Dynamic Energy Aware Deployment and Routing Protocol for Wireless Sensor and ROBOT Networks: Topological Perspective

Md. Enamul Haque

Department of Computer Engineering
King Fahd University of Petroleum and Minerals
Saudi Arabia

g201204920@kfupm.edu.sa

April 9, 2014

Coming up

Todays Agenda

- How ROBOTS configure the layout of a sensor network from scratch?
- How ROBOTS foster faulty nodes after network deployment?
- How ROBOTS replace and/or fix a damaged module?
- Future challenges.
- My research area.

Wireless Sensor and ROBOT Network

- Collaborative interaction of standalone sensing units.
- Collection of autonomous sensing nodes that communicate via wireless links.
- Deployed both indoor and outdoor scenarios.
- Expected to work unattended.
- Monitors the region of interest (ROI).
- Reports measurements to central location.

- Consists of resource-rich, mobile ROBOTs and resource-constrained stationary sensors.
- Collaborative action performed by the ROBOTs.
- Actions are triggered by the static sensor data.
- Distributed control system.
- Examples: Underwater WSRN, Robotic gardening system.

Projects by DRL at MIT

Aquanode and AMOUR

Robot assisted and dependent scenarios

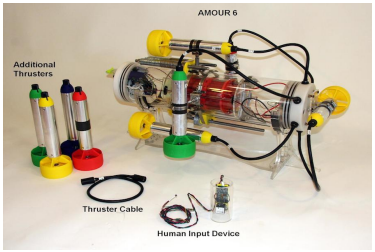


Figure : AMOUR VI with spare thrusters and HID

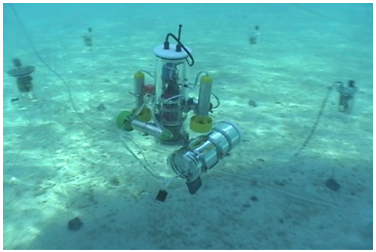


Figure : AMOUR with sensors in Moorea

Figure : Aquanodes drying in the sun

Image Copyright ©Distributed Robotics Lab (DRL), MIT

Underwater WSRN

Aquanode sensors

- Static nodes
  - Aquanode.
  - Equipped with communication, sensing, and processing modules.
  - Packed in cylindrical watertight containers.
  - Built in acoustic modem.
  - Anchored with weights to make static.
  - Measure water temperature, pressure, and other chemical indicators.
- Mobile nodes
  - AMOUR type
  - Functionality of Aquanode + advanced camera system to avoid obstacles.
  - Downloads data from Aquanodes.
  - Moves to the location to sample the environment on an event of interest.
  - Experiments conducted in oceans, rivers, and lakes with promising outcomes.

ROBOTic gardening system

Autonomous greenhouse

- Static nodes
  - Monitors soil humidity.
  - Issue watering requests.
- Mobile nodes
  - Perform as gardeners by delivering water and nutrients to the plants.
  - Custom wearing system with arm, eye-in-hand camera, mesh networking.
  - Harvesting the fruits upon demand.
  - Plant specific growth model.

Figure : Robotic gardening.

Image Copyright ©Distributed Robotics Lab (DRL), MIT

What can we infer from those scenarios?

Degree of bondage between Robots and sensors?

What happens in absence of robotic agents?

- Aquanodes are loosely coupled with AMOUR.
- Robots and soil sensors are strongly coupled.

Degree of integration

- Robot assisted network: Low degree of integration. Aim is to augment the existing capability of the sensors.
- Robot dependent network: High degree of integration of the robotic nodes into the WSN's core functionality.

What next?

Operational Challenges

What makes us dig more into this area?

- Resource depletion (energy, memory, processing)
- Connectivity disruption.
- Malicious attacks.
- Harsh external conditions.
- Real time inter robot task allocation.
- Distributed sensor-robot coordination mechanisms.

Robot Assisted Sensor Placement

Robot Assisted Sensor Placement

Where human intervention is impossible

Node placement by mobile actuators has been recently identified as a key strategic direction

1. Can drop off sensors from its reservoir expand network coverage.
2. Can act as intermediate relays to support node failures.

Deployment Topologies

According to the published works

Past trend

- BLANKET coverage problem.
- Objective: Lay nodes in such a way that the total coverage area is maximized.
- Examples: Environment mapping and monitoring applications.
- Method Described: Backtracking-Based Sensor Placement.

Recent trend

- FOCUSED coverage problem.
- Objective: Monitor a POI and its vicinity.
- How: Maximal coverage defined as the radius of the largest hole-free disc centered at POI.
- Examples: Disaster management. Earthquake, Active volcano crater monitoring, etc.
- Method Described: Carrier Based Focused Coverage Formation.

Past Trend

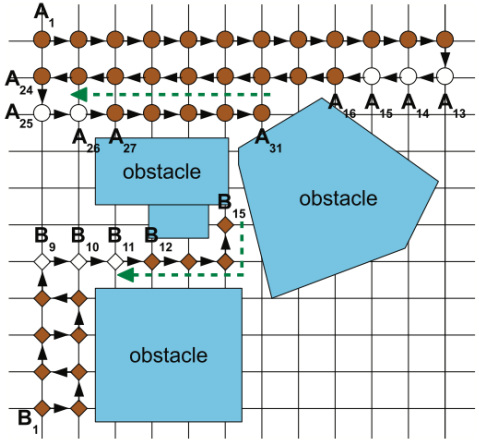
Assumptions:

- Each robot carries unlimited number of sensors.
- Virtual graph.
- No sensor pickup is considered.
- Square grid with 4 visiting directions.
- Trajectory governed by predefined direction ranking.
- Can not move beyond ROI.

