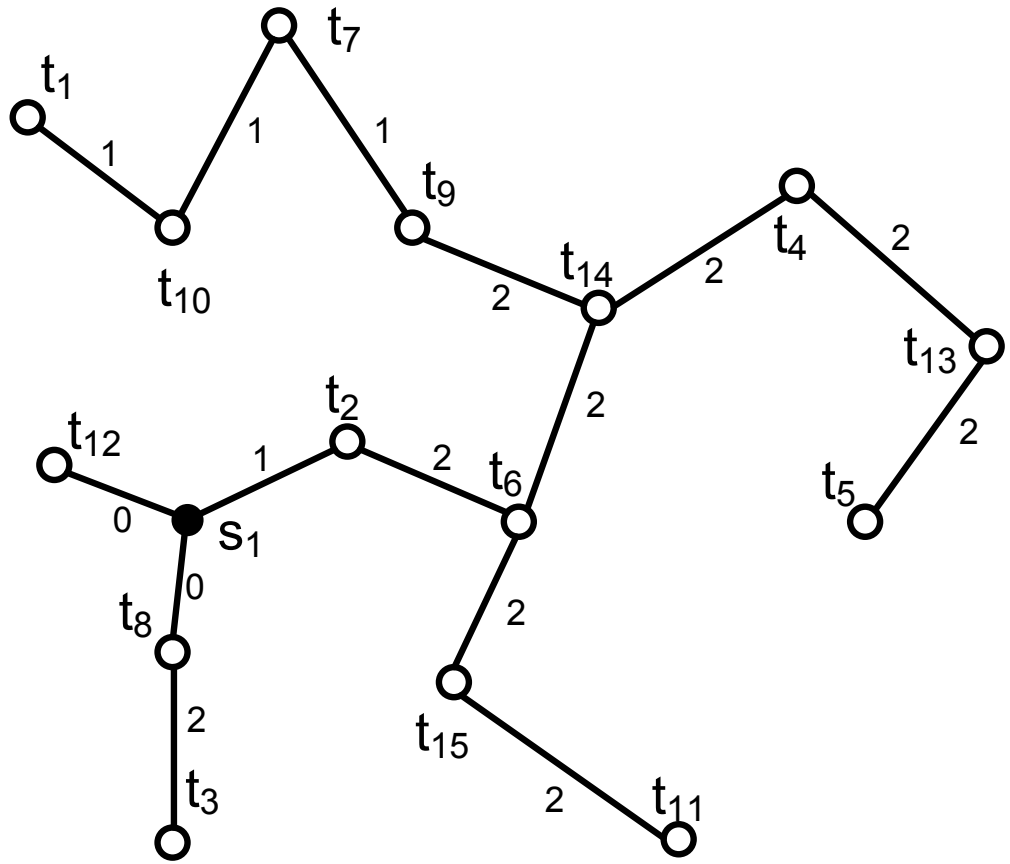
IO-DT Example



Triangle T_i	$W_{dfp}(T_i)$	$W_{mst}(T_i)$
(t_7, t_9, t_{10})	2	2
(t_1, t_7, t_{10})	2	2
(t_2, t_8, t_{12})	2	3
(t_1, t_{10}, t_{12})	2	3
(t_6, t_{11}, t_{15})	3	4
(t_3, t_8, t_{15})	3	4
(t_2, t_6, t_{15})	3	2
(t_2, t_9, t_{14})	3	3
(t_2, t_6, t_{14})	3	3
(t_4, t_5, t_{13})	3	4
	3	2

Delaunay Triangulation of terminals is calculated and list of triangles is sorted according to their dfp-weight

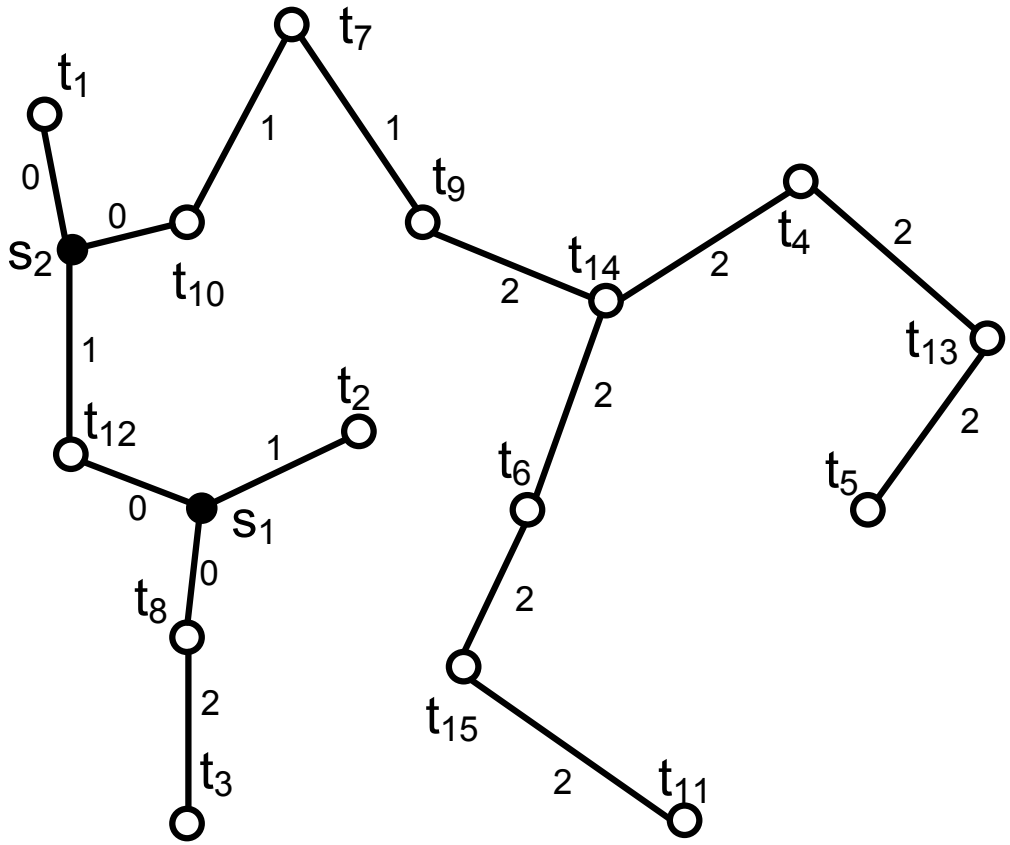
IO-DT Example



Triangle $T\downarrow i$	$W\downarrow dfp\ (T\downarrow i)$	$W\downarrow mst\ (T\downarrow i)$
$(t\downarrow 7, t\downarrow 9, t\downarrow 10)$	2	2
$(t\downarrow 1, t\downarrow 7, t\downarrow 10)$	2	2
$(t\downarrow 2, t\downarrow 8, t\downarrow 12)$	2	3
$(t\downarrow 1, t\downarrow 10, t\downarrow 12)$	2	3
$(t\downarrow 6, t\downarrow 11, t\downarrow 15)$	3	4
$(t\downarrow 3, t\downarrow 8, t\downarrow 15)$	3	4
$(t\downarrow 2, t\downarrow 6, t\downarrow 15)$	3	2
$(t\downarrow 2, t\downarrow 9, t\downarrow 14)$	3	3
$(t\downarrow 2, t\downarrow 6, t\downarrow 14)$	3	3
$(t\downarrow 4, t\downarrow 5, t\downarrow 13)$	3	4
$(t\downarrow 2, t\downarrow 9,$	3	2

First triangle where $W_{dfp} < W_{mst}$ is (t_2, t_8, t_{12})
We place a relay to the DFP of (t_2, t_8, t_{12})
and update the *mst* accordingly

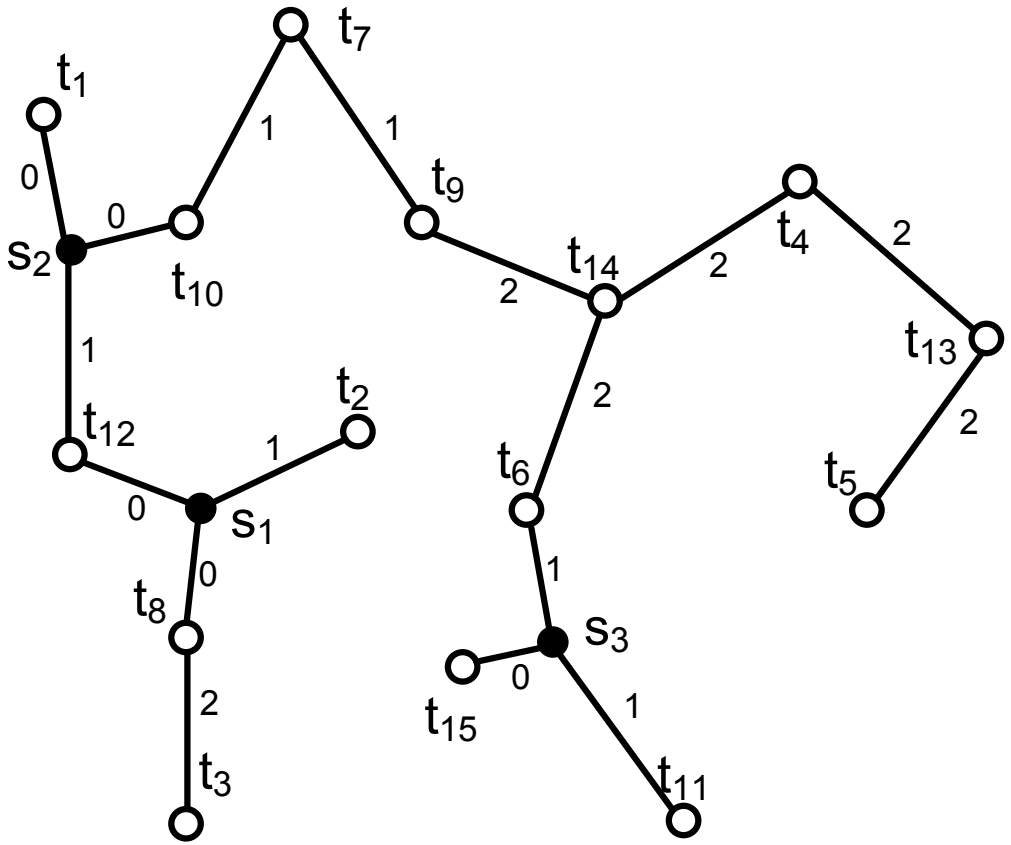
IO-DT Example



Similarly we continue iterating until all triangles are processed

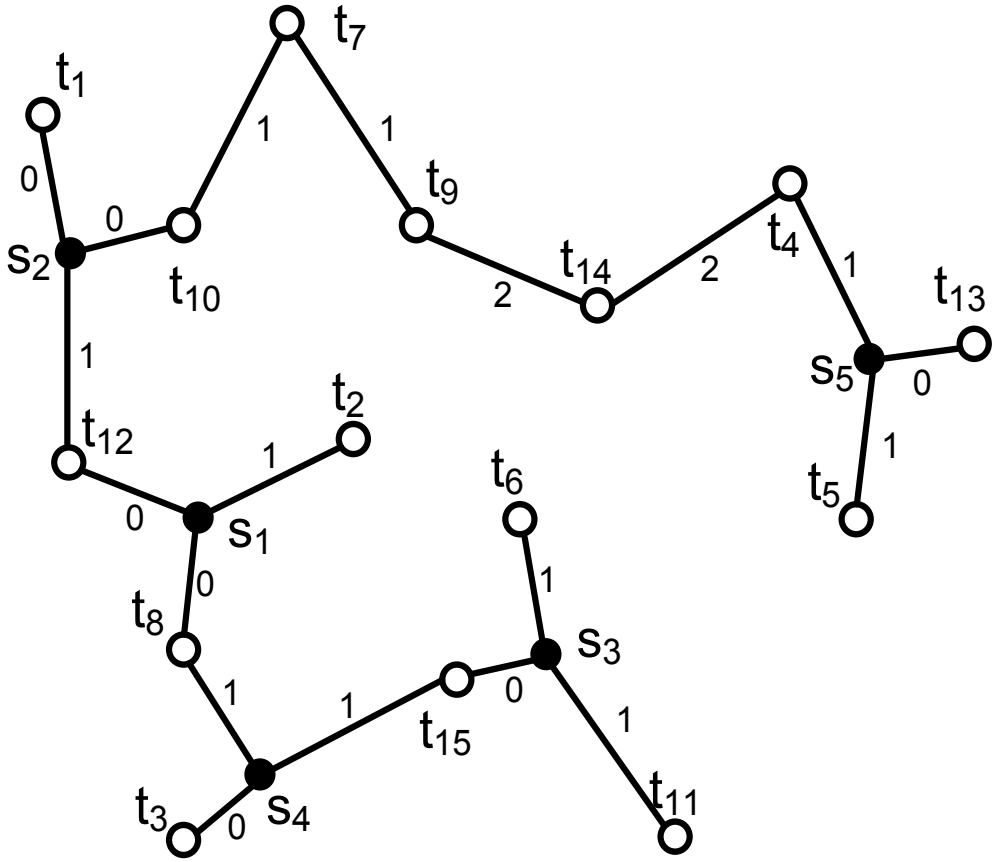
Triangle T_i	$W_{dfp}(T_i)$	$W_{mst}(T_i)$
(t_7, t_9, t_{10})	2	2
(t_1, t_7, t_{10})	2	2
(t_2, t_8, t_{12})	2	3
(t_1, t_{10}, t_{12})	2	3
(t_6, t_{11}, t_{15})	3	4
(t_3, t_8, t_{15})	3	4
(t_2, t_6, t_{15})	3	2
(t_2, t_9, t_{14})	3	3
(t_2, t_6, t_{14})	3	3
(t_4, t_5, t_{13})	3	4
(t_4, t_9, t_{14})	3	2

IO-DT Example



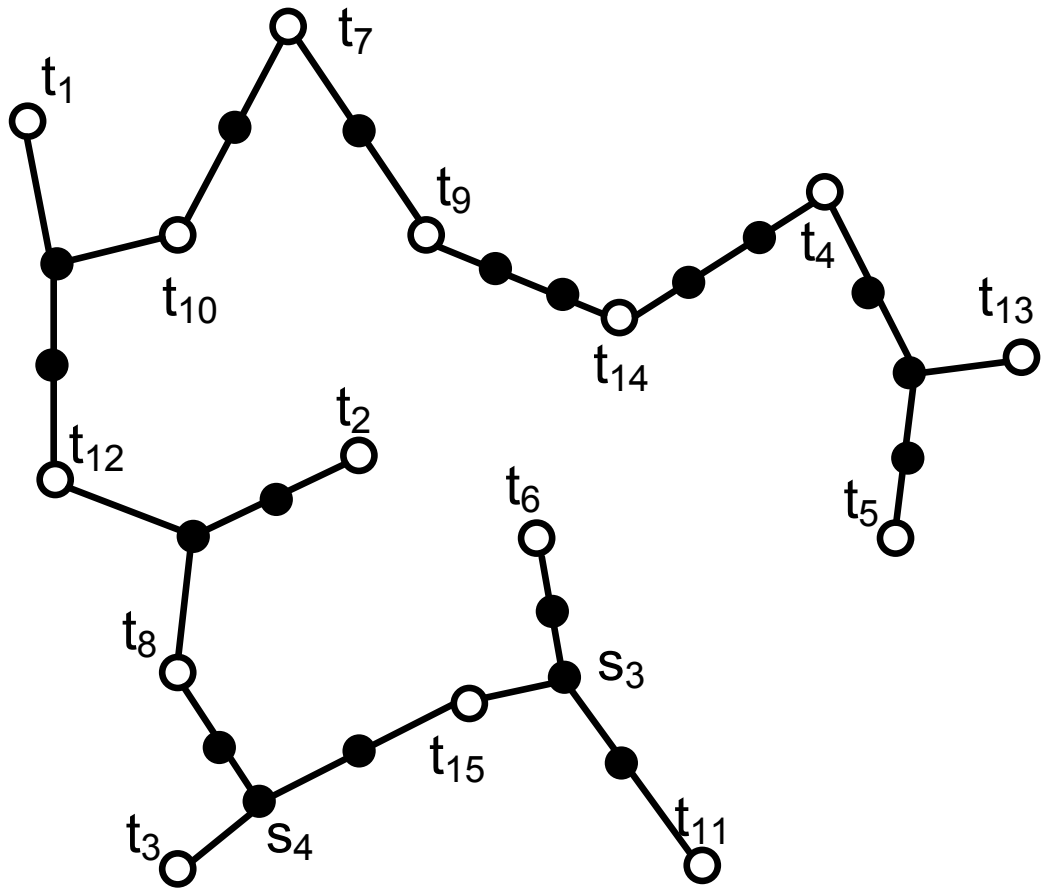
Triangle T_i	$W_{dfp}(T_i)$	$W_{mst}(T_i)$
(t_7, t_9, t_{10})	2	2
(t_1, t_7, t_{10})	2	2
(t_2, t_8, t_{12})	2	3
(t_1, t_{10}, t_{12})	2	3
(t_6, t_{11}, t_{15})	3	4
(t_3, t_8, t_{15})	3	4
(t_2, t_6, t_{15})	3	2
(t_2, t_9, t_{14})	3	3
(t_2, t_6, t_{14})	3	3
(t_4, t_5, t_{13})	3	4
	3	2

IO-DT Example



Triangle T_i	$W_{dfp}(T_i)$	$W_{mst}(T_i)$
(t_7, t_9, t_{10})	2	2
(t_1, t_7, t_{10})	2	2
(t_2, t_8, t_{12})	2	3
(t_1, t_{10}, t_{12})	2	3
(t_6, t_{11}, t_{15})	3	4
(t_3, t_8, t_{15})	3	4
(t_2, t_6, t_{15})	3	2
(t_2, t_9, t_{14})	3	3
(t_2, t_6, t_{14})	3	3
(t_4, t_5, t_{13})	3	4
(t_2, t_9, t_{10})	3	2
(t_2, t_{10}, t_{12})	3	2
(t_3, t_8, t_{12})	4	1
(t_3, t_{11}, t_{15})	4	2
(t_2, t_8, t_{15})	4	2
(t_5, t_6, t_{14})	4	4
(t_4, t_5, t_{14})	4	3
(t_4, t_9, t_{14})	4	4
(t_4, t_7, t_9)	5	3
(t_5, t_6, t_{11})	5	3

