

Course “Algorithmic Foundations of Sensor Networks”

Lecture 7: Sensor Network Deployment

Sotiris Nikolettseas

Department of Computer Engineering and Informatics
University of Patras, Greece

Spring Semester 2020-2021

Sensor Network Deployment

The Problem: "Given:

- a set of sensors
- a region of interest
- a certain application context

find:

- how
- where

these sensors should be placed"

- Connectivity: the resulting network topology should allow *information routing* to take place
- Coverage: the collaborative monitoring of the environment should allow a certain *quality of collected information*

Other aspects: equipment cost, energy limitations, robustness, etc.

Basic Deployment Questions (I)

- ① Structured versus randomized deployment
 - structured placement, by hand or via autonomous mobile robots
 - randomly scattered deployment (sensors thrown by e.g. a helicopter)
- ② Flat/over-deployment versus incremental deployment
 - are all sensors (possibly redundant) deployed *at the start*?
 - can sensors be added or replaced *incrementally*?
- ③ Network topology
 - simple star, grid, arbitrary multi-hop mesh, hierarchical
 - what type of connectivity/robustness guarantees are desired

- ④ Homogeneous versus heterogenous deployment
 - are all sensors of the same type?
 - is there a mix of high and low-capabilities sensors?

- ⑤ Coverage levels

What is the desired type and accuracy of monitoring information?

- event detection probabilities
- how many sensors should be able to sense every event?

I1. Structured vs randomized deployment

- randomized deployment is more suitable in large scale remote monitoring applications (e.g. when sensors are dropped from aircraft over a forest or an after-disaster area)
- structured deployment is more relevant in small - medium scale sensor networks (e.g. in a smart building, in precision agriculture)

I2. A possible methodology for structured placement

- step1: place the sink/gateway device(s) at a location (e.g. where a wired network connection or power supply is available)
- step2: place sensors in a prioritized manner where sensor measurements seem more necessary
- step3: if needed, add additional sensors to guarantee a certain desired connectivity level

Comment on step2: if needs are not known in advance, a uniform or grid-like placement can be used.

Comment on step3: it may be impossible to place sensors at particular locations (due to e.g. obstacles) $\Rightarrow$ a *delicate balance* between step 2 sensors for sensing and step 3 sensors for routing must be engineered.

13. Challenges in a randomized deployment

- problem: no way to configure a priori the exact location of each sensor
- solution 1: *post-deployment self-configuration mechanisms* (e.g. adjusting the transmission range, mobility) to achieve desired levels of coverage and connectivity (note: this also applies to structured deployments)
- solution 2: calculate (using random graph theory) the minimum *number of sensors* needed and their *operating specifications* to get satisfactory (on the average/with high probability) guarantees.

Occupancy analysis of randomized deployment - I

- occupancy problem: "put randomly m balls into n bins"

⇒ questions:

- minimum number of balls so that whp *all bins non-empty*?
answer: $cn \log n$ balls ($c \geq 2$) suffice since:

$$\Pr\{\exists \text{ empty bin}\} \leq n\left(1 - \frac{1}{n}\right)^{cn \log n} \leq nn^{-c} = n^{-(c-1)} \rightarrow 0$$

Occupancy analysis of randomized deployment - II

- *maximum/minimum number of balls in any cell?*
 - when $m=n$ then whp no bin has more than $\frac{e \ln n}{\ln \ln n}$ balls
 - when $m = n \log n$, then whp every bin gets $O(\log n)$ balls (i.e. the maximum is asymptotically equal to the expected)

II. Flat/over-deployment versus incremental deployment-I

- Flat deployment: all sensors deployed *at the start*
- Incremental deployment: sensors *incrementally deployed* during network operation/protocol execution

Note 1: For comparison *fairness*, in each case the *same total number of sensors* is deployed

II. Flat/over-deployment versus incremental deployment-II

Note 2: Sensors can be added in g groups g_i , at certain time instances t_i , e.g. $n = \sum_{i=1}^g g_i$

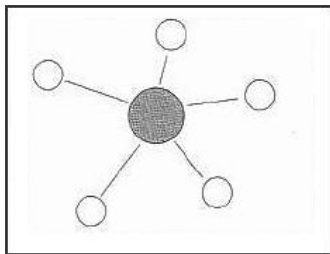
Question: what is the best incremental deployment scenario?

e.g.

- how to distribute sensors into groups?
- when to deploy each group?