Backtracking-Based Sensor Placement (cont'd)

Improvement of LRV method

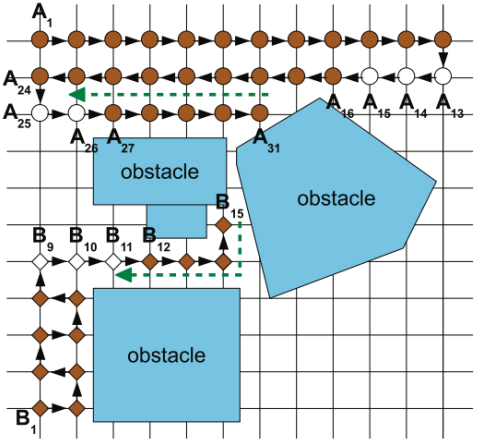
- What happens at sensor nodes?
  - Stores a vector (position, $robot_{ID}$, $sequence_{NR}$, color, $back_{PTR}$) in its local memory.
  - Robots provide first 3 components.
  - Last 2 components are learned dynamically - hello message exchange.
- Coloring and back pointer:
  - White: have at least one adjacent empty vertex.
  - Black: no adjacent empty vertex.
  - $back_{PTR}$: location of the first white node along the robot's backward path.



Backtracking-Based Sensor Placement (cont'd)

Example

Two robots continually lay sensors on the ground.
Top row nodes A1 to A13 have no back pointer. A14 points to A13, A15 points to A14, A16 refers to A15 and this knowledge is propagated down the second row till A24 and A25.
Then A26 points to A25 whereas A27 points to A26. Successors of A27 until A31 also point to A26.



Recovery from dead ends.

When a robot has no other way out!

⋆ When stuck at a dead end, an actuator asks the current sensor for its back pointer.
⋆ If it is null, it tries to find a stored back pointer in its neighborhood.
⋆ If none was available, the actuator terminates.
⋆ Otherwise, it moves to the next adjacent sensor with lowest sequence number whose backPTR coincides with the actuators destination.
⋆ Robot A travels from A31 to A26 and robot B departs from B15 to B11. Both resume placing sensors from there.

Null in the sensor + no pointer in the neighborhood = All the region is covered.

White sensor contention

How to avoid?

What happens when multiple robots try to turn a white sensor into black?

⋆ Search back pointer to get to white node.
⋆ If this node is placed by another robot?
⋆ The robot which reaches first serves.

So the path is being stolen.

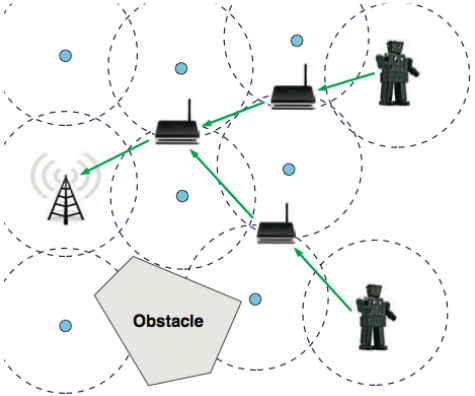
To avoid, the robot informs its current sensor to erase back pointers along the forward path to the first found white node or the first node with null in its backPTR field

- **What to appreciate?**
  - Low message passing overhead due to localized nature.
  - Terminates within finite time.
  - No additional cost in failure free area.
  - Minimal cost in failure-prone environments.
  - Significant savings in number of robot moves and messages compared to LRV.
- **What are the limitations?**
  - Deletion of back pointer may cause deadlock for other robot.
  - So, the serving robot has to take care of all the region by itself.
  - Load balancing is needed between robots.
  - Coverage guarantee is failed if the serving robot fails.
  - Paramount limitation: Unrealistic belief that robot can carry unlimited number of sensors.

Multi-Robot Search and Monitoring with Static Relays

How to choose sensor and relay points?

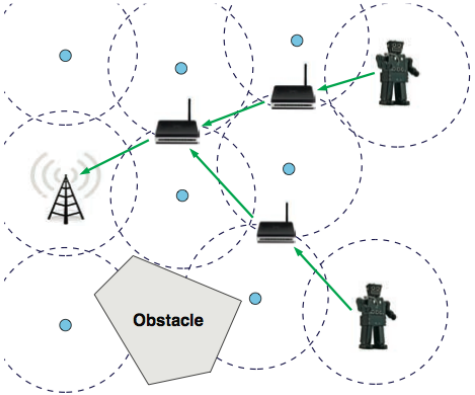
- Consideration
  - Obstacle distribution known.
  - Offline computation enabled.
  - How many sensing points?
  - How many relay nodes?
- Principles
  - Path of relay nodes has to be pre-constructed.
  - Robots carry limited number of sensors and can drop wherever needed.
  - Two disjoint robot tours for mapping the environment are to be derived.
  - Each robot visits designated sensing and relay nodes.
  - The longest tour must be minimized.



STARS (STAtic Relay-aided Search)

How to choose sensor and relay points?