IO-DT Example



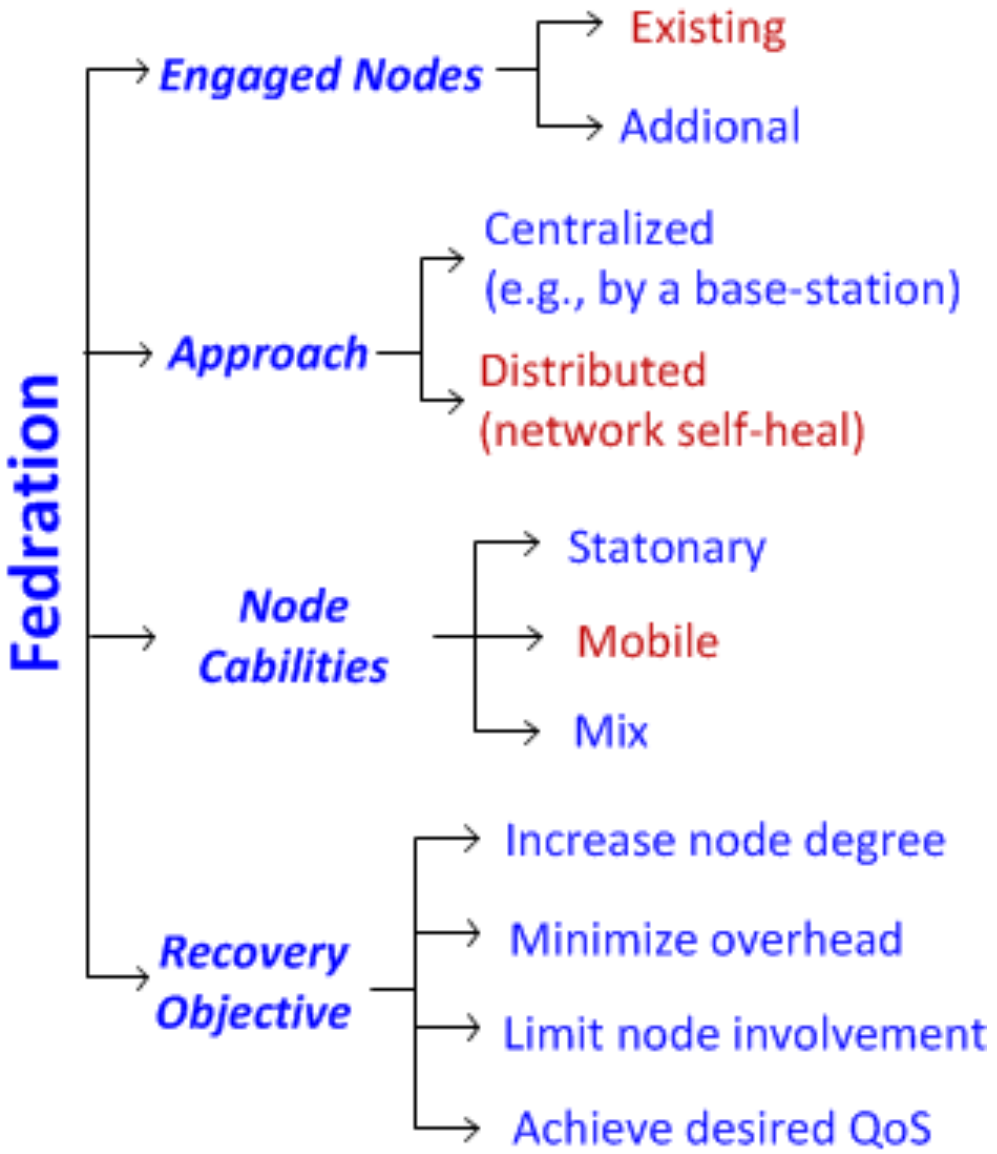
Final Topology after steinerizing mst edges

Triangle T_i	$W_{dfp}(T_i)$	$W_{mst}(T_i)$
(t_7, t_9, t_{10})	2	2
(t_1, t_7, t_{10})	2	2
(t_2, t_8, t_{12})	2	3
(t_1, t_{10}, t_{12})	2	3
(t_6, t_{11}, t_{15})	3	4
(t_3, t_8, t_{15})	3	4
(t_2, t_6, t_{15})	3	2
(t_2, t_9, t_{14})	3	3
(t_2, t_6, t_{14})	3	3
(t_4, t_5, t_{13})	3	4
(t_4, t_9, t_{14})	3	2

Sample Federation Approach

An example:

Distributed algorithm for Optimized Relay node placement using Minimum Steiner tree (DORMS)

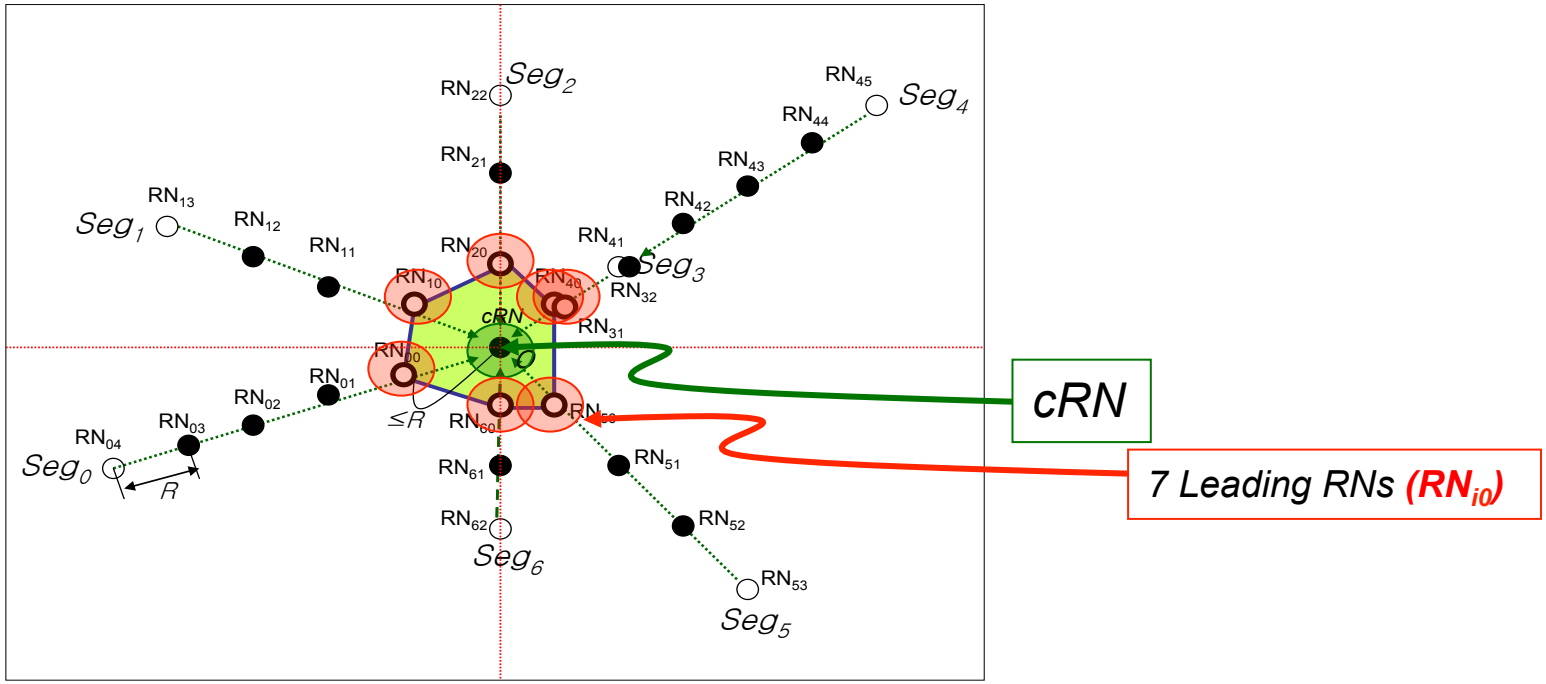


Distributed algorithm for Optimized Relay node placement using Minimum Steiner tree (DORMS)

- *Self-healing* a large scale damaged WSNs
- *No network-wide analysis* required to diagnose where segments are located after damage
- Employing *the available relays in each segment* that survived from the damage.
- Intra-segment topology is assumed to be maintained.
- *3 main steps*
  - *Quick initial deployment* towards the core of all segments from each segment
  - *Optimization* for reducing the number of deployed relay using the minimum Steiner tree obtained by k-LCA.
  - *Relocation* of the relays based on the minimum Steiner tree based topology by retreating nonessential relays to their respective segments.

3 Main steps of DORMS (1/3)

1. Initial deployment of relay nodes
   - DORMS initially populates relays along a path P_i from each segment towards a center "O" of the deployment area with a displacement of "R", where R is the communication range of a relay node.
   - Upon terminating initial deployment, each populated relay reports its ID (RN_{id}) and its location (Loc_RN_{id}) to its representative relay, RN_{i0}.
   - The number of nodes NP_i on a path P_i will be at most $\lceil Length(P_i)/R \rceil$-1.



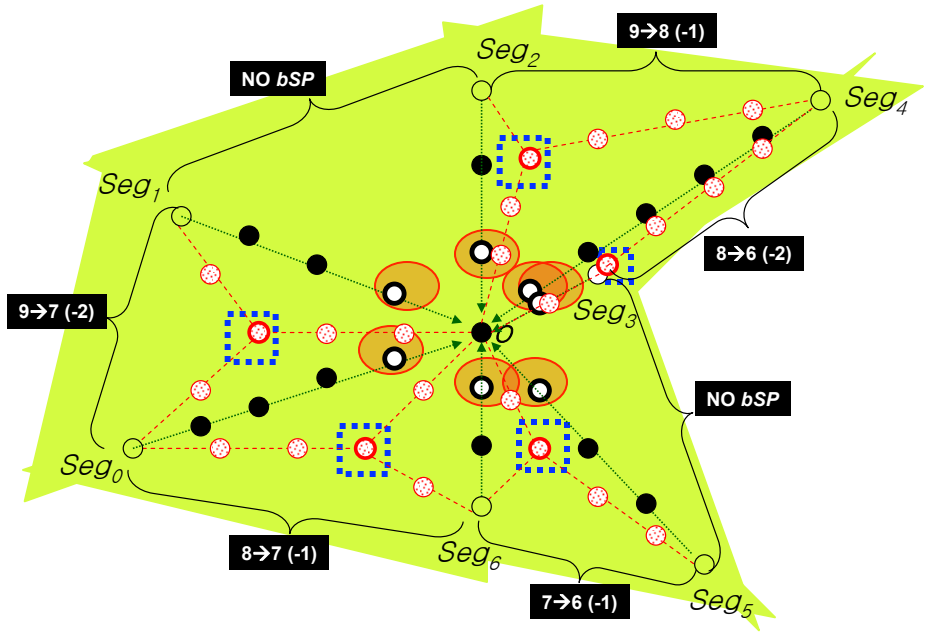
3 Main steps of DORMS (2/3)

2. Optimization of the number of populated RNs
   - Using the best-known approximation algorithm, k-LCA to find a Minimum Steiner Tree (MST), DORMS opts to *identify essential nodes for the inter-segment connectivity and return unnecessary RNs back to their respective segment*

   a) 7 RN_{i0}'s exchange the location of their serving segments.

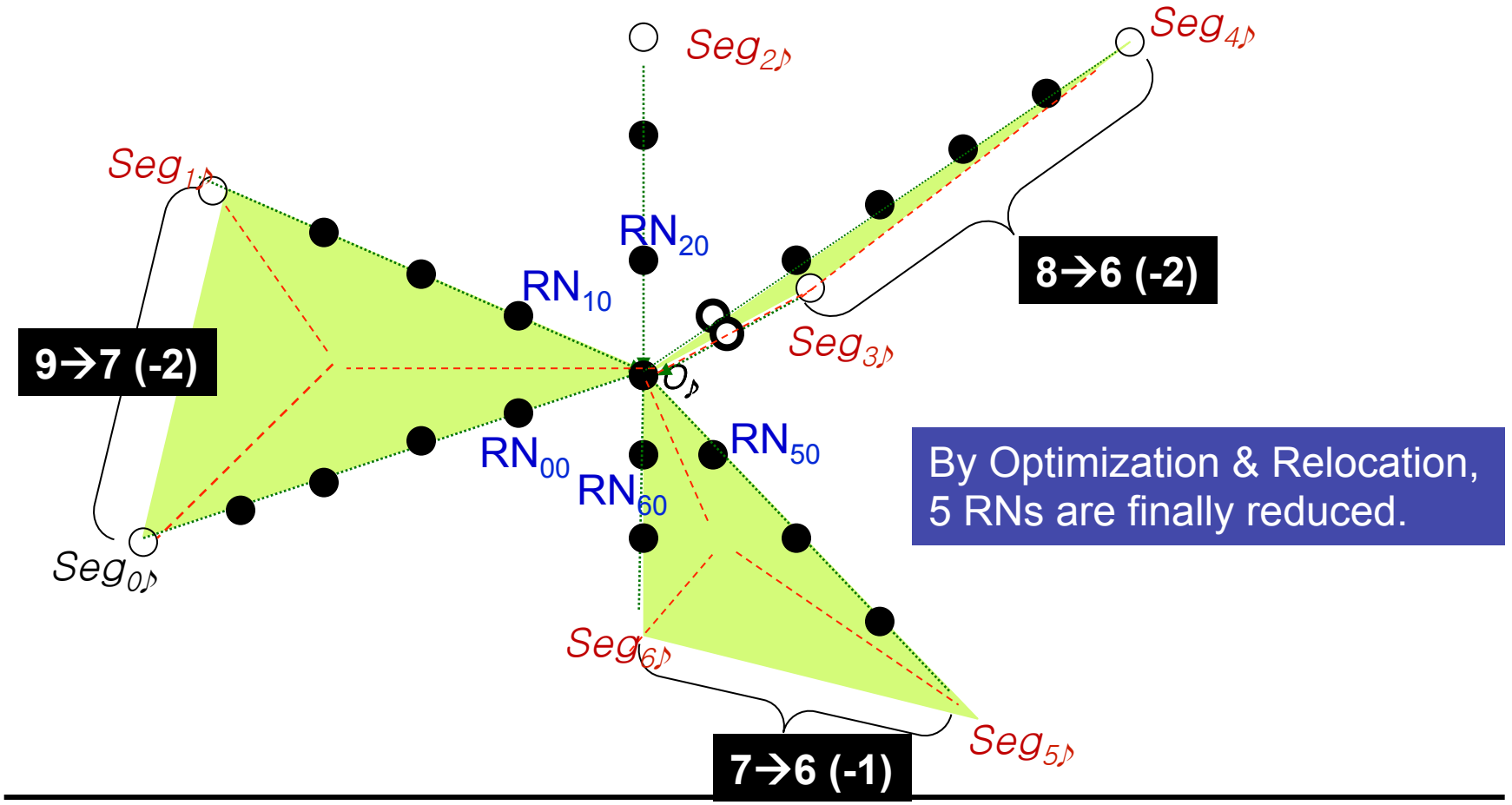
   b) Compute minimum Steiner tree of {Seg_i, O, Seg_j}

   c) Find "*best Steiner Points*" that minimize the number of relays required for connecting Seg_i, Seg_j and O.



3 Main steps of DORMS (3/3)

3. Relay relocation to reduce relay count needed for inter-segment connectivity
   - Select the best MSTs which *reduce the total number of required RNs* overall.
   - Relocating *minimizing the relocation time and then distance*
   - *Retreat* unnecessary RNs into the respective segment



Federation through Touring

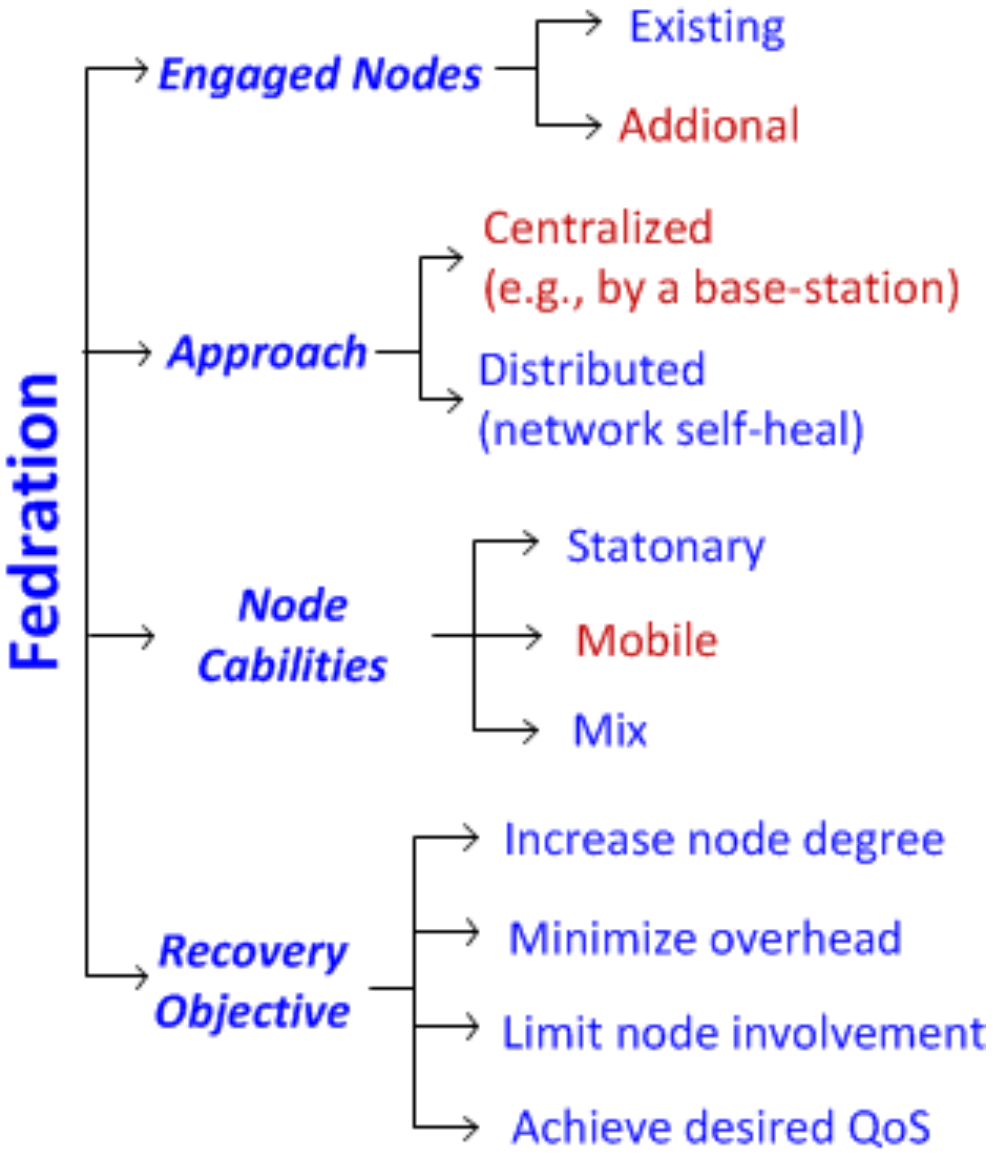
- To recover from a large scale damage and restore connectivity in a structurally damaged WSN:
  - Solution: Employing additional resources (i.e. Mobile Data Collectors)
- **System Model:** Each network segment is represented by a single node (i.e., representative)
- **Problem Statement:** Given the set of *n* representatives $S=\{s_1, s_2, ..., s_n\}$ and *k* MDCs, determine the set of tours $\{T_{S_1}, T_{S_2}, ..., T_{S_k}\}$ such that $S_i \subseteq S$, $1 \le i \le k$ and $S = \bigcup_{i=1}^{k} S_i$ where $\sum_{i=1}^{k} |T_{S_i}|$ is minimized."