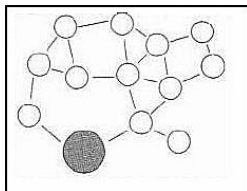
III1. Network Topologies: The single-hop star

- the *simplest* one, when all sensors directly communicate to the sink
- in small networks (e.g. in a smart room/home) it is feasible and *simplifies networking aspects* a lot
- weakness 1: *poor scalability* (e.g. in large networks the direct transmission may be of low quality, expensive or not possible at all)
- weakness 2: *poor robustness* (when a link fails)

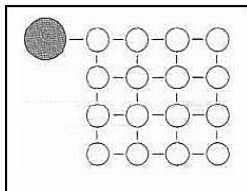


III.2. Network Topology: Multi-hop mesh and grid

- for larger areas: multi-hop routing is necessary
- the *arbitrary mesh* (weaker model, less performance guarantees, e.g. diameter may become high)



- the *grid* (stronger model, more easy to handle)



III3. Hierarchical/clustering network topologies-I

- sensors form *clusters* and transmit only to their *cluster heads*
- cluster heads aggregate, compress received data and transmit to the sink

Note 1: This approach is very relevant in *heterogeneous* settings when some devices/sinks/gateways are more powerful and can act as cluster heads.

III3. Hierarchical/clustering network topologies-II

Note 2: In structured deployments (e.g. smart buildings) a *second-tier* utilizing higher bandwidth (even wired networking) may be available.

Note 3: In random placements, *regular sensors may periodically become cluster-heads* (e.g. via self-election); in large networks however (and high data traffic) this approach can become inefficient (due to e.g. expensive distant transmissions by remote cluster-heads and since cluster-head rotation may be too slow and cluster heads spend their energy quickly).

III.4. Random graph topology models

- random graph: *a probability space whose sample points are graphs. Models of dynamic networks with node/links failures, unavailabilities, interactions etc.*
- $G_{n,p}$ random graphs: n vertices and each possible edge exists with probability p , independently.

Critique: in sensor networks dense, frequent interactions and physical proximity exist, so the edge independence assumption is not *very realistic* (e.g. if edges uv and vw exist, then uw possibly also exists).

III.4. Thresholds for monotone properties in random graphs

- $p_0 = f(n)$ threshold for property $A \Leftrightarrow$
$$\Leftrightarrow \begin{cases} p < p_0 \Rightarrow \Pr\{A\} \rightarrow 0 \\ p > p_0 \Rightarrow \Pr\{A\} \rightarrow 1 \end{cases}$$

Note: sharp transition around the threshold

- Examples in $G_{n,p}$ random graphs
 - $p = \frac{1}{n}$: threshold for existence of a giant connected component
 - $p = \frac{\log n}{n}$: threshold for connectivity

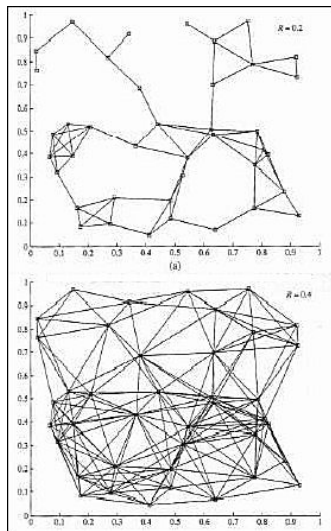
III.4. More relevant random graph models

- $G(n, R)$: random geometric graphs
- $G(n, k)$: nearest k -neighbors random graphs
- $G_{grid}(n, p, R)$: unreliable sensor grid
- $G_{n,m,p}$: random intersection graphs

Random Geometric Graphs

- definition: n vertices are placed uniformly randomly in the $[0,1]^2$ square and an edge (u,v) exists if the Euclidean distance of u and v is less than R
- R captures ability to wirelessly communicate in one hop (the wireless transmission range)
- important nice property: edges are not independent, e.g. the existence of an edge (u,v) is not independent of the existence of edges (u,w) and (w,v)

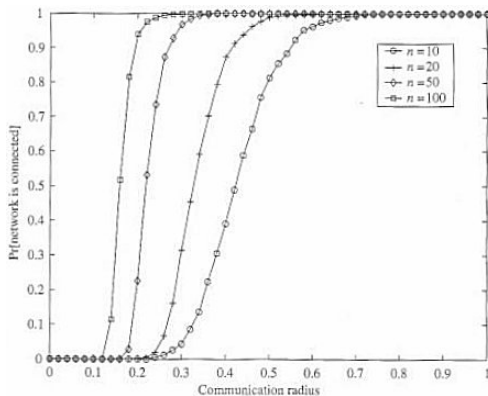
Illustration of sparse and dense $G(n,R)$