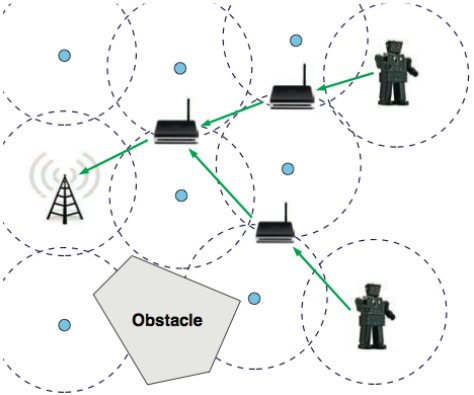
- Sensing Point Identification.
  - It follows the set covering problem (NP hard).
  - Sensing point with maximum number of uncovered cells.
  - Triangle lattice pattern for greedy heuristic
- Relay Point Identification.
  - Greedy heuristic is run to find dominating set.
  - Relays are used to make DS.
  - Intermediate nodes are added to connect DS.



STARS (STAtic Relay-aided Search) (cont'd)

How to choose sensor and relay points?

- Assignment of Precedence Constraints.
  - Precedence graph is generated at base station.
  - Breadth first search is used.
  - Leaves are sensing points not serving as relays.
- Tour Generation.
  - New generalization of the Traveling Salesman Problem (TSP).
  - Precedence Constrained Two TSP (PC2TSP).
  - Set of cities is split into consumers and suppliers



- **What are the limitations?**
  - Priori knowledge of the obstacle distribution in the field.
  - Insufficient to perform a massive scatter over a large region.
  - Not fault-aware, for unexpected failures.
- **Any good thing?**
  - Reduces the infrastructure cost by employing a few mobile robots and a plethora of cheaper, static nodes to monitor suspicious events in the region of interest.

Recent Trend

Tessellation

Periodic tiling

A tessellation of a flat surface is the tiling of a plane using one or more geometric shapes, called tiles, with no overlaps and no gaps.

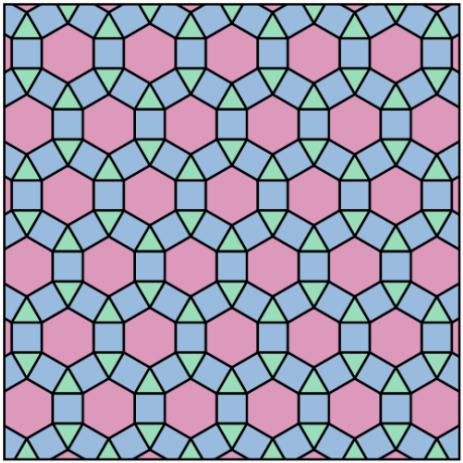
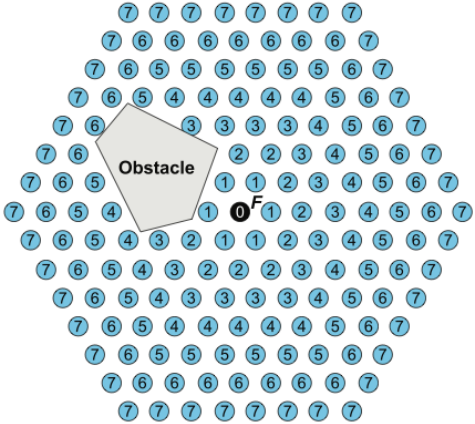


Image Copyright ©Wikipedia

Focused Coverage Formation

For unknown 2D environment

- Problem Statement
  - Unknown 2D environment.
  - Location of POI = F.
  - One / few mobile robots.
  - Base points.
  - Robots are aware of their locations.
- Goal
  - Develop carrier based sensor placement algorithm.
  - Hexagonal layers, F-coverage.
  - Equilateral triangle tessellation.
  - Vertex coverage problem solution.



F-Coverage Problem Over TT Graph

Vertex Coverage Problem

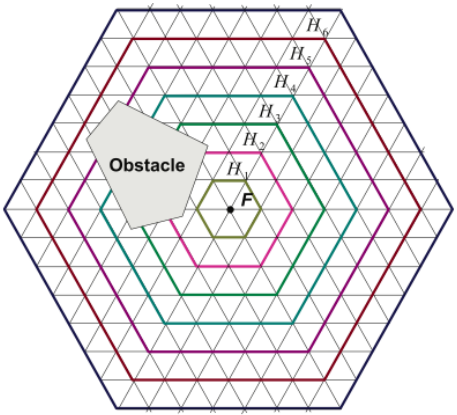


Figure : F-coverage problem envisioned as a vertex coverage problem over a TT graph

RAR (reactive advertising routine)

1. RAR runs locally on each deployed sensor on demand.
2. Its objective is to publish along the network border any empty vertex induced by node failure.

ISP (iterative sensor placement)

3. ISP runs in a distributed manner on sensors and robots spanning multiple iterations.
4. At each iteration, robots enter the environment from base points, place sensors at empty vertices, and then return to fetch sensors for a new iteration.

Reactive Advertising Routine.

At a glance

- At local sensors
  - Empty vertex list (EVL).
  - List contains 6 vertices from TT.
  - Exchange of hello message.
- Failure propagation
  - Forwarded outward, away from F, using Greedy-FACE-Greedy (GFG) routing.
  - Ascending order from one hexagon to another.
  - Message stops when traverses the perimeter of the outer hexagon.
  - Local EVL of all the border nodes contain all the empty vertices inside the network.