Sample Federation Approach

Two examples:

1. Heuristic for Interconnection of Disjoint wireless network Segments using K Mobile Data Collectors (IDS-kMDC)
   - ➢ Focus on travel distance overhead

2. Touring of Clustered Segments (ToCS)
   - ➢ Strive to minimize the data delivery delay

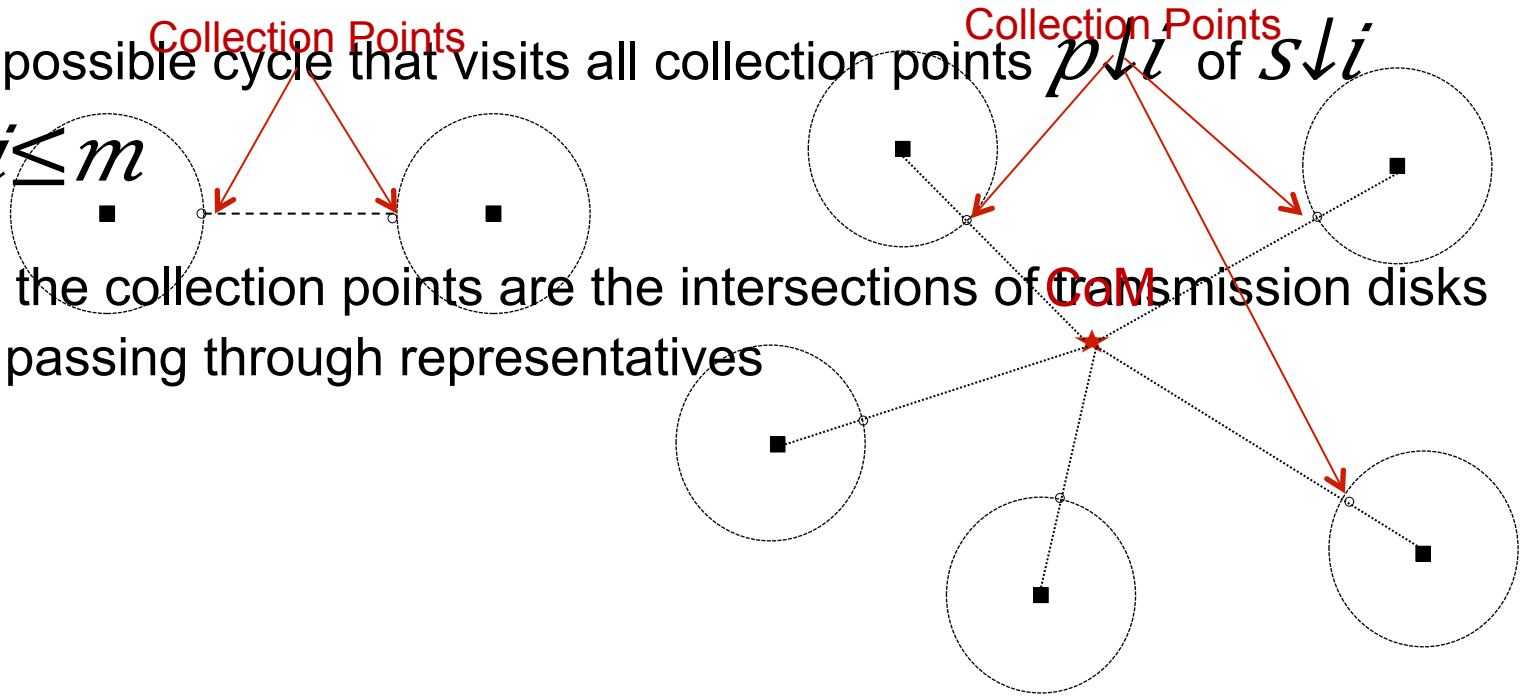


Federation Using *K* Mobile Relays

- Few **mobile** relays are available for federating multiple network segments
  - ✓ **Employ them as mobile data carrier (MDC)**
- **System Model:** A segment is modeled by a single node (i.e., representative)
- **Problem Statement:** Given the set of *n* representatives $S=\{s\downarrow 1 , s\downarrow 2 , ..., s\downarrow n\}$ and *k* MDCs, determine the set of tours $\{T\downarrow S\downarrow 1 , T\downarrow S\downarrow 2 , ..., T\downarrow S\downarrow k\}$ such that $S\downarrow i \subseteq S$, $1\leq i\leq k$ and $S=U i=1\uparrow k S\downarrow i$ where $\sum i=1\uparrow k |T\downarrow S\downarrow i|$ is minimized."
- We can split the problem into two sub-problems:
  1) For a given set of segments $S\downarrow i$, how to calculate the shortest tour $T\downarrow S\downarrow i$
     - Is equivalent to **Euclidean TSP** problem which NP-Hard.
  2) How to partition the set S into k different subsets of $\{S\downarrow 1 , S\downarrow 2$

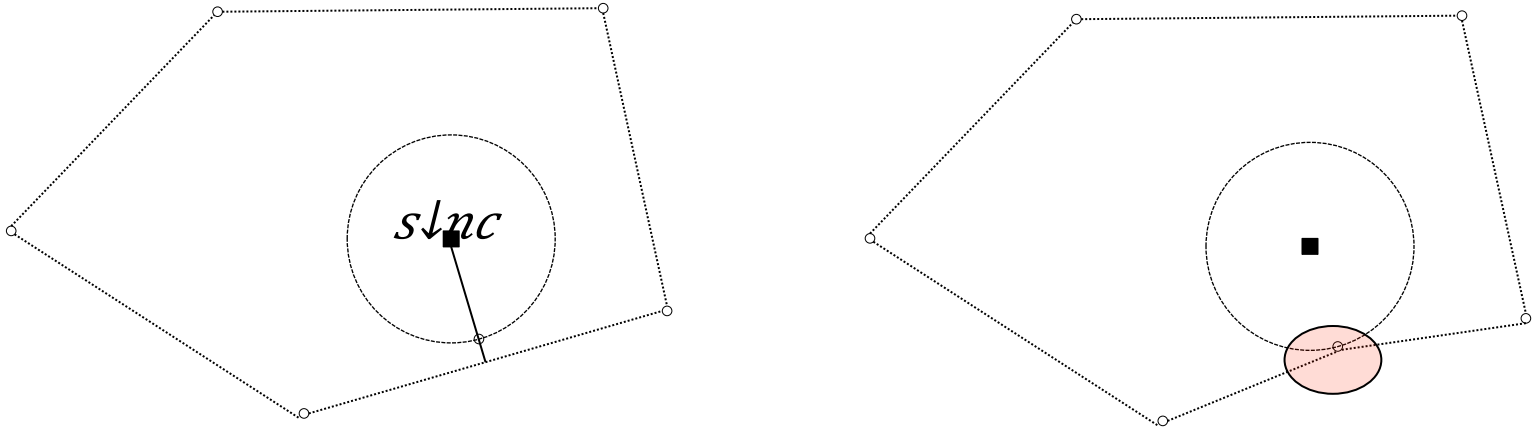
1st Problem: Optimized Tour Computation

- Let s_i be the point where ith segment representative is located. The point is called a **collection point of** s_i if $d(s_i, p_i) \leq R$ where $d(s_i, p_i)$ is the Euclidean distance between s_i and p_i, and R is the communication range of a sensor.
- Given a set of *m* representatives $S = \{s_1, s_2, ..., s_m\}$ a Tour is the shortest possible cycle that visits all collection points p_i of s_i where $1 \leq i \leq m$
- For $m = 2$, the collection points are the intersections of transmission disks with the line passing through representatives
- For $m > 2$:



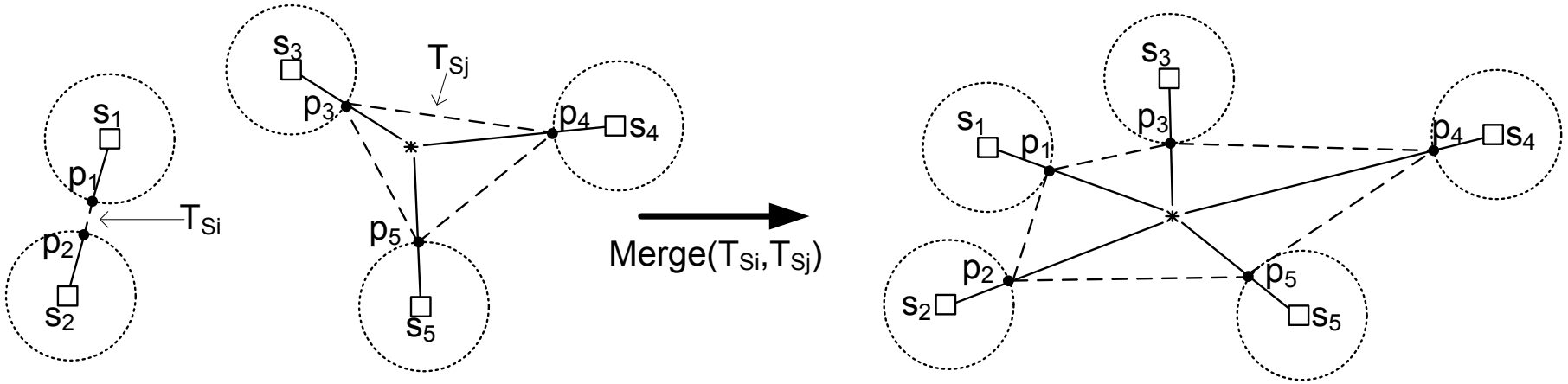
Handling Non-Convex Points

- Collection point of a non-convex representative $s{\downarrow}nc$ is the intersection of the transmission disk of $s{\downarrow}nc$ with the perpendicular from $s{\downarrow}nc$ to the closest tour edge



IDS-*k*MDC Heuristic in Detail

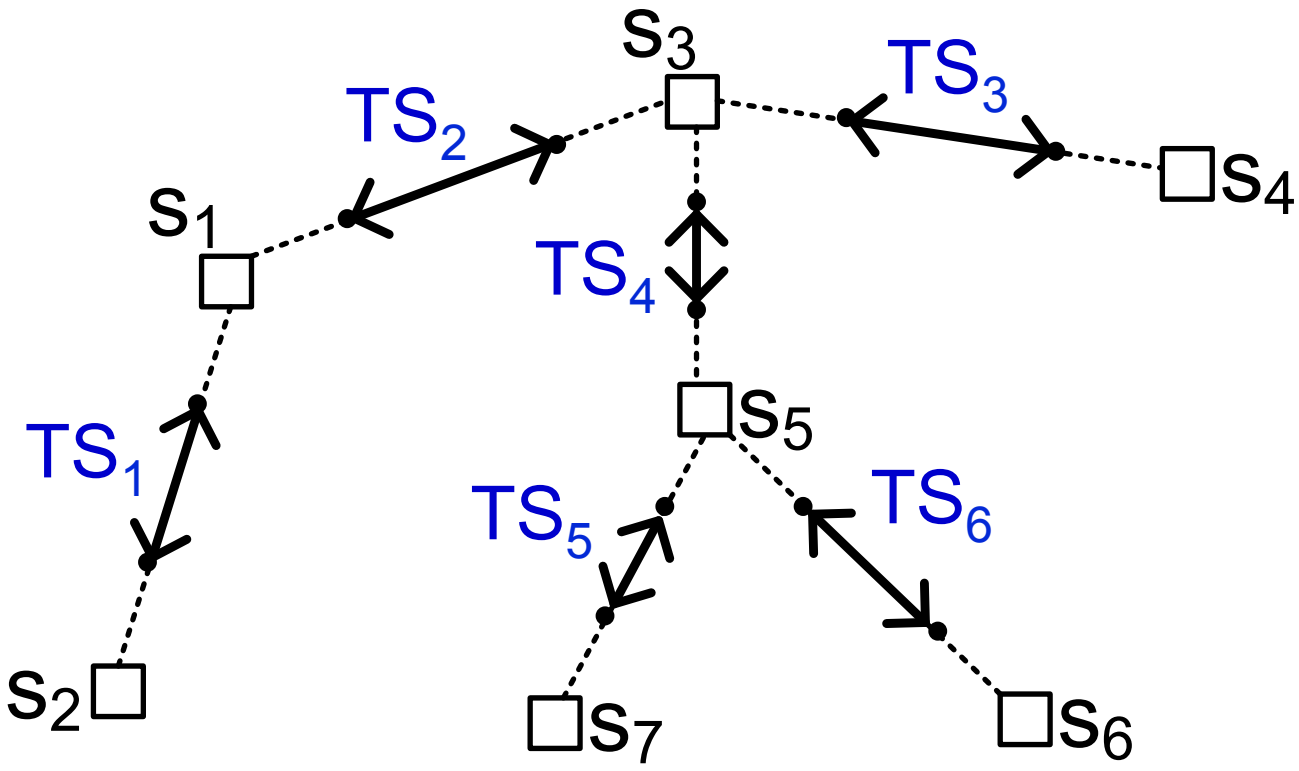
- **Initialization**:
  - Calculate minimum spanning tree (*mst*) of representatives
  - Assign one mobile node for each *mst* edge and form the tour for two segments
- **Idea**:
  - Iteratively reduce the number of mobile nodes by merging two tours into one tour
  - Continue merging until number of available mobile nodes matches the tours
- **Merging the Tours**



Cost of merging = Length of new tour – (Sum of the two merged tours)

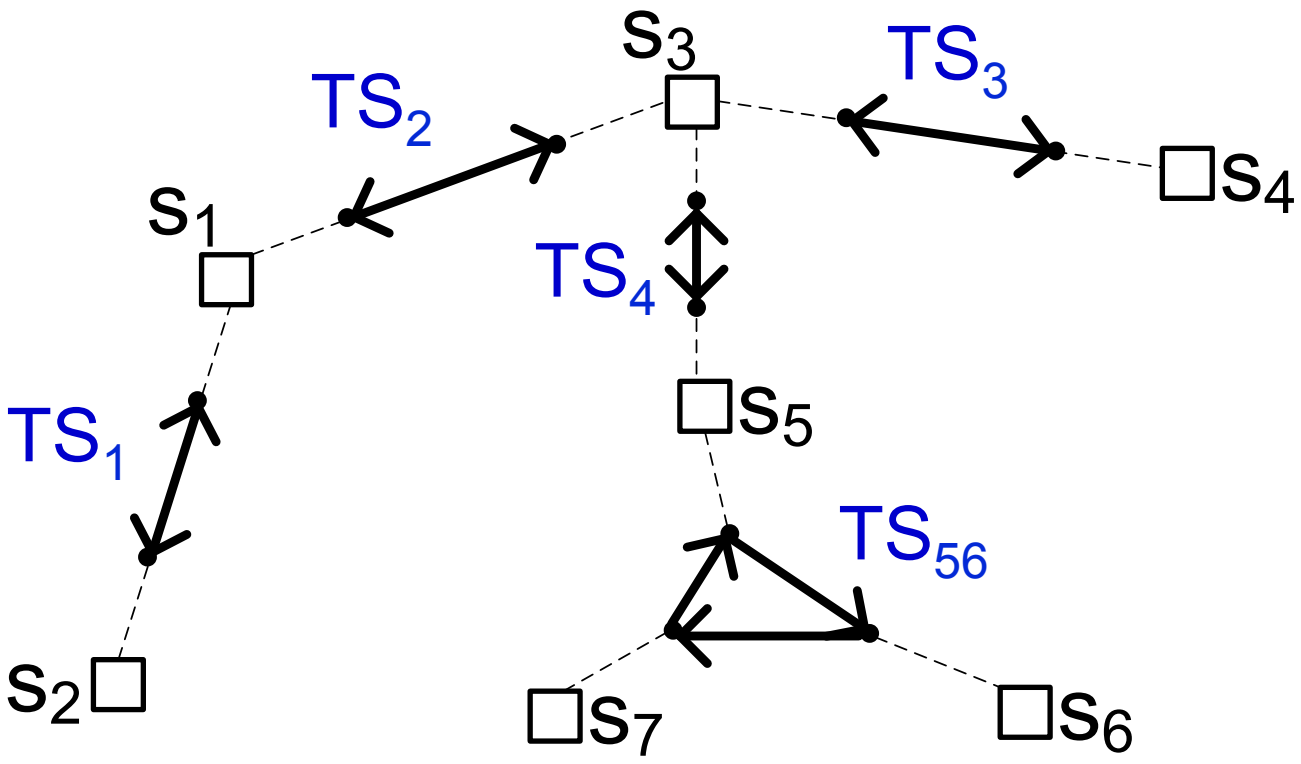
Illustrative Example

- Requirement: Federate 7 segments and 3 Mobile relays
- **Initialization**:
  - Calculate *mst* of representatives
  - Assign one mobile node for each *mst* edge and calculate the tour for two representatives