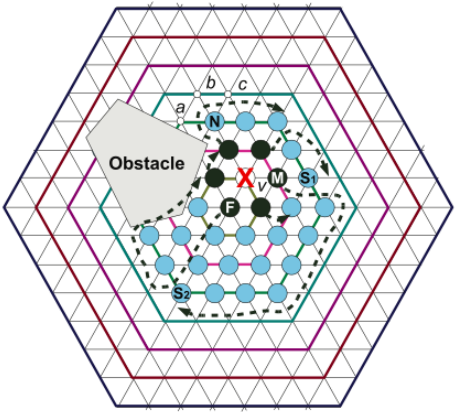


Figure : Reactive Advertising Routine

Reactive Advertising Routine (cont'd)

Algorithm

Algorithm 1 Reactive Advertising Routine (RAR): runs on sensors

Input: incoming HELLO messages from 1-hop neighbors
Output: FAILURE_NOTIFICATION message sent to network border

```
1:  EVL ← {coordinates of all 1-hop neighbors};
2:  while true do
3:      if incoming HELLO message from neighbor ⟨i, j, k⟩ arrives then
4:          EVL ← EVL \ ⟨i, j, k⟩;
5:      end if
6:
7:      if missed HELLO message from neighbor ⟨i, j, k⟩ T times then
8:          EVL ← EVL ∪ ⟨i, j, k⟩;
9:          send( FAILURE_NOTIFICATION, ⟨i, j, k⟩ );
10:     end if
11:
12:     if incoming FAILURE_NOTIFICATION message from neighbor ⟨i, j, k⟩ arrives then
13:         if I am a border node then
14:             EVL ← EVL ∪ ⟨i, j, k⟩;
15:         end if
16:         if this message not received before from the same direction then
17:             send( FAILURE_NOTIFICATION, ⟨i, j, k⟩ );
18:         end if
19:     end if
20: end while
```

Figure : Reactive Advertising Routine

Reactive Advertising Routine (cont'd)

Greedy forwarding and Face routing

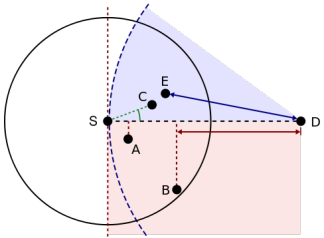


Figure : Greedy forwarding variants: The source node (S) has different choices to find a relay node for further forwarding a message to the destination (D). A = Nearest with Forwarding Progress (NFP), B = Most Forwarding progress within Radius (MFR), C = Compass Routing, E = Greedy

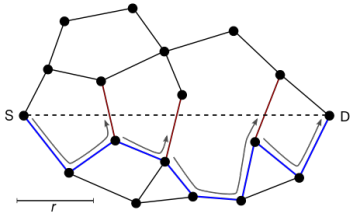


Figure : Face routing: A message is routed along the interior of the faces of the communication graph, with face changes at the edges crossing the S-D-line (red). The final routing path is shown in blue.

How robot marches?

Robot moves greedily along the straight line.

Obstacle found: it rotates around it following a predetermined direction until greedy movement can be resumed.

It periodically transmits a beacon message containing the target location.

Other robots reply with hello message.

Robot B stops marching and beaconing when it receives a hello message or it reaches F.

Avoiding multiple deployment

Another robot approaching or already deployed

Issue: Another robot tries to approach / Already sensor is placed there.
Solution: Before dropping sensor at v, B waits for maximum number of times and transmits beacon. It identifies the following cases.

- *Void Case:* No neighboring sensors.
- *Occupancy Case:* Hello message received whose origin is at v.
- *Competition Case:* Another robot is attempting to place a sensor at v.
- *Ordinary case:* Any other except from the above.

Competition Case

Which robot is gonna win?

1. A and B both have cargo capacity of one.
2. Travel along thick arrow lines.
3. Meet at F and compete for sensor placement.
4. Robot with minimum ID (A) wins the competition.
5. As B loses, it gets beacon from node placed at v. Migrates to search phase
6. In the next iteration, if any node is reserved, robot does not discover.
7. Robots return to Base point when all carried sensors are deployed or no empty vertex is found.

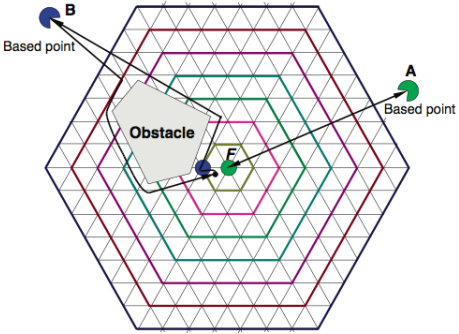


Figure : Competition at F.

Ordinary Case

When robot wins.

- Robot drops sensor at v.
- Beacons to find another empty vertex.

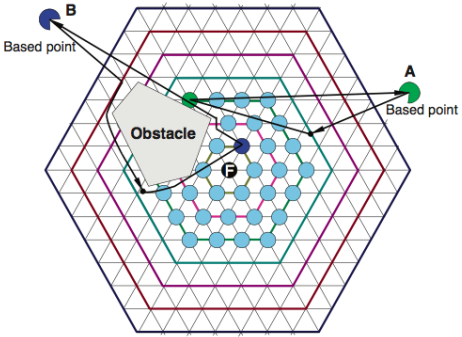


Figure : Ordinary case

Flow chart of an ISP iteration

At a glance

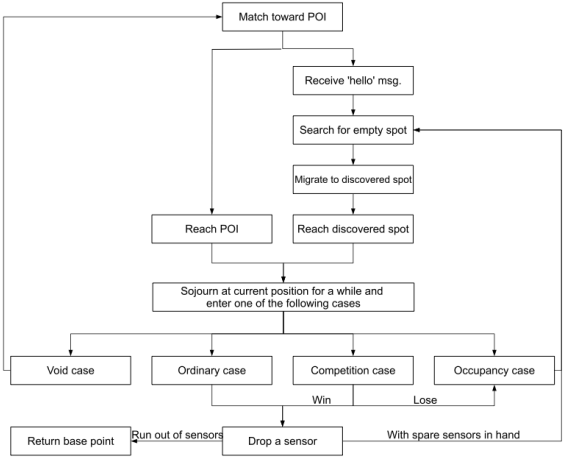


Figure : Flow chart of an ISP iteration

- First algorithm in which the sensor placement task in F-coverage environments is carried out by mobile actuators (robots).
- Strictly localized.
- Communication overhead is limited to the immediate neighborhood. (k=1)
- Obstacle aware.
- Efficient in inter robot coordination.
- Robust and guarantees maximal coverage while node failures.
- Realistic about the carrying limitations of the robots.

How Far We Reached?

What's left?

- How ROBOTS configure the layout of a sensor network from scratch?
- How ROBOTS foster faulty nodes after network deployment?
- How ROBOTS replace and/or fix a damaged module?
- Future Challenges.

How to replace failed sensors?

Constraint: Without altering the overall network topology.

Solution: Use redundant sensors with nodal geographic migration.

Sensor Maintenance (cont'd)

Most common issue

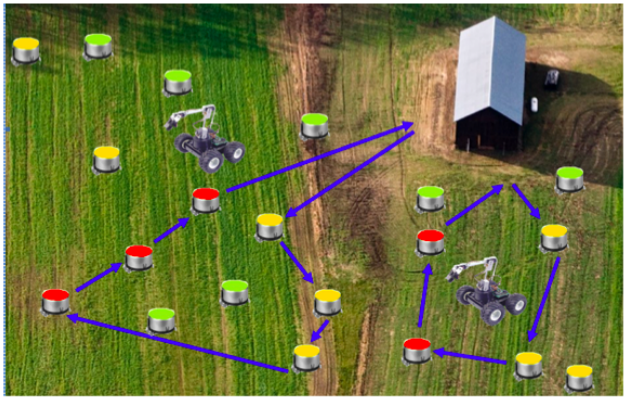


Figure : Illustration of the CBCR problem. Active, passive and damaged sensors are drawn in green, yellow and red, respectively. Blue arrows indicate the direction of the replacement trajectories.

Pioneers in Robot Assisted Sensor Relocation

Significant researches

1. Greg Fletcher, Xu Li, Amiya Nayak, and Ivan Stojmenovic. Randomized robot-assisted relocation of sensors for coverage repair in wireless sensor networks. In Proceedings of the 72nd IEEE Vehicular Technology Conference (VTC-Fall), Ottawa, Canada, October 2010, pp. 15.

2. Rafael Falcon, Xu Li, Amiya Nayak, and Ivan Stojmenovic. The one-commodity traveling salesman problem with selective pickup and delivery: An ant colony approach. In IEEE Congress on Evolutionary Computation (CEC), Barcelona, Spain, 2010, pp. 43264333.

3. Rafael Falcon, Benoit Depaire, An Caris, Xu Li, Koen Vanhoof, Amiya Nayak, and Ivan Stojmenovic. A Hybrid Metaheuristic Approach to Sensor Relocation with a Single Robot. Manuscript submitted.

1. Partition the network into active and passive static nodes.
2. Active nodes: responsible for monitoring the region.
3. Passive nodes: go into sleep mode to save energy and are thus redundant.

Notes: Static nodes are abundantly scattered due to cheap cost.

Randomized robot-assisted relocation of sensors for coverage repair in wireless sensor networks.

Robotic agents follow a *randomized* trajectory all over the filed.
Stationary sensors guide robots to find the coordinates of nearby damaged and spare sensors.
Grid based implementation.
Low message overhead, minimization of the repair latency.
Robots treated as unconstrained-units is chimerical.

How robots sense holes and passive sensors?

In periodic discovery cycles robots send beacon message carrying its own location. Active sensors reply with the location of nearby passive sensors and holes within their neighborhood. It is localized within small vicinity.

What are the robot action after notification?

1. Moves to the area.
2. Fixes the nearest hole using the local passive unit or spare sensor at hand.
3. If there is no hole but redundant units are available, it picks with probability $\frac{1}{S}$.
4. Notifies the corresponding proxy (active node) to update the list of redundant nodes.
5. Task contention is ruled out using sever rules.

Basic idea

1. Implementation using grid topology.
2. Robots use LFV policy to visit next grid vertex to probe.
3. Why? To restrict the direction of the robots to travel + eliminating randomness.
4. Superior results in terms of moving distance, number of moves and communication cost compared to original.

Positives

1. Simple algorithm.
2. G-R3S2 does not suffer from coverage repair when LFV ensures full exploration.

Negatives

1. R3S2 suffers from coverage repair.
2. Local termination condition is not clear.
3. Robots may loose all its power wandering for passive node collection.