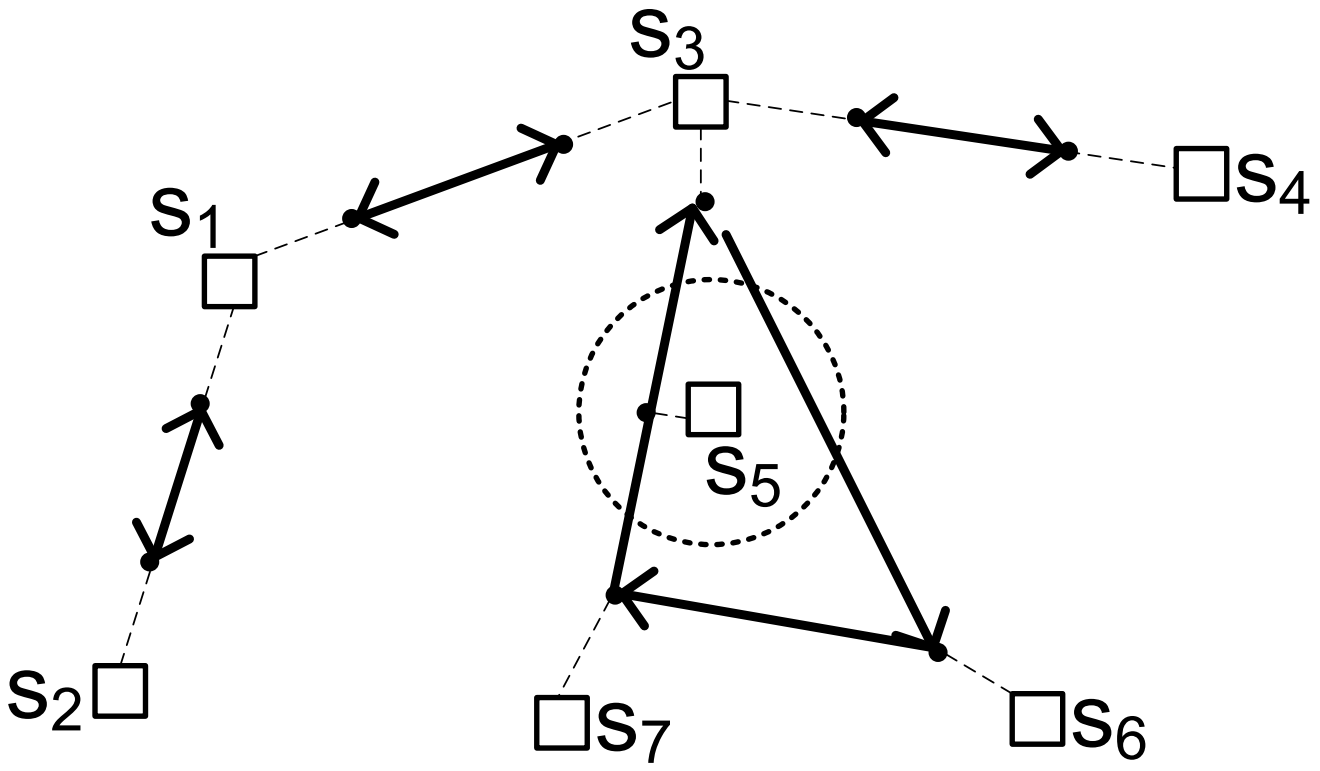
Ilustraive Example

- **Merging**: $TS_5 = \{S_5, S_6\}$ and $TS_6 = \{S_5, S_7\}$
  - Merging Cost= $(|TS_5|+|TS_6|)-|TS_{56}|$



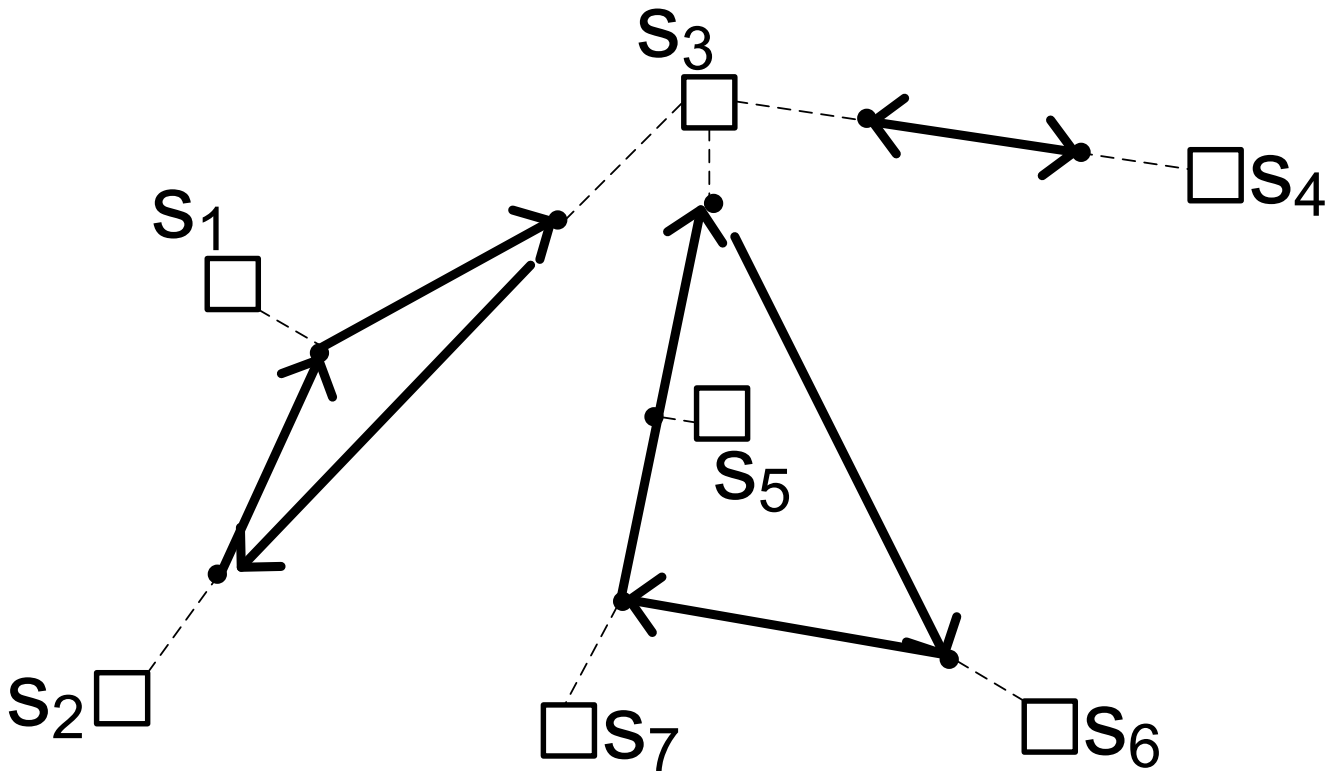
Illustrative Example

- S_5 is a Non-Convex segment



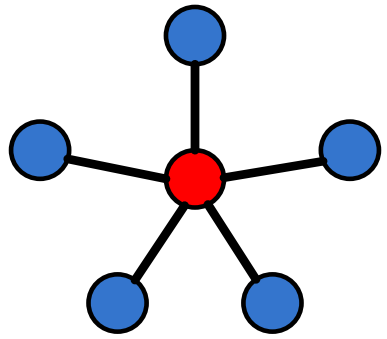
Illustrative Example

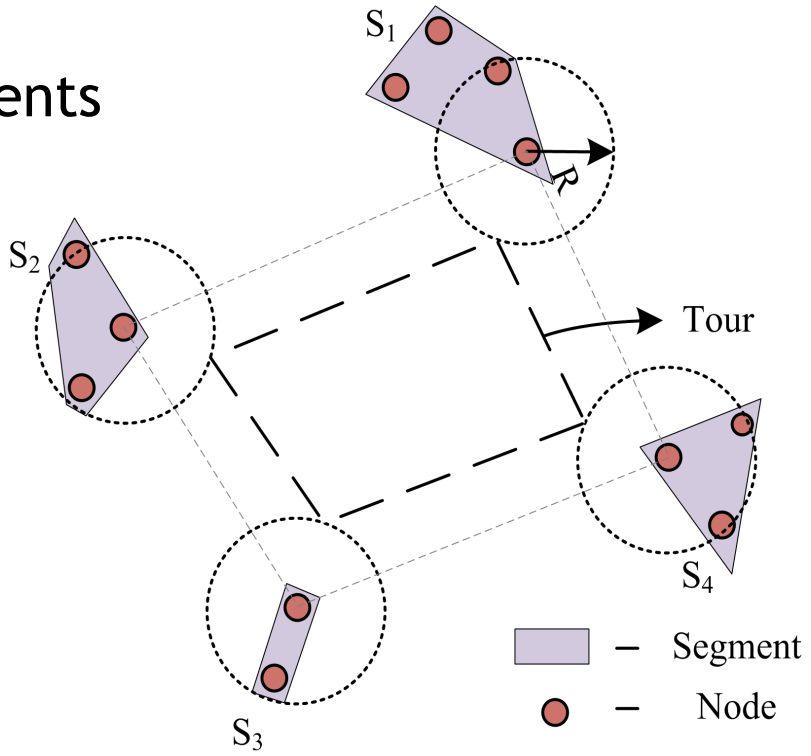
- Final Iteration



Mobile Relays Based Federation with Reduced-Latency

- Relay count is insufficient to establish stable inter-segment topology
- Communication traffic is delay sensitive
  ➔ Optimizing the data delivery in the federated network
- Approach: Touring of Clustered Segments (ToCS) to reduce the maximum inter-segment delay
- ToCS Operates in two phases:
  1. Group segments into clusters to form star-shaped inter-cluster topology
  2. Optimize clusters boundaries to facilitate for tour synchronization of individual mobile relays



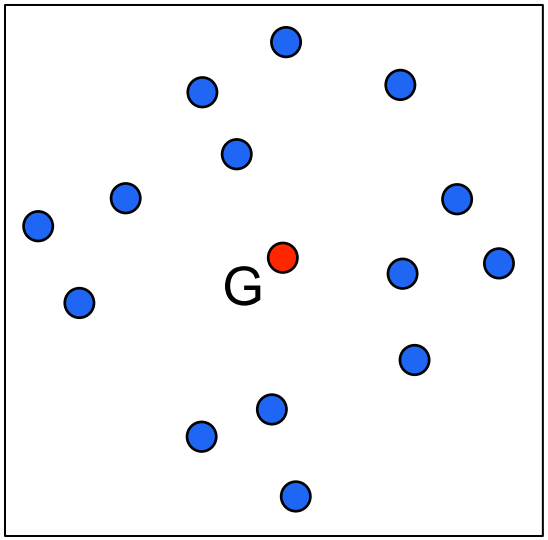


Star Clustered Inter-segments Topology

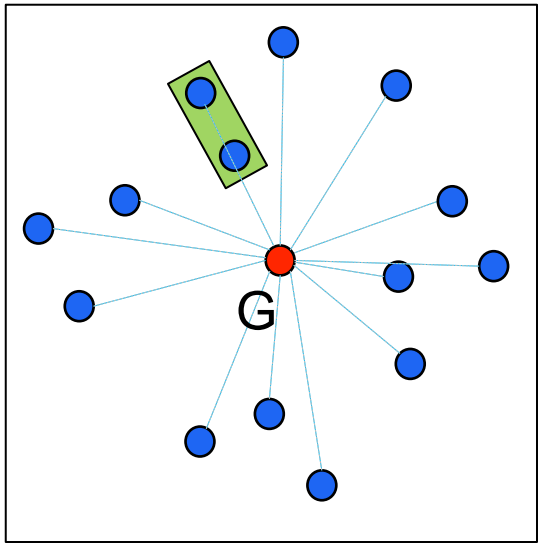
- Segments are grouped into *K* clusters, matching the available mobile relays
- To form a star inter-cluster topology:
  - The centroid "G" of the entire network is considered as a segment to denote the central cluster
  - The central cluster is kept fixed through the clustering process
  - The segments are thus grouped to *K-1* clusters.
- The ToCS clustering algorithm runs in rounds:
  - Initially, each segment forms an independent cluster with the centroid
  - In each round, the two clusters (C_i and C_j) with the least merging cost are combined together
  - Merging cost of C_i and C_j : f_{ij} = *Tour*(C_i+C_j+*Centroid*) - *Tour*(C_i+*Centroid*)
  - At the end of each round, the number of clusters is reduced by one
  - Process is repeated until forming *K-1*clusters (i.e., a max of *N-K-1* rounds, where N is the number or segments
  - The K^{th} cluster corresponds to the centroid with zero touring length.

Illustrative Example

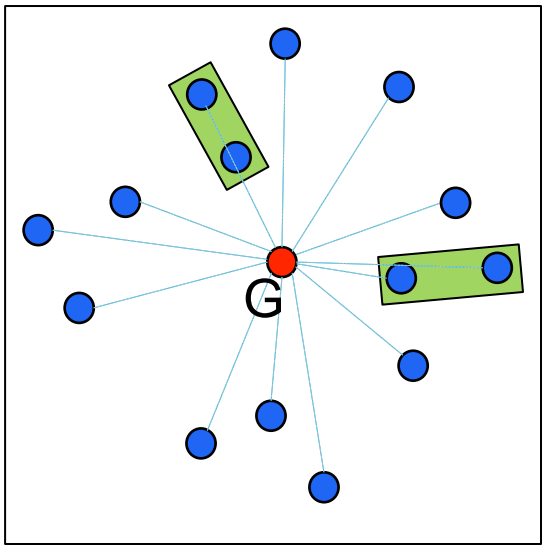
segments and centroid G form individual clusters



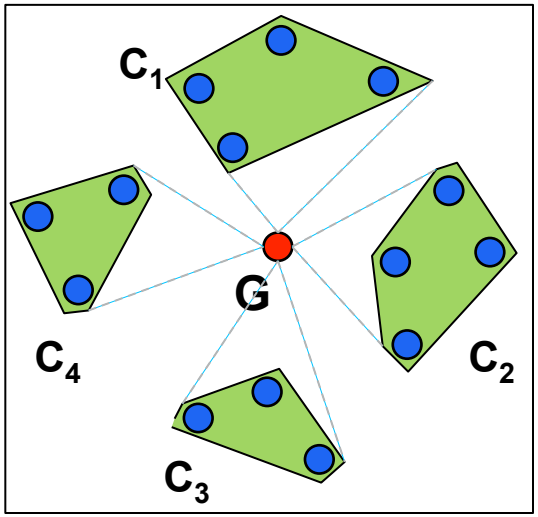
Combine two clusters with most merging gain



Two more clusters are combined

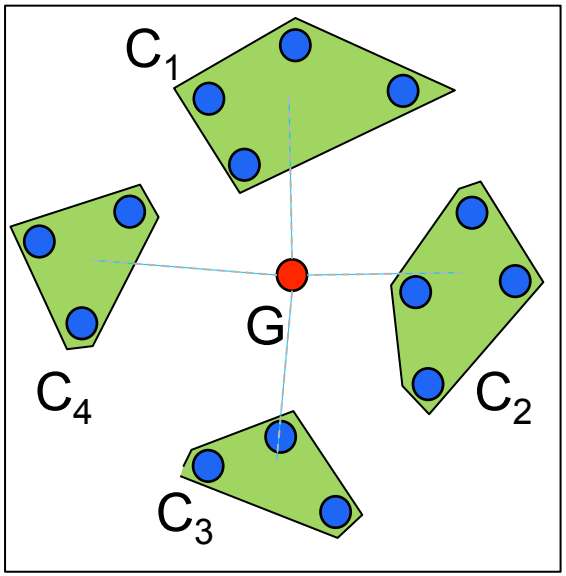


Repeat merging clusters for 7 more iterations

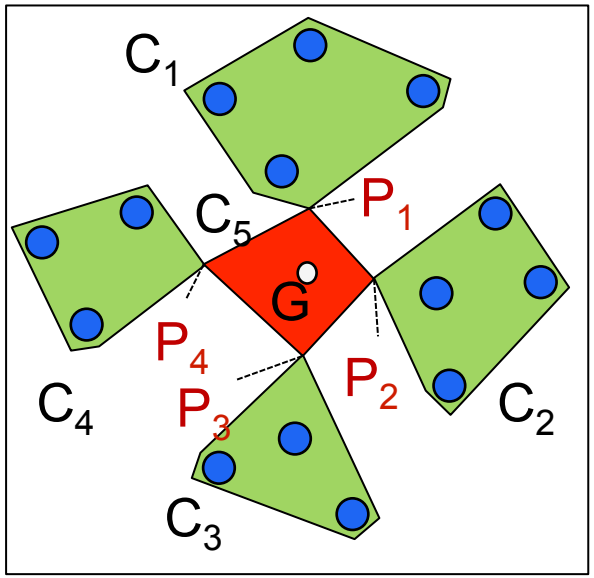


Total of 4 segment clusters and the centroid (for federation using 5 mobile relays)

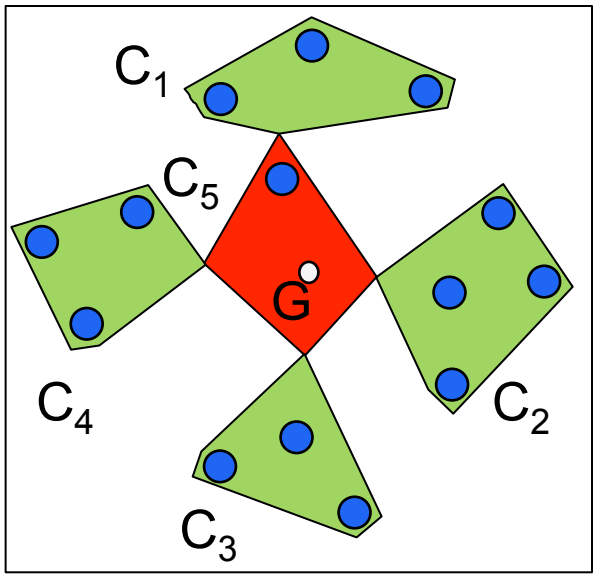
Optimizing Clusters for Synchronization



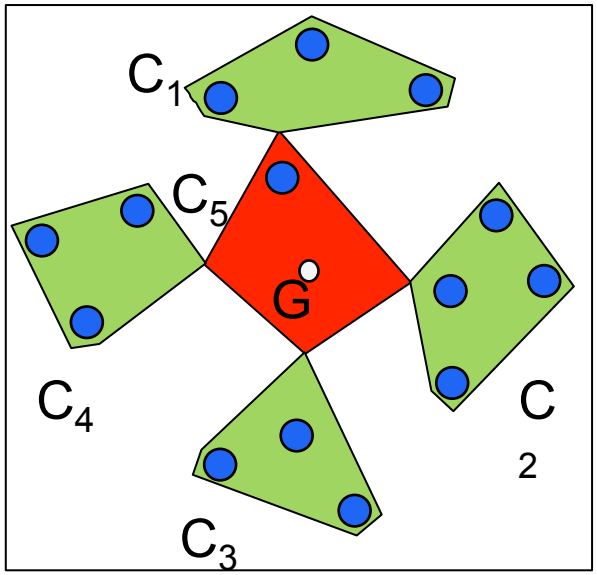
Determine rendezvous points for the MDCs based on the closest point on the individual tours to G