The one-commodity traveling salesman problem with selective pickup and delivery: An ant colony approach.

- Ant colony optimization (ACO) architecture is used.
- MMAS, one variant of ACO, is used to deal with large TSP instances.
- Resultant scheme is called $MMAS - SC^2R$

- WSN is described as complete graph $G = (V, E)$.
- $V = v_1, v_2, \ldots, v_n$ is the vertex set, E is the edge set.
- e_{ij} is directed edge from v_i to v_j.
- Each edge has associated travel cost c_{ij}
- Elements of V is either passive sensors (pickup) or damaged sensors (delivery).
- Each customer v_i has unitary demand q_i.
- q_i is $+1$ when delivery and -1 when pickup.
- v_1 is base station with demand $q_1 = 0$.

- $V = \{v_1 \cup V_D \cup V_P\}$
- $V_D = \{v_i \in V : q_i = 1\}$
- $V_P = \{v_i \in V : q_i = -1\}$
- $V_D \cap V_P = \varnothing$
- Robot can carry at most Q sensors. Initially it carries Q_0 from base station where $0 \leq Q_0 \leq Q$.
- Profit assigned to robot for delivery node, $p_i = 1$.
- Profit assigned to robot for pickup node, $p_i = 0$.
- Tour is feasible for robot only if $p_t = |V_D| + 1$ and capacity constraint is never violated.

At the beginning of every iteration,

- Ants must be initialized on different vertices to start their tour and add components.
- Ant is placed depending on its initial cargo Q_0.
- The feasible initial neighborhood N_k for the kth ant is defined as equation 1.
- $Q_{max} = min(n_D, Q)$, n_D= number of remaining damaged nodes. initially $n_D = |V_D|$

$$N_k = \begin{cases} V - V_D - \{v_1\}, & \text{if } Q_0 = 0 \\ V_D, & \text{if } Q_0 = Q_{max} \\ V - \{v_1\}, & \text{otherwise} \end{cases} \quad (1)$$

Heuristic desirability η_{ij} of transitioning from the current problem state v_i to any neighbor state v_j is posed as a function of their distance, the type of the current node (pickup or delivery) and the current cargo of the search agent.
Heuristic functions envisioned:

- Drop off first rule.
- Pickup first rule.
- Nearest neighbor rule.
- Look ahead rule.
- Majority rule.
- Stochastic rule.

Performance of the six heuristic rules in the MMAS model

Majority wins!

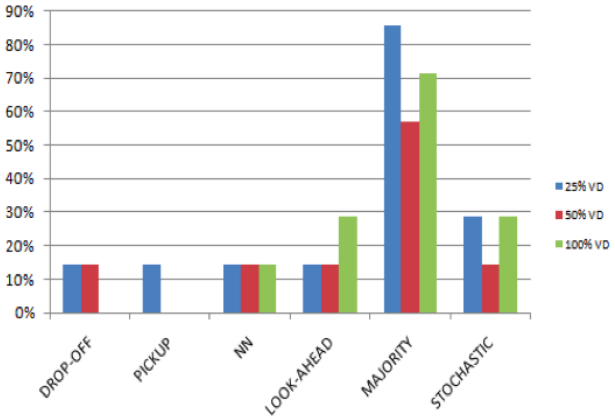


Figure : % of tested files where the best overall solution was found per heuristic rule.

A Hybrid Approach

The Deterministic Robot Motion Scenario: A Hybrid Approach

At a glance.

- Genetic algorithm and ACO both are used.
- Cyclic path representation.
- Problem specific local search operator.
- Adaptive population renewal mechanism.

Cyclic Path Representation

Adopted as solution encoding.

Tour starts at v_1 and visits nodes to the right at closed loop until gets back.

$$|\overline{\phi}| = |\{e_{ij} \in \overline{\phi}\}| = 2.|v_D| - Q_0 + 1 \tag{2}$$

- Initial population is built by heuristic and local search operators.
- Candidate list preparation.
- Candidate pickup list preparation from the initial candidates = CPL.
- Generate tour from the CPL.
- Calculate improved individual list from tour and CPL.
- Evaluation of the total population.
- Offspring generation till the stop criteria met.

Algorithmic Overview

Input: pop_size, cl, p_{CR}, β, p_0, ageIni, maturity, p, ρ
Output: Best ever-found individual $\vec{\varphi}_*$

1: CL ← *ComputeCandidateList( )*;
2: CPL ← *ComputeCandidatePickupList( )*;
3: **for** i = 1 to pop_size **do**
4: $\vec{\varphi}_i$ ← *GenerateIndividual*(CPL);
5: $\vec{\varphi}_i$ ← *ImproveIndividual*($\vec{\varphi}_i$, CL);
6: **end for**
7: *EvaluatePopulation( )*;
8: **while** termination criterion not met **do**
9: *offspring.clear( )*;
10: **for** i = 1 to pop_size / 2 **do**
11: $[\vec{\varphi}_{p_1}, \vec{\varphi}_{p_2}]$ ← *SelectCrossoverParents( )*;
12: **if** *rand( )* ≤ p_{CR} **then**
13: $\vec{\varphi}_{p_1}$ ← *ApplyMutation*($\vec{\varphi}_{p_1}$) with $p_M(\vec{\varphi}_{p_1})$;
14: $\vec{\varphi}_{p_2}$ ← *ApplyMutation*($\vec{\varphi}_{p_2}$) with $p_M(\vec{\varphi}_{p_2})$;
15: $\vec{\varphi}_{child}$ ← *ApplyCrossover*($\vec{\varphi}_{p_1}, \vec{\varphi}_{p_2}$);
16: $\vec{\varphi}_{child}$ ← *ImproveIndividual*($\vec{\varphi}_{child}$, CL);
17: *offspring.add*($\vec{\varphi}_{child}$);
18: **end if**
19: **end for**
20: *UpdatePopulation( )*;
21: *EvaluatePopulation( )*;
22: *UpdatePheromoneTrails( )*;
23: **end while**
24: **return** best-ever individual $\vec{\varphi}_*$;