Adjust tours that exceed average by moving rendezvous points away from G



Segments may switch the centroid clusters as part of the tour length balancing

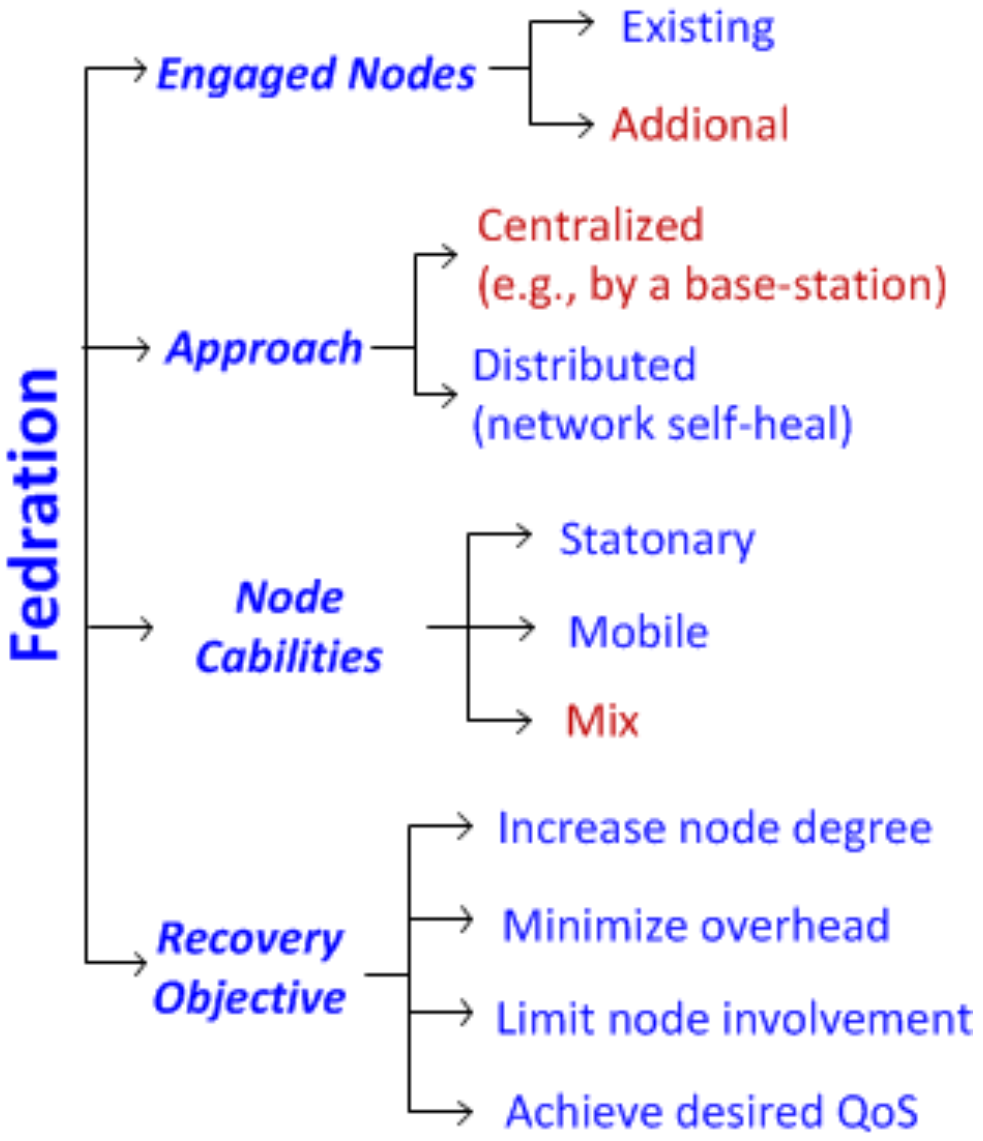


Repeat the tour length adjustment as needed

Sample Federation Approach

An example:

Mix of Mobile & Stationary nodes for Inter-connecting a set of segments (MiMSI)



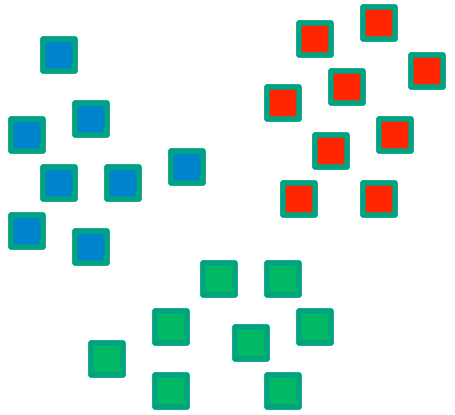
Mix of Mobile & Stationary nodes for Inter-connecting a set of segments (MiMSI)

- **Objective**: connect a set of *N* segments (terminals)
  - Using a mix of I_S stationary relays nodes & I_M mobile data collectors with $I_M \geq 1$
  - All $I_S + I_M$ nodes have the same communication range *R*
  - With the minimum total travel distance

- **MiMSI operates in three phases:**
  1. Generate a stable topology that is used as a baseline for optimization:
     a) Determine a set of Steiner points for inter-connecting the terminals using stationary node
     b) Group the Steiner points into clusters each is to be served by a mobile node
  2. Identify some of the RNs in the baseline topology that minimize the travel distance for the mobile nodes (gateways between clusters)
  3. Define tours for the mobile nodes that involve the least travel distances

1. Form a Stable Inter-Segment Topology

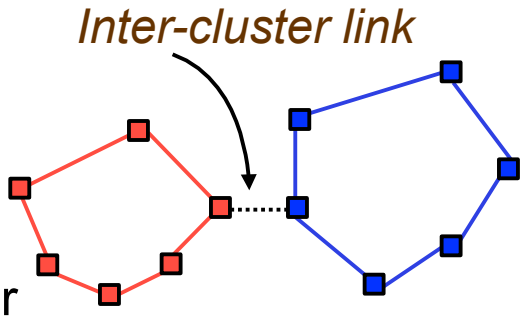
- Solve the federation problem using stationary relays
  - Employ one of the numerous polynomial time heuristics for solving the SMT-MSPBEL problem in the literature
  - Help in identifying positions for some, or all, of the l_S RNs in the final interconnected system.
- Clustering: groups nodes (Steiner points and segments) based on proximity
  - Using one of the many proximity-based clustering algorithms in the literature, e.g. K-means
  - # clusters = # available mobile nodes, l_M
  - Having high node degree in the set helps clustering
  - More gateways between the clusters
  - ➔ enhances the solution achieved by MiMSI



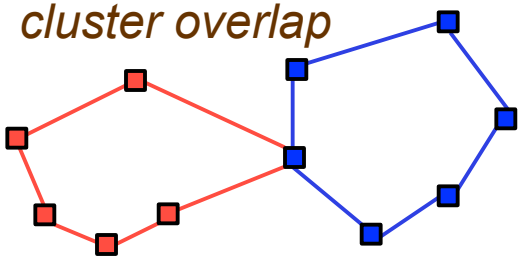
- Why proximity-based clustering ?
  - To group a set of segments that are close to each other
  - Mobile nodes tour them with the least travel distance

2. Position Available Stationary Relays

- Identify the gateway nodes that connect two or more clusters
  - Form a minimum spanning tree on the set of Steiner points and terminals
  - Select shortest edges connecting pairs of clusters
  - Unselected Steiner points are not considered any further

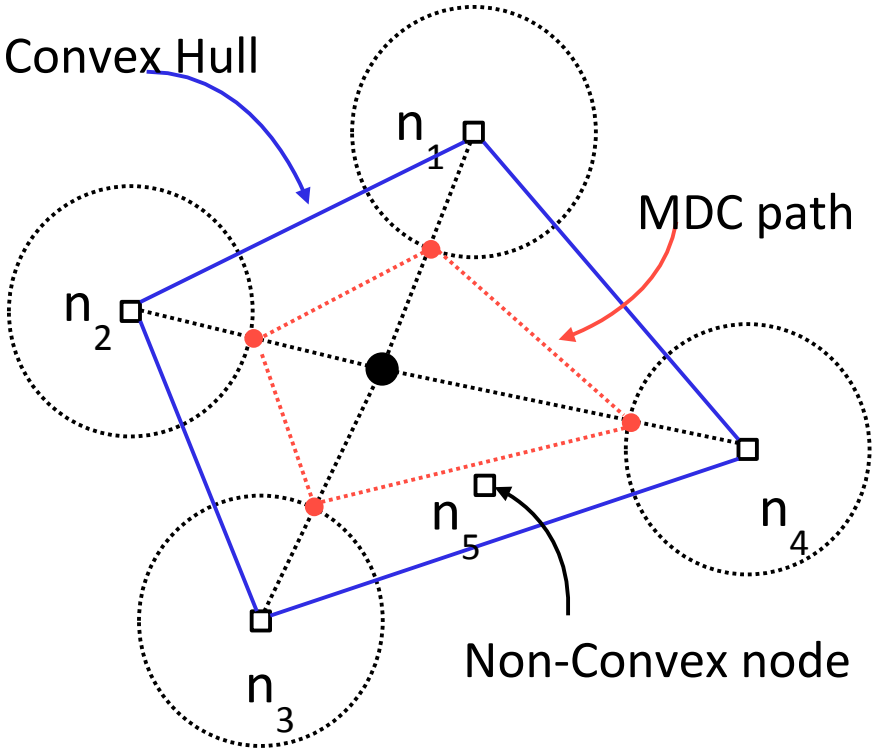


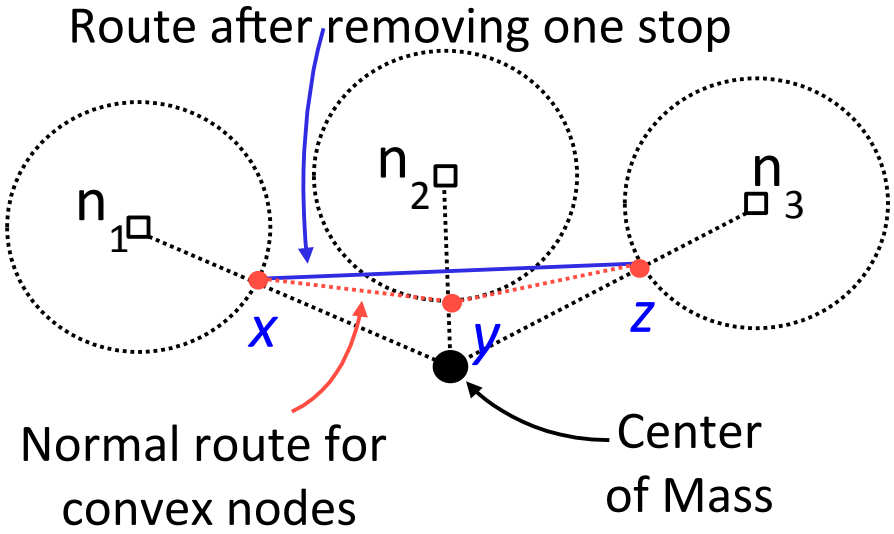
- Approach depends on Depending on the relationship between the number of gateway Steiner point "g" and l_S, 3 different cases will result
  - g = l_S : simplest case
    - Relay are positioned at the picked Steiner points to serve as inter-cluster gateways
  - g < l_S : plenty of stationary relays
    - Populate all "g" gateway Steiner points with stationary relays.
    - Remaining (g - l_S) relays reduce the travel paths of the mobile nodes in 3rd phase
  - g > l_S : insufficient stationary relay count
    - Some clusters need to overlap



3. Determining the MDC Tours

- Each of the l_M clusters contains a subset of the N segments & possibly one or multiple relays that serve as gateways to other clusters
  - These segments and relays are considered stops on the MDC tour for the cluster

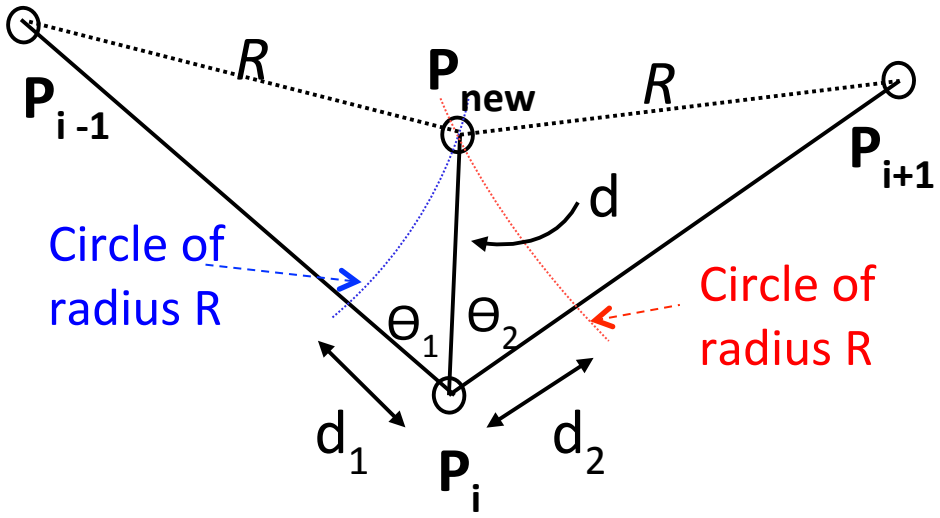




- MDC tour is formed like explained earlier with some additional optimization

Utilizing Extra Stationary Relays ($g < l_s$)

- The optimization simply tries to place a relay in an inner point to the tour while being in range of a convex stop so that the travel path is shortened



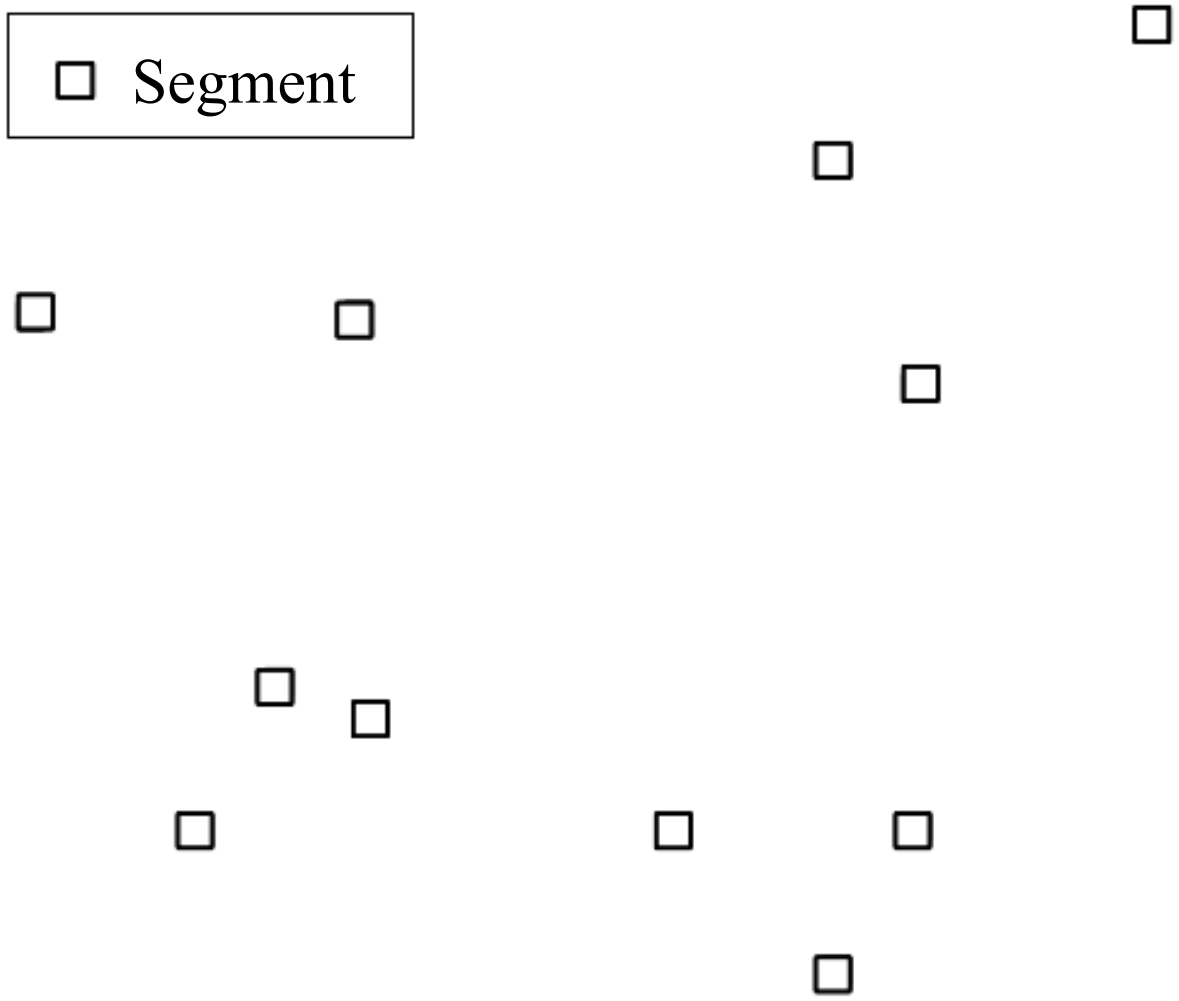
- Consider a convex polygon that has n corners point $\{P_1, P_2, ..., P_n\}$.
- The distance between any two points of $\{P_1, P_2, ..., P_n\} > R$
- Replacing one of the points P_i with a new point P_{new} that is at most R units away from P_i, will achieve the maximum reduction in the perimeter of the polygon if:

  (1) P_i is the corner with the smallest angle θ_{min} in the polygon

  (2) P_{new} is located on the bisector of θ_{min} at a distance R from P_i.