Name	Description
pop_size	Number of individuals in the population at every generation.
cl	Size of the candidate pickup list and candidate list.
p_{NFC}	Probability of selecting the nearest feasible customer during population initialization.
p_{CR}	Crossover probability.
β	Sensitivity coefficient of the heuristic information in ACO.
p_0	Probability of "greedy" next element selection during ACO-based crossover.
ageIni	Number of generations before parent mutation occurs.
maturity	Aging threshold for a chromosome.
p	Percentage of pickup nodes to be replaced during mutation.
ρ	Pheromone persistence rate.

Figure : Parameters of the GA-ACO approach

Robot Assisted Sensor Maintenance

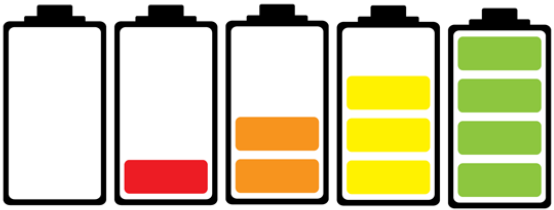
Sensor Maintenance (cont'd)

Can we buy new ones?

Sensor Maintenance (cont'd)

Most common issue

Battery depletion



Possible Options to Recover Battery Depletion

Ambient energy sources

Introduction of ambient energy sources.

- Solar.
- Wind.
- Water.
- Kinetic and
- Thermal.

⋆ In adverse scenarios the above sources maybe of little or no use.

Studying the feasibility of energy harvesting in a mobile sensor network.

- **Assumptions:**
  - Geographically distributed sensor network composed of many individual nodes.
  - Energy producer nodes can recharge themselves from environment using solar panel.
  - Energy consumer nodes consume energy in computation and communication.
- **Challenges:**
  - Energy producers need to work with a non-uniform geographic energy distribution.
  - How energy producers deliver the energy they have gathered to the consumer nodes.
- **Suggestion:**
  - Energy producer nodes are treated as mobile and called the energy equalizer.

Recharging sensor nodes using implicit actor coordination in wireless sensor actor networks

- Distributed approach for recharging using actor fleet.
- Sensors submit energy status to the actors within range.
- Energy status includes ID, location, current power, power ratio and interest coefficient.
- Actors execute an implicit coordination mechanism based on the above information using evolutionary and swarm intelligence.

Future Challenges

- SC^2R framework can use fleet of robots instead of using single robot. Sensor replacement becomes much faster.
- 1-VRP-SELPD (One commodity vehicle routing problem with selective pickup and delivery): How many replacement trajectories needed? Which nodes must be visited?
- Route has to be predefined for the Robots.
- In vehicular routing, all the possible routes have to be considered for decision making.

Robot Assisted Wireless Sensor Networks (cont'd)

Probable solutions.

- Divide the network field into small geographic regions
- Each robot is assigned to a unique region.
- Shrink or expand local regions by embracing passive or damaged nodes from adjacent regions.
- Change has to be beneficial for both regions according to some cost function.
- Local regions changes dynamically, so trajectory has to be readily recalculated.
- No global knowledge required.
- This solution yields deterministic robot tours.

- Robot dispatch problem: Sensors report events from the filed to robots. Robots are dispatched accordingly.
- What happens when new events occur?
- Multiple rounds of Robot dispatch needed.
- Goal: Schedule the robot traveling paths (task fulfillment) in an energy balanced way so that their overall lifetime (the number of dispatch rounds) is maximized.
- For the single-event case, the service discovery algorithm iMesh finds the nearest (or a nearby) robot with lower communication overhead.
- Distance and reluctance is used for the robots to take the decision.

- Reluctance is inversely proportional to the remaining battery level.
- How to expand this hybrid scheme so that it can handle multiple events?
- How to handle when multiple concurrent event occur?
- Events should not have access to all the Robots.
- Robots are regarded as static entities waiting for events.
- Good distribution of robots with respect to the location of likely events.
- Self-configuration of the robotic network become pivotal issues

My Research Area

Major Agenda

Two fold

- Robot assisted sensor deployment in hazardous area.
- Energy efficient routing algorithm.

Sensor Deployment by Robots

Sensor Deployment by Robot

Executed in Robots memory

```
1: procedure DeploySensors(start, cp)
2:     start ← {0, 0}
3:     offset ← TR_sensor
4:     m_dir ← right
5:     BF ← 0
6:     maxX = 100, maxY = 100
7:     if o_dir = NORTH then
8:         PLACESENSOR(maxY, cp, m_dir,offset)
9:     else if o_dir = SOUTH then
10:        PLACESENSOR(maxY, cp, m_dir,−offset)
11:    else if o_dir = EAST then
12:        PLACESENSOR(maxX, cp, m_dir,offset)
13:    else if o_dir = WEST then
14:        PLACESENSOR(maxX, cp, m_dir,−offset)
15:    end if
16:    function PlaceSensor(maxP, cp, m_dir,offset)
17:        if o_dir = NORTH||SOUTH then
18:            co ← Y
19:        else if o_dir = WEST||EAST then
20:            co ← X
21:        end if
22:        if o_dir = NORTH||EAST then
23:            ops ←≤
24:        else if o_dir = WEST||SOUTH then
25:            ops ←≥
26:        end if
27:        while cp.co ops maxP.co do
28:            while obstacle = 0 or BF = 0 do
29:                put_sensors()
30:                forward to m_dir
31:            end while
32:        end while
33:        go_back()
34:        add_offset()
35:        BF ← 0
36:    end function
37: end procedure
```

Ensuring Whole Coverage by Finding Holes

Executed in Sensors memory

```
1:  procedure FindHoles(sensors)
2:      for all the sensors do
3:          check holes within range
4:          if hole found then
5:              inform robot with location
6:              DEPLOYSENSOR()
7:          else no holes found
8:          end if
9:      end for
10: end procedure
```

Figure : Critical region exploring algorithm

Post Deployment Event Monitoring

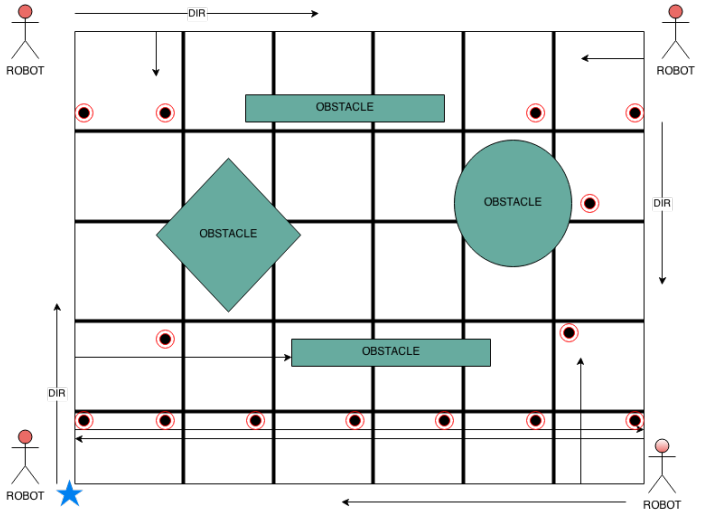
Executed in both Sensors and Robots memory

```
1: procedure MONITOR(area)
2:     Divide the area into virtual blocks
3:     Assign robots to blocks
4:     if EVENT == TRUE then
5:         run AODV
6:         send data to robot
7:     else monitor()
8:     end if
9: end procedure
```

Figure : Event monitoring algorithm

Deployment Region

Sample Deployment



Guaranteed Routing

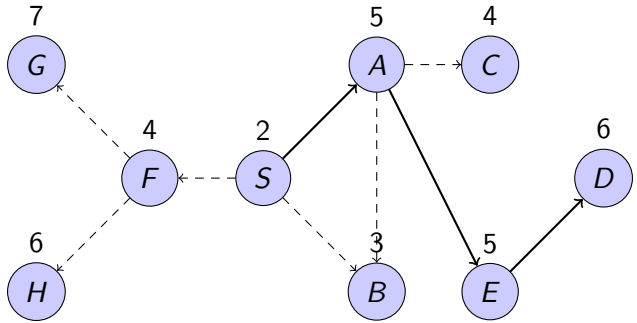


Figure : Example routing scenario. Each node is marked with remaining energy in Joules.

What if some node fails?

Find alternate route.

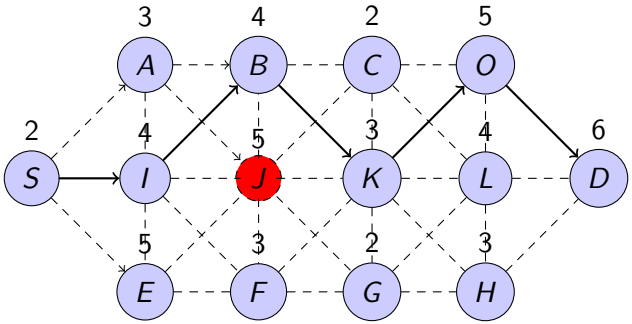


Figure : Case Study: Alternate route usage when any node failure occurs

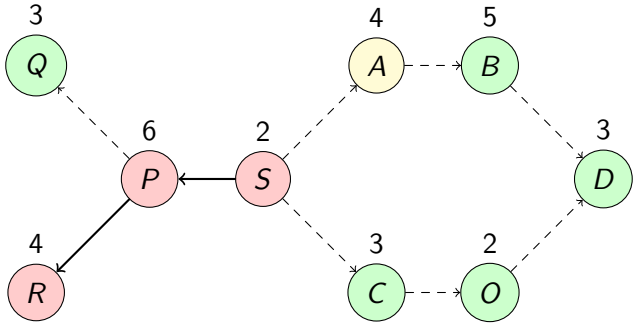


Figure : Case Study : Destination is on the Opposite Direction

Thank you for your cooperation

Feedback?

Q & A