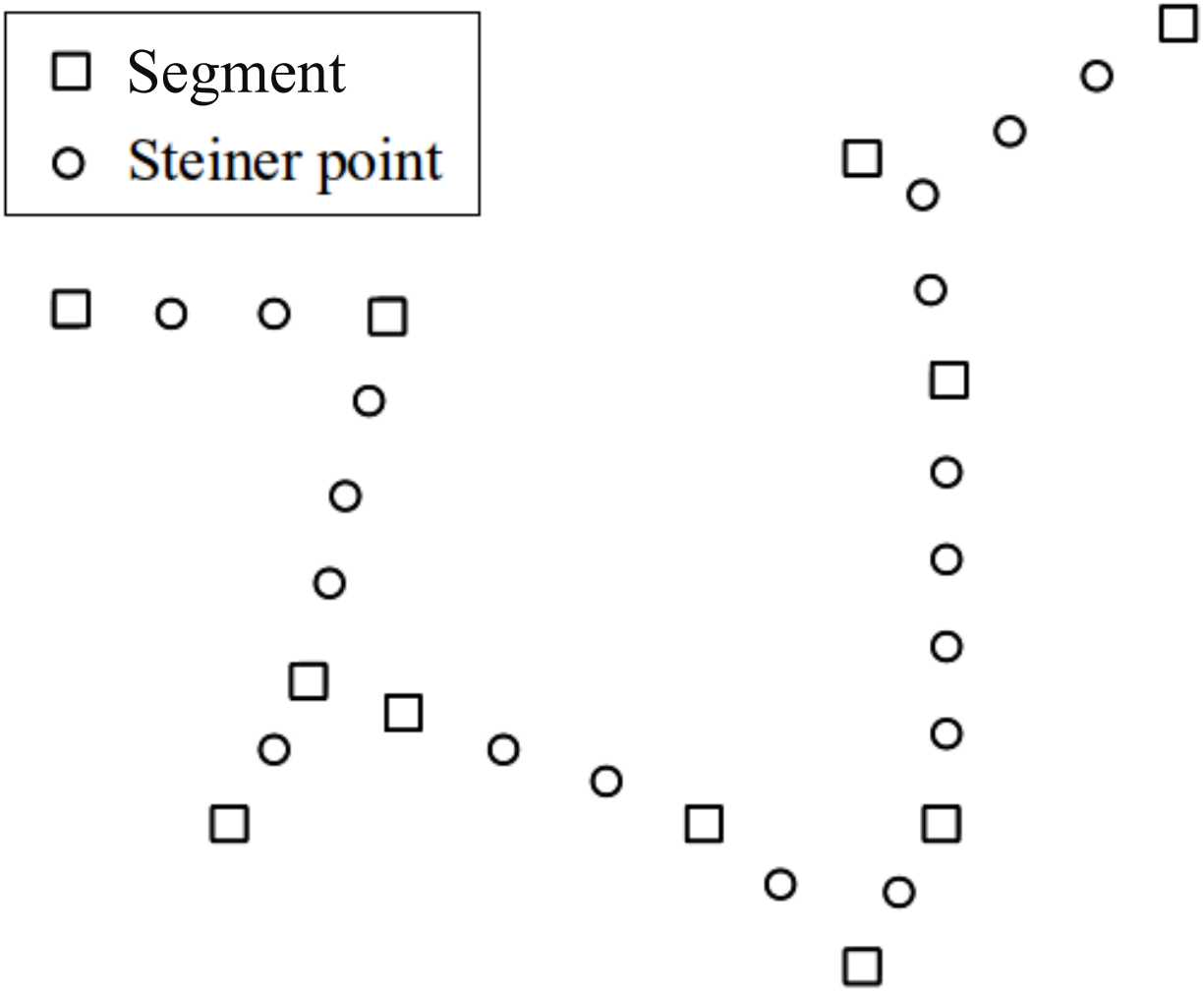
Example

11 segments (terminals) are to be interconnected using 3 mobile and 2 stationary relays



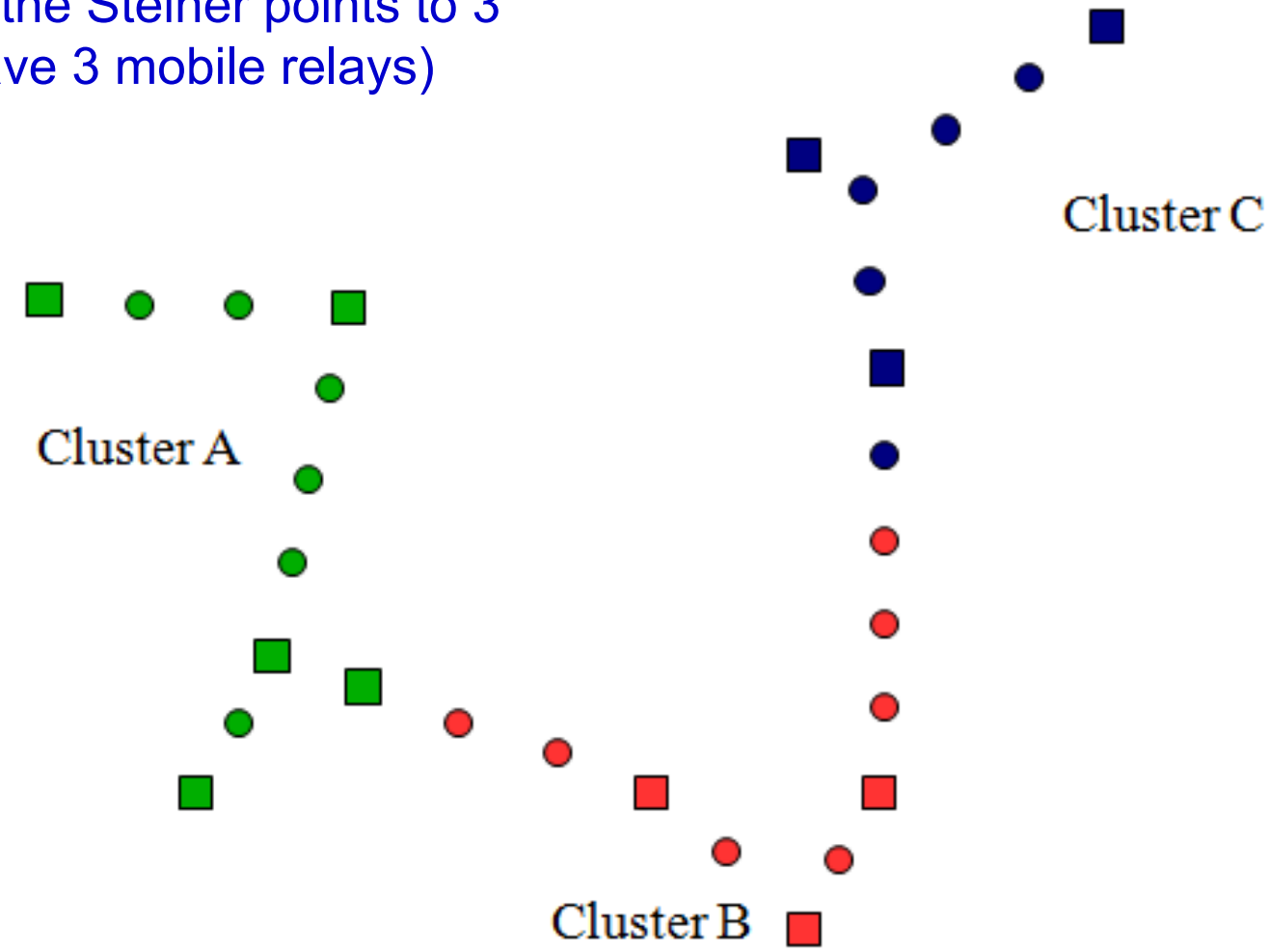
Example (cont.)

SMT-MSPBEL results in 18 Steiner points.



Example (cont.)

Grouping segments the Steiner points to 3 clusters (since we have 3 mobile relays)

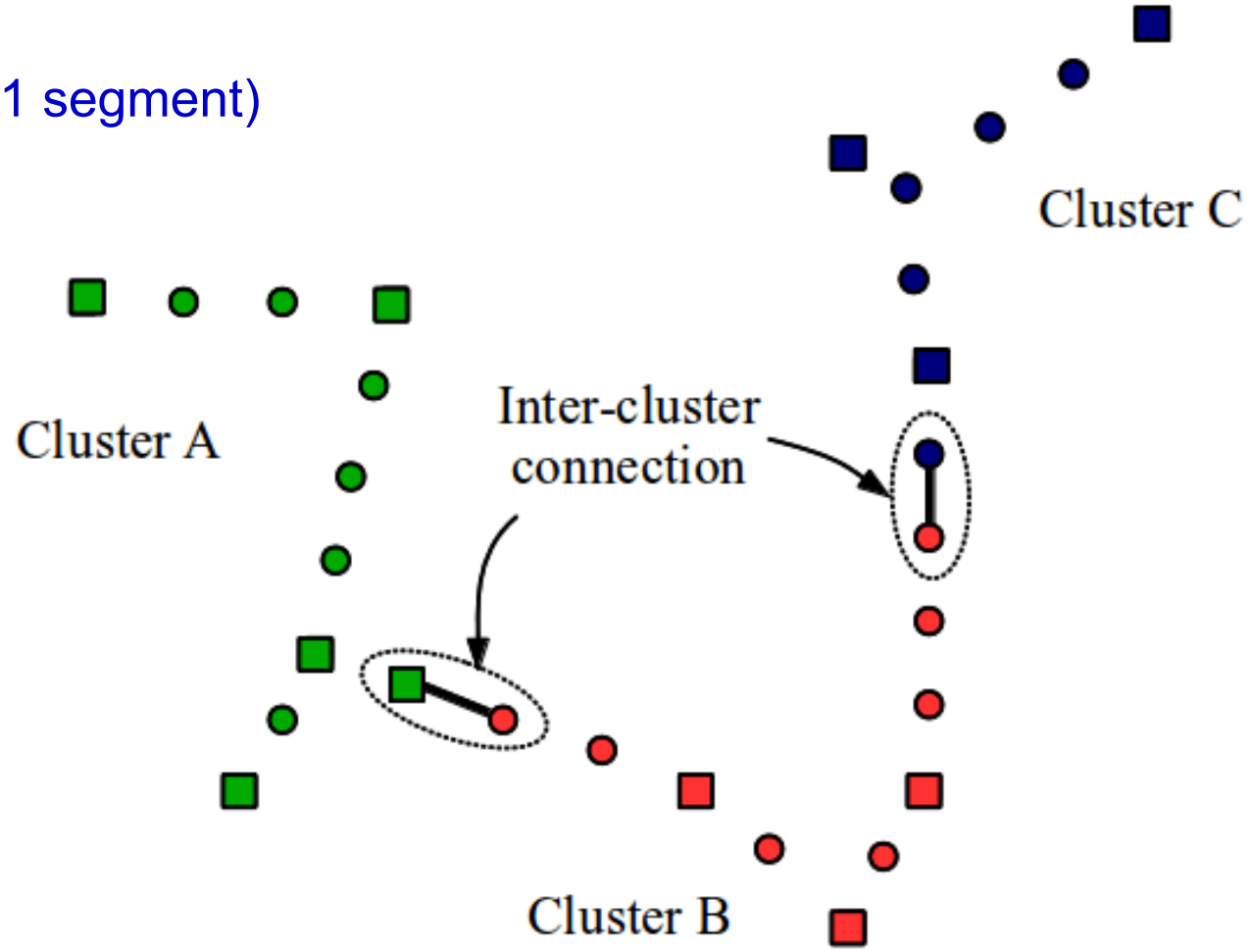


Example is based on using Kmean++ algorithm

Example (cont.)

Gateways that connect pairs of clusters are selected

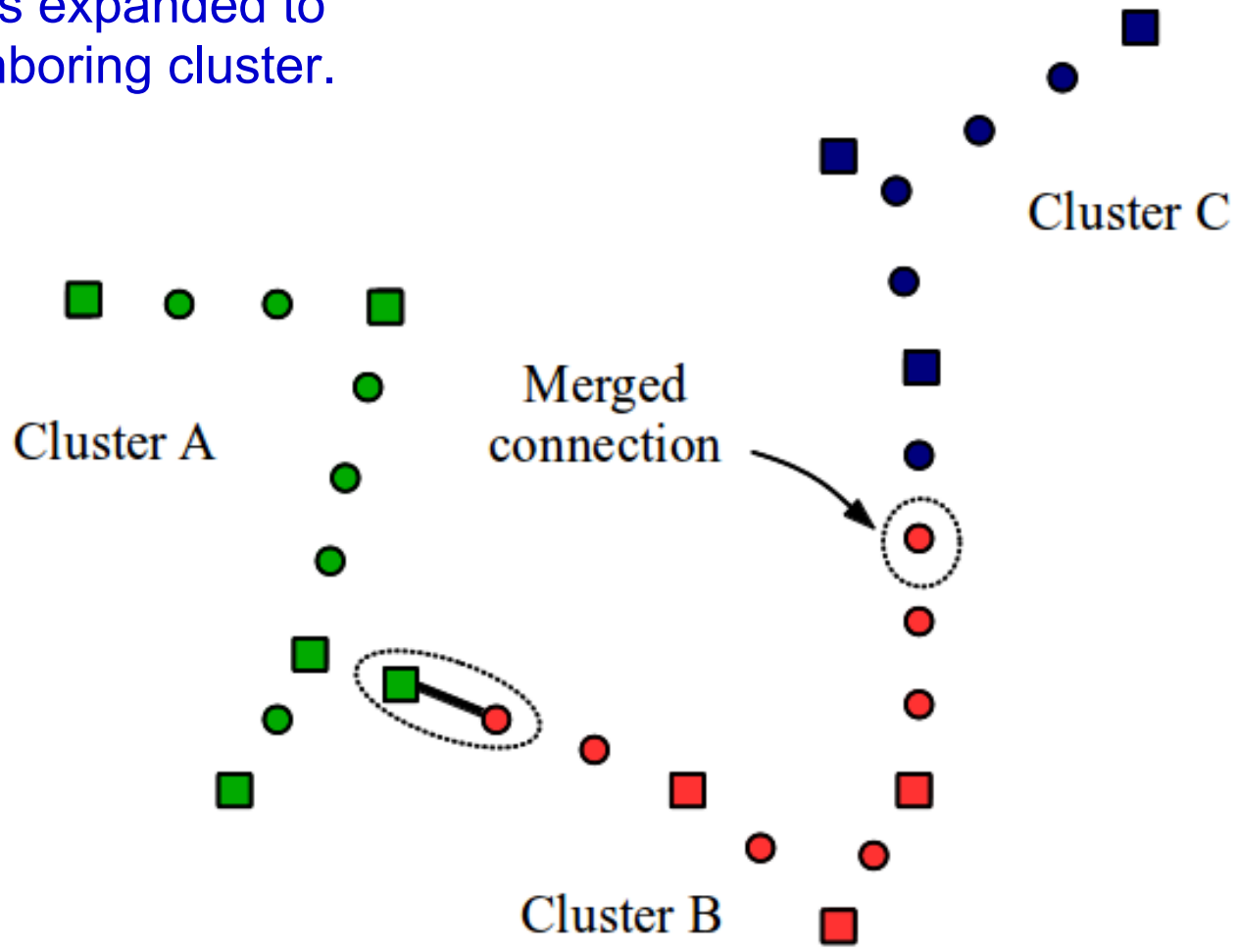
(3 Steiner points and 1 segment)



Example (cont.)

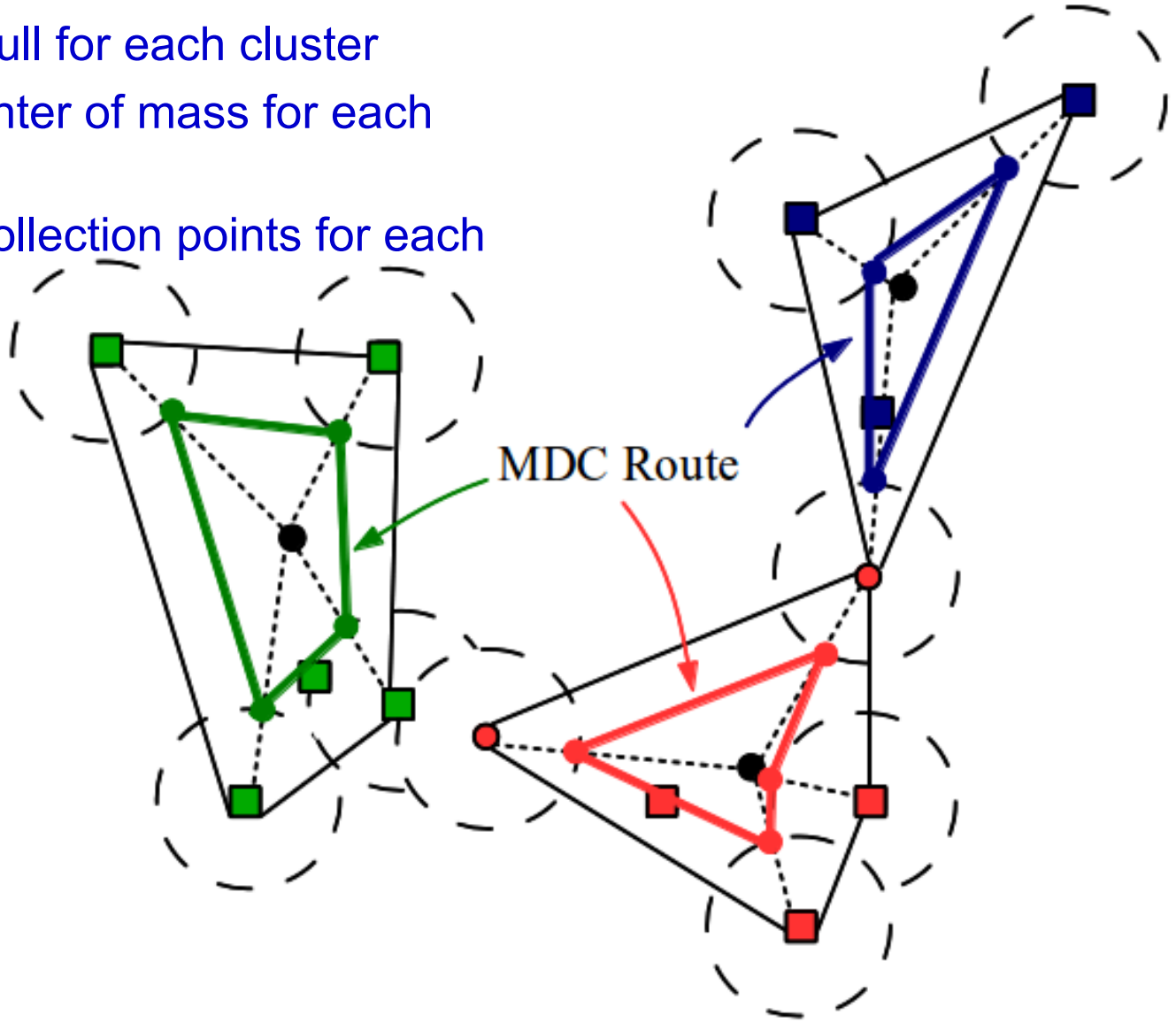
Only 2 available stationary relays

➔ one of the clusters is expanded to overlap with a neighboring cluster.



Example (cont.)

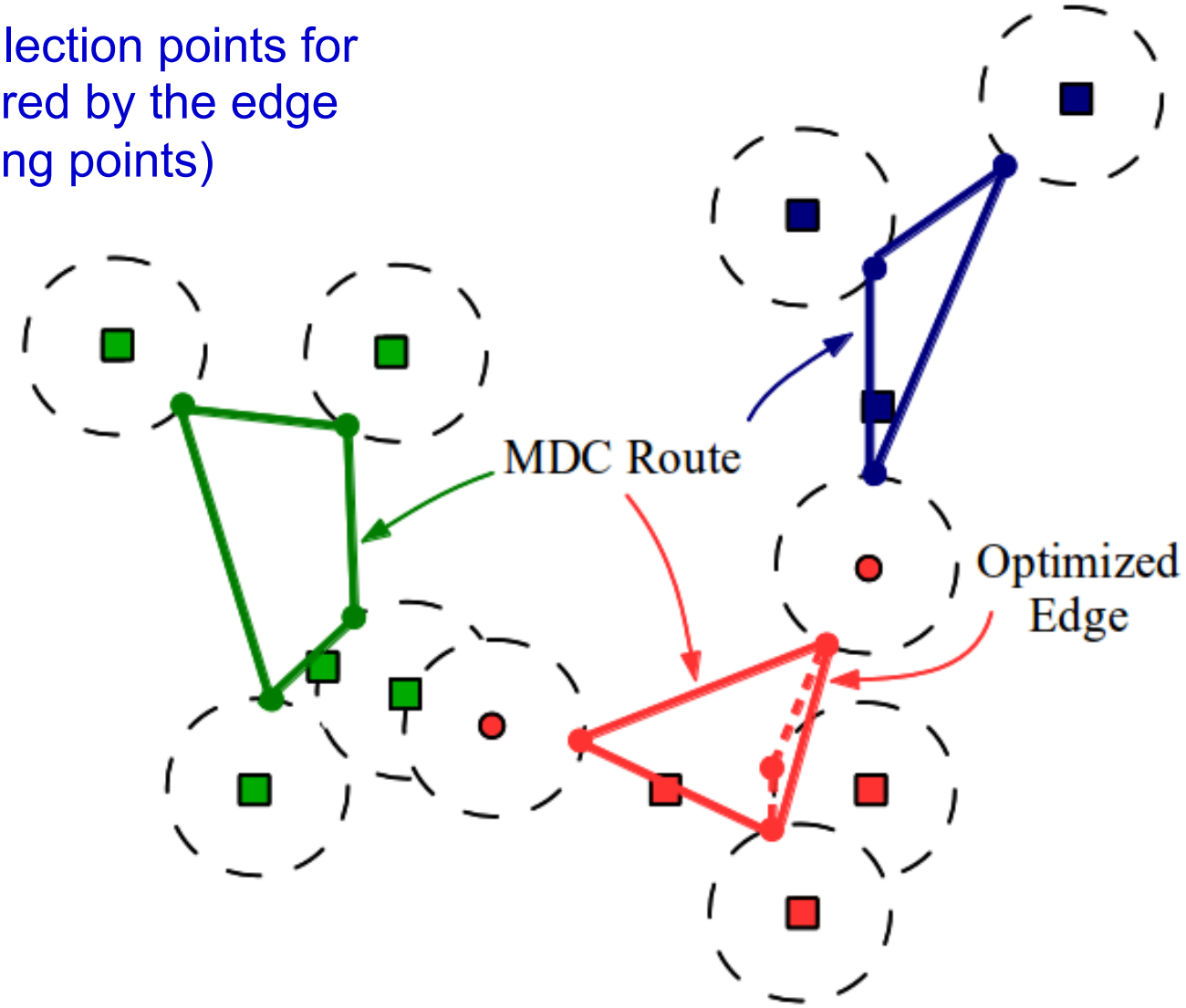
1. Forming convex hull for each cluster
2. Identifying the center of mass for each cluster
3. Determining the collection points for each convex node



Example (cont.)

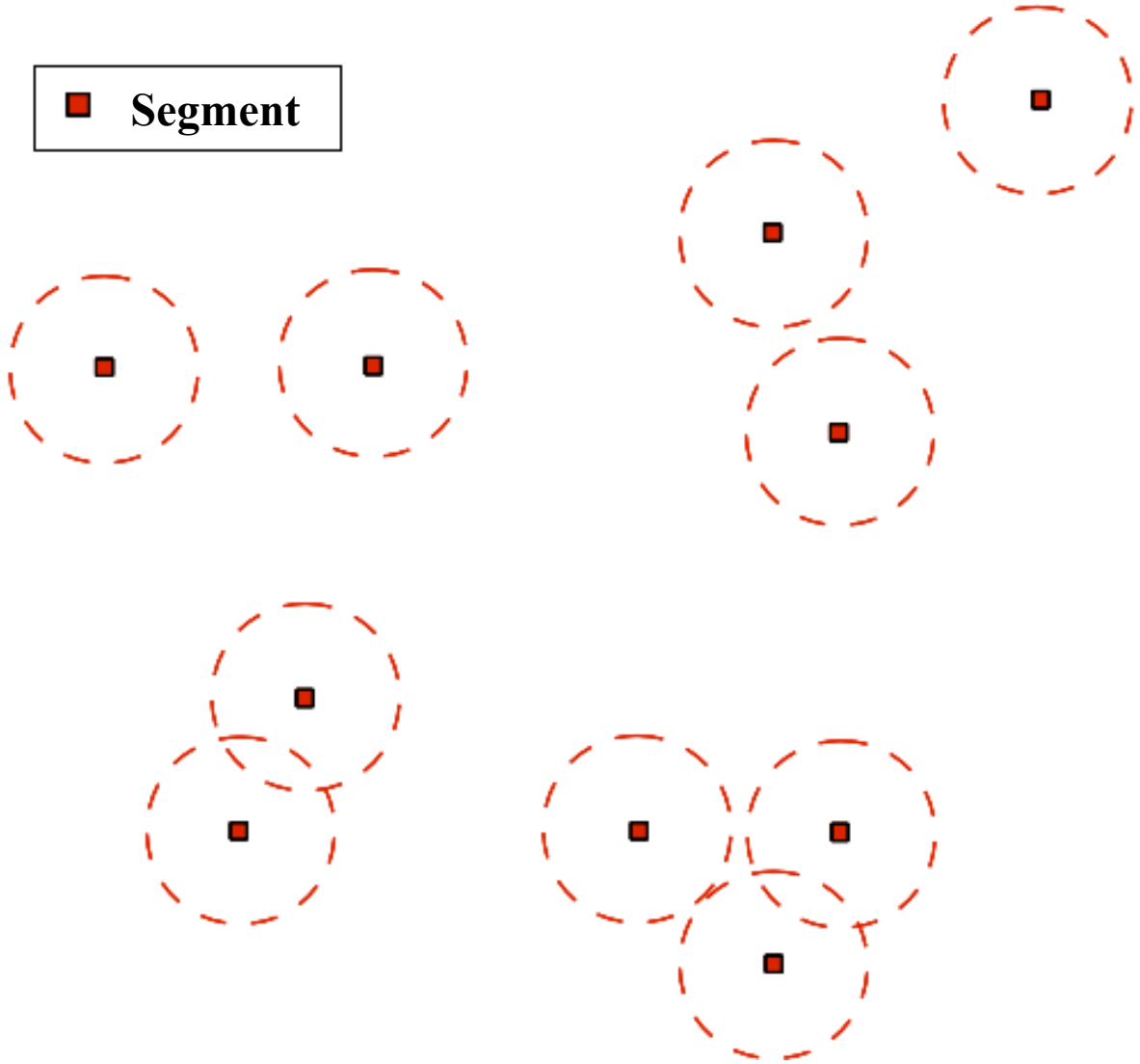
Edges Optimization:

removing one of the collection points for and convex node (covered by the edge formed by other collecting points)



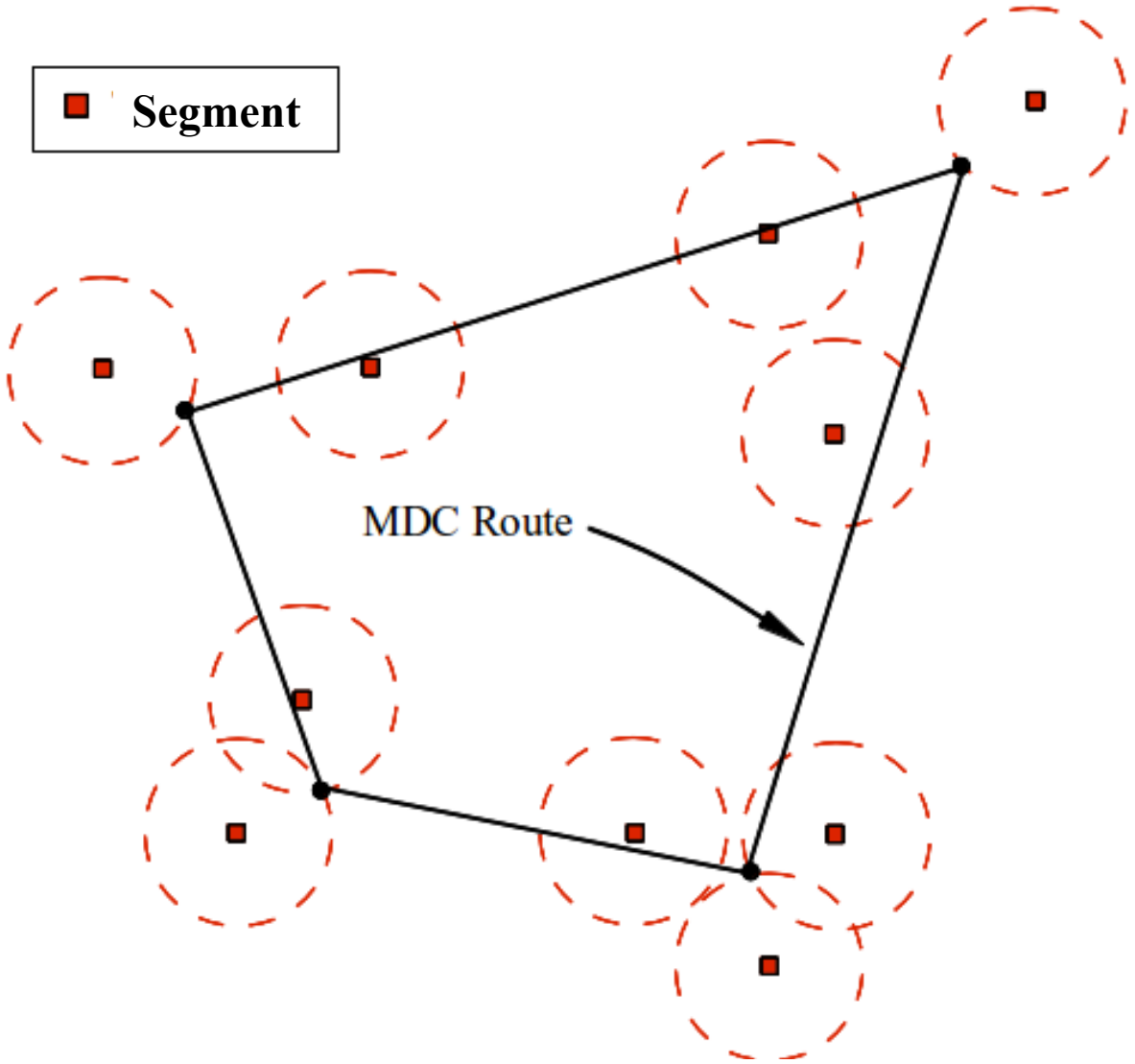
Special Case: Federation using 1 MDC

- No clustering is warranted
- Example: 15 Segments to be Interconnected using 1 mobile and 9 stationary relays



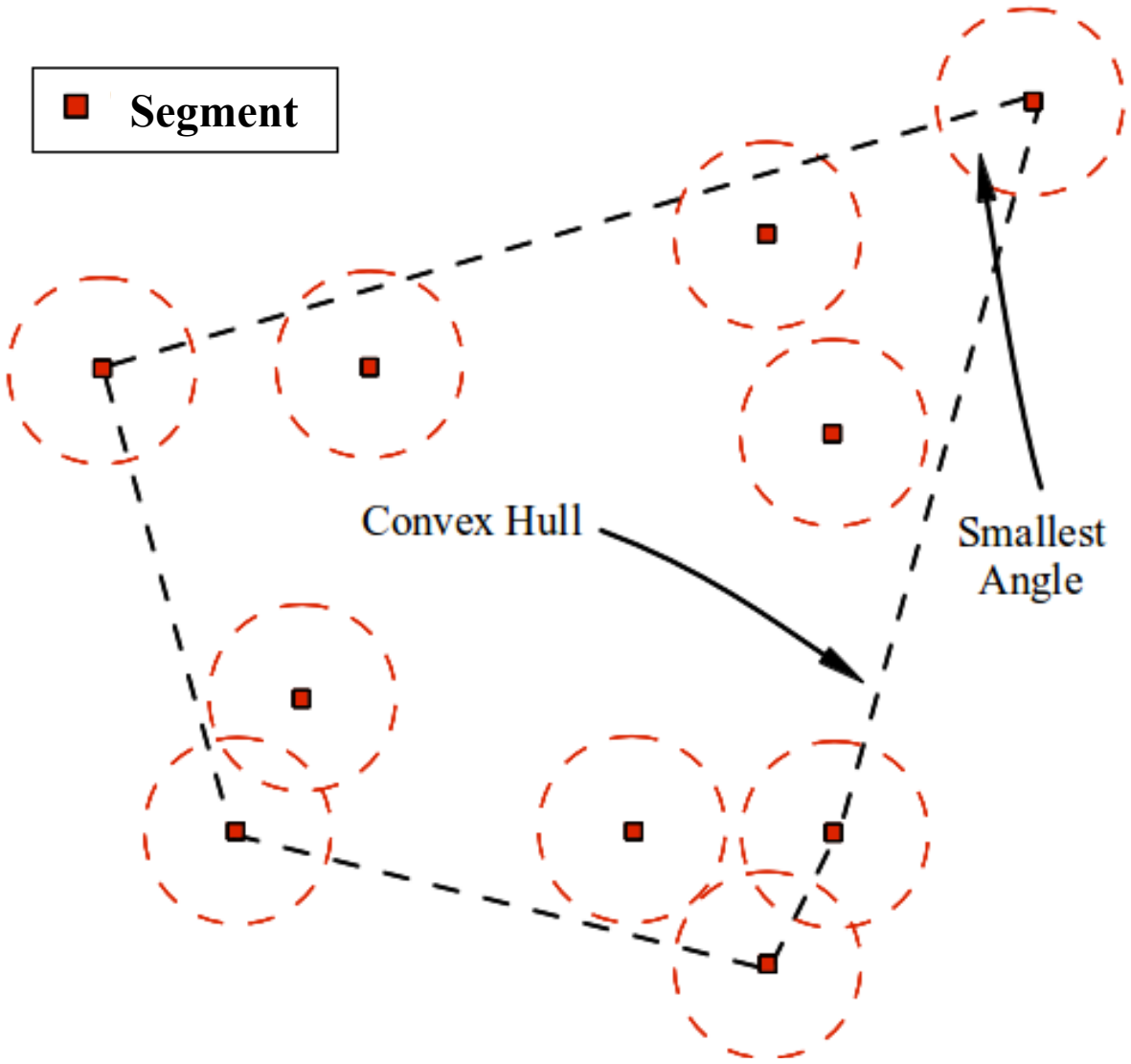
Example (cont.)

The MDC route if stationary nodes are not used



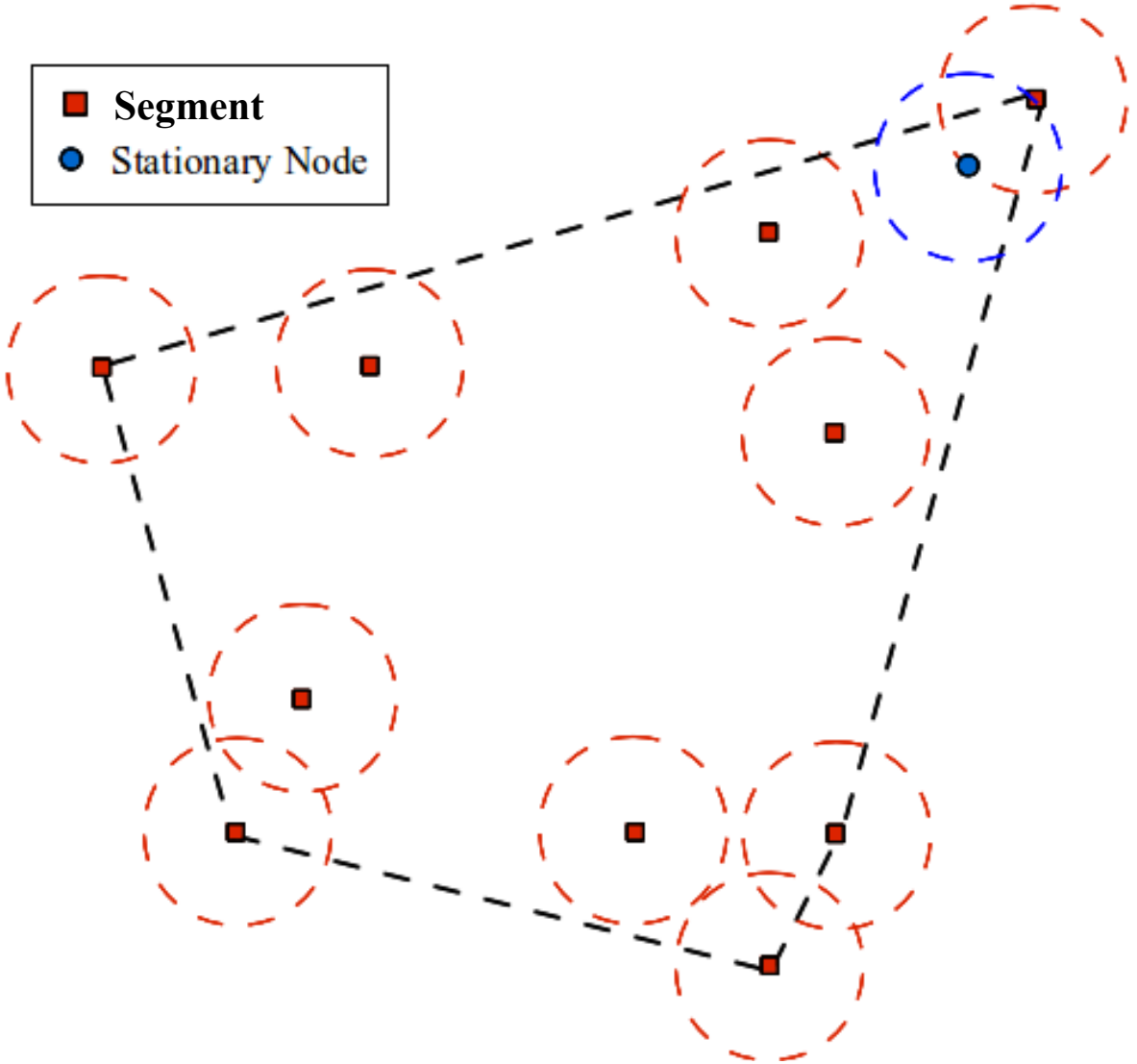
Example (cont.)

Forming the convex hull & Determining the smallest angle Θ_{min}



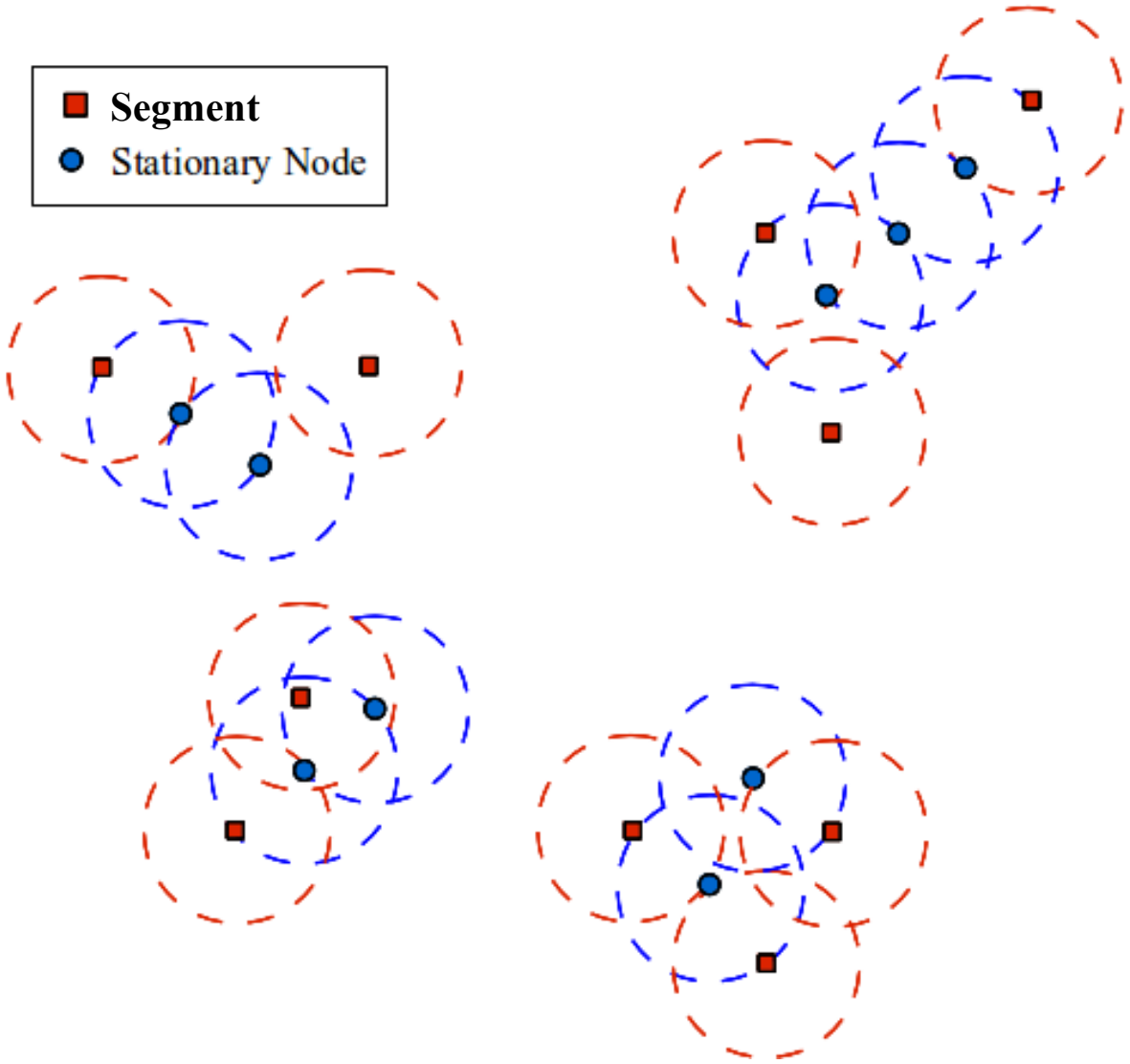
Example (cont.)

Placing the first stationary node at distance R on the bisector of Θ_{min} towards the centroid



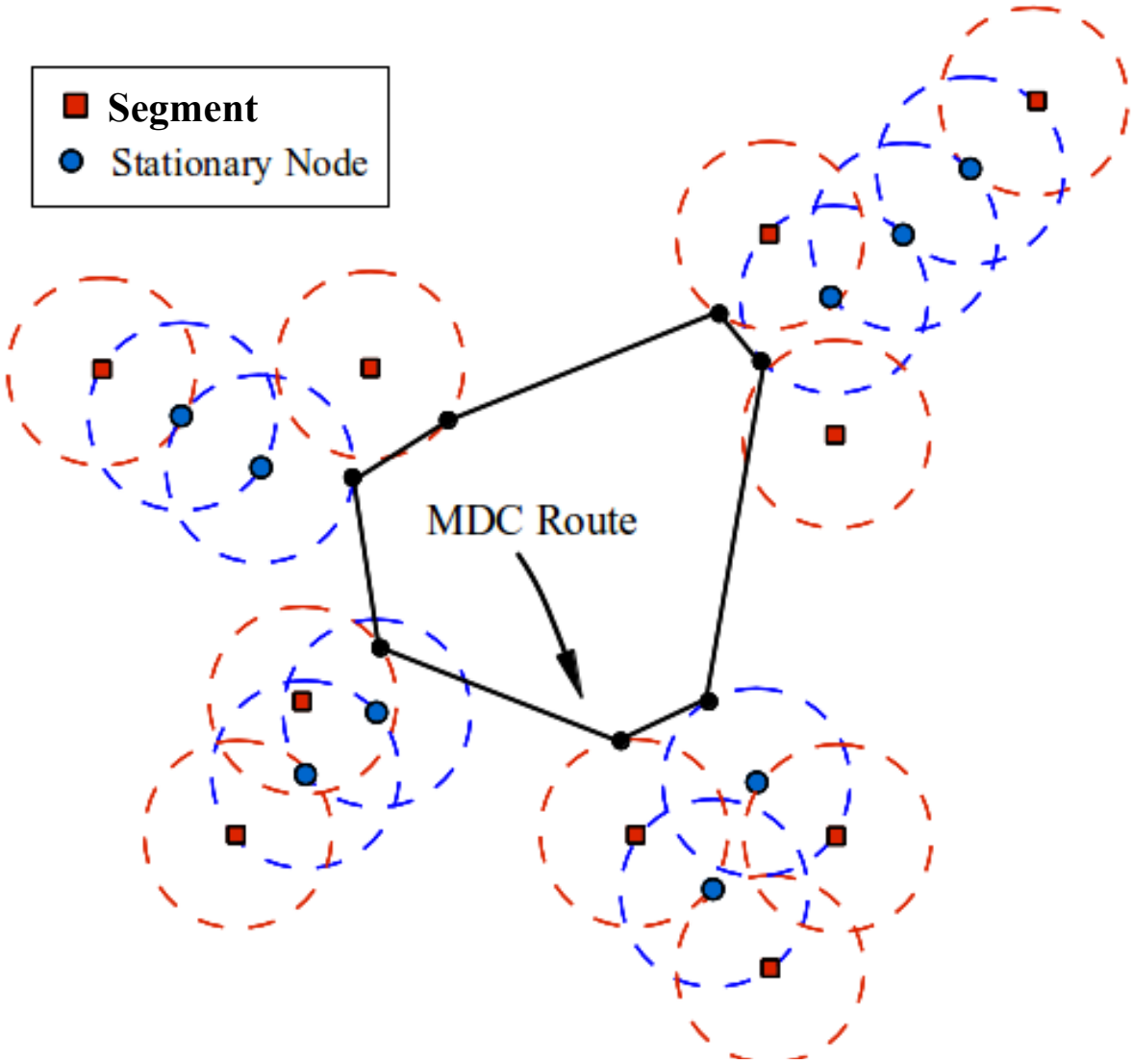
Example (cont.)

Last 2 steps are repeated until all 9 stationary nodes are placed

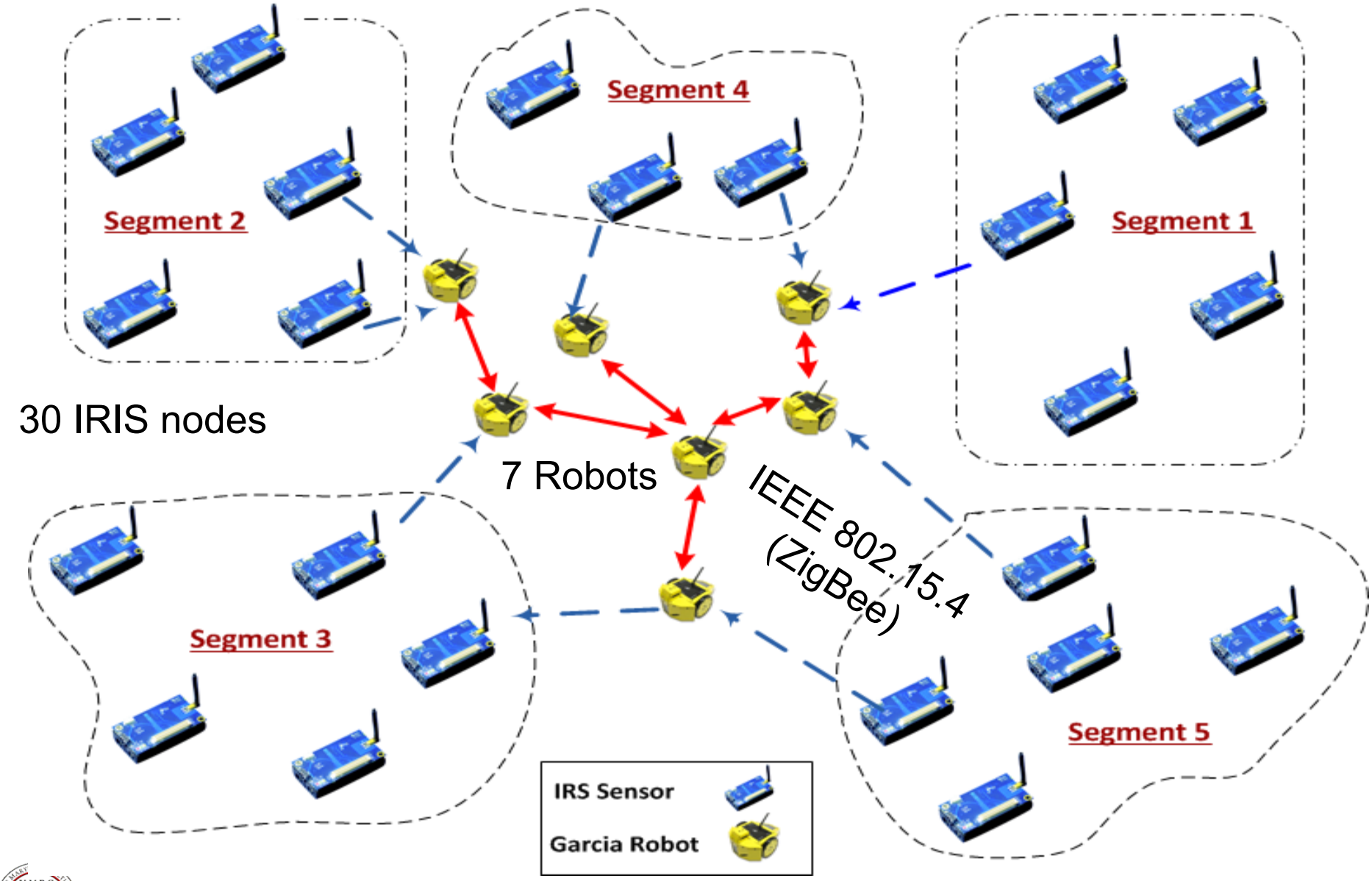


Example (cont.)

The last step is to determine the MDC path



Hardware Prototyping



Hardware Prototyping (Cont.)

- Planned on using Garcia robot and acquired one unit to try it out and develop software libraries
- A classroom kit of IRIS motes from MEMSIC Inc.
- The kit and the robot were for class projects in the wireless sensor networks class
- Recently a cost-effective solution is developed using iRobot Create (as an alternative mobile node)
- Since iRobot Create does not have radio, we developed solutions for integrating it with the IRIS mote.
- Used the cargo bay connector to interface serial port on the iRobot to that of the AVR microcontroller of IRIS node
- Such a configuration have made the IRobot a mobile IRIS node and enabled controlling the robot motion though the radio of the IRIS node
- Further, developed a printed circuit board to make the connection more rugged and professional.

iRobot Create

\+

IRIS mote

=

fully functional mobile sensor node

Conclusion

- In many WSN applications, nodes operates on small batteries and in harsh environments and becomes susceptible to failure
- Node failure can cause a major damage to network connectivity and prevent a wireless sensor network from functioning
- Approaches can be classified based on the recovery process into provisioned and reactive
- Reactive recovery is more resource efficient and is also valuable as an intermediate solution if the failure is significant
- Reactive recovery solutions can be further classified based on the number of failed nodes and the required state and analysis
- Similar scenarios arise in federation multiple stand alone networks in order to aggregate their capabilities to serve an emerging event
- Solutions differs bases on the node capabilities, available resources and the required topology features for serving the application needs
- Few sample approaches to highlight the differences among the various federation methodologies

Research Plan

- On-going and future investigation
  1. Detection of large scale failure
  2. Identification of partitions boundaries
  3. Localization of partitions: where is everybody?
  4. Handling of multiple and simultaneous node failure
  5. Hybrid solutions involving the placement of new nodes and the repositioning of existing ones
  6. Recovery subject to QoS requirements
  7. Efficient restoration of k-connectivity properties
  8. Coverage-aware connectivity restoration
  9. Application-guided failure recovery
  10. Mathematical formulations and analytical solutions
  11. Hardware-based prototyping and implementation

Project is currently funded by the National Science Foundation

Sample of Recent Publications

1) M. Younis, S. Lee, I. F. Senturk, and K. Akkaya, "Topology Management Techniques for Tolerating Node Failure," Book Chapter, The Art of Wireless Sensor Nets, Ed. Habib Ammari, *Springer*, to appear in 2013.

2) A. Abbasi, M. Younis, and U. Baroudi, "Restoring Connectivity in Wireless Sensor-Actor Networks with Minimal Topology Changes," *IEEE Trans. on Vehicular Tech.*, Vol. 62, No. 1, pp. 256-271, January 2013.

3) S. Lee and M. Younis, "Optimized Relay Node Placement for Connecting Disjoint Wireless Sensor Networks," *Computer Networks*, Vol. 56, No. 12, pp. 2788 – 2804, 2012.

4) A. Alfadhly, U. Baroudi, and M. Younis, "--An Adaptive Connectivity Restoration Algorithm for Wireless Sensor-Actor Networks," *International Journal of Autonomous and Adaptive Systems* (to appear).

5) A. Abbasi, M. Younis, and U. Baroudi, "A Least-Movement Topology Repair Algorithm for Partitioned Wireless Sensor-Actor Networks," *Int'l Journal of Sensor Networks*, Vol. 11, No. 4, pp. 250-262, 2012.

6) F. Senel, and M. Younis, "Relay Node Placement in Structurally Damaged Wireless Sensor Networks via Triangular Steiner Tree Approximation," *Computer Comm.*, Vol. 34, No. 16, pp. 1932 – 1941, 2011.

7) F. Senel, M. Younis, and K. Akkaya, "Bio-inspired Relay Node Placement Heuristics for Repairing Damaged Wireless Sensor Networks," *IEEE Trans. on Vehicular Tech.*, 60(4), pp. 1835 – 1848, April 2011.

8) M. Younis, S. Lee and A. Abbasi, "A Localized Algorithm for Restoring Inter-node Connectivity in Networks of Moveable Sensors," *IEEE Trans. on Computers*, Vol. 59, No. 12, pp. 1669-1682, December 2010.

9) J. L.V.M. Stanislaus, and M. Younis, "Mobile Relays Based Federation of Multiple Wireless Sensor Network Segments with Reduced-latency," in the *Proceedings of the IEEE International Conference on Communications (ICC 2013)*, Budapest, Hunger, June 2013 (to appear).

10) F. Senel, and M. Younis, "Optimized Relay Node Placement for Establishing Connectivity in Sensor Networks," *Proc. of the IEEE Global Comm. Conf. (GLOBECOM 2012)*, Anaheim, CA, December 2012.

11) A. Abbas, and M. Younis, "Interconnecting Disjoint Network Segments Using a Mix of Stationary and Mobile Nodes," *Proc. of the IEEE Conf. on Local Computer Nets (LCN 2012)*, Clearwater, FL, Oct. 2012.

Thank You