Connectivity properties in $G(n,R)$

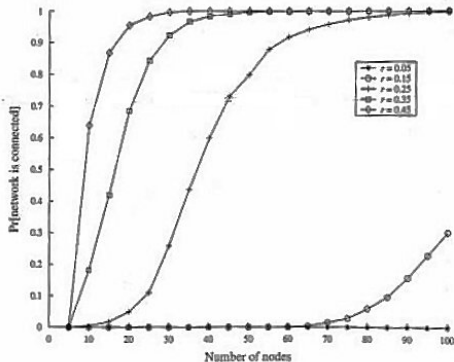
- Theorem (Gupta and Kumar): If $\pi R^2 = \frac{\log n + c(n)}{n}$, $G(n,R)$ is connected almost certainly when $c(n) \rightarrow \infty$, while it is almost certainly disconnected when $c(n) \rightarrow -\infty$
- i.e. the *critical wireless transmission range for connectivity* is $O(\sqrt{\frac{\log n}{n}})$
- Multiconnectivity: the critical R for $G(n,R)$ having the property that all nodes have at least k neighbors is *asymptotically equal* to the critical R for k -connectivity (i.e. existence of k vertex-disjoint paths between all pairs of vertices).

Connectivity probability with respect to R



- different critical R for different number of vertices n
- the transition becomes sharper for larger n

Connectivity probability with respect to n



- different critical densities for different R
- again, sharper transitions for large R (dense graph)
- critical R for different n is very relevant to network deployment

Connect to k-nearest vertices random graphs $G(n,k)$

- definition: n vertices are randomly placed in $[0, 1]^2$ and *each vertex is connected to its k nearest vertices*
- this captures ability of sensors to vary their transmission range (and thus power consumption) until k sensors are included in their neighborhood
- when $k \leq 0.074 \log n$, then network is disconnected
when $k \geq 0.9967 \log n$ connectivity is almost certainly guaranteed

The unreliable grid random graph $G_{grid}(n,p,R)$

- definition: n vertices are placed on a square grid within $[0, 1]^2$, p is the probability that a vertex (sensor) is active (not failed) and R is the transmission range of each sensor
- Properties:
 - for the active sensors to form a *connected topology* pR^2 must be at least $\frac{\log n}{n}$
 - the *diameter of the "active network"* (i.e. the maximum number of hops to travel between any two active nodes) is $O(\sqrt{\frac{\log n}{n}})$

Random Intersection Graphs $G_{n,m,p}$

- $G_{n,m,p}$ space: n vertices, m labels, each vertex chooses randomly, independently labels with probability p and vertices are connected by an edge iff they share at least one common label
- this model captures *resource sharing in a distributed setting*, e.g.:
 - vertices: servers, sensors
 - labels: printers, wireless frequencies
 - edges: nodes with conflict

III5. Connectivity using power control

- basic idea: *tune the transmission range* of sensors (through power control) to *adjust the connectivity properties* of the deployed network
- A complex and challenging *cross-layer issue*, with several interrelated (positive and negative) consequences, like:
 - increasing transmission range can improve connectivity (availability of end to end paths)
 - it may avoid obstacles and faulty areas
 - however, it can induce additional interference that reduces capacity and increases congestion
 - it increases energy consumption

III5. Basic power control methods - I

- Minimum energy connected network consumption (MECN)
(ensure that for any pair of nodes there exists a path in the graph that consumes least energy compared to any other possible path)
- Minimum common power setting (COMPOW)
(ensure that the lowest common power level that results to maximum network connectivity is chosen by all nodes)
- Minimizing maximum power (in a connected topology with non-uniform power levels, minimize the maximum power level among all nodes in the network)

III5. Basic power control methods - II

- Cone-based topology control (CBTC)
(each node increases its transmission range until it has at least one neighboring node in every α' -cone or it reaches its maximum transmission range)
- Local minimum spanning tree construction (LMST)
(construct a spanning tree topology in a completely distributed, local manner)

IV. Heterogeneous deployments - I

- some sensors have *more energy* than the rest
(for fairness reasons, in performance comparisons the total energy is the same as in the uniform case)
- some sensors may be *mobile*
- there may be *more than one sink* nodes/gateways

IV. Heterogeneous deployments - II

Basic implications:

- the existence of "super-nodes" may increase performance when handled properly
- mobility reduces energy a lot but may increase latency, thus satisfactory trade-offs are necessary.
Useful methods: biased/partial random walks, distributed motion coordination reducing visit overlaps

V. Coverage aspects

- more diverse and application-dependent than connectivity
- two *qualitatively different* sets of coverage metrics
 - k-coverage metrics: measure the degree of sensor coverage overlap
 - path-observability metrics: related to tracking moving objects

- The basic definition: An operating region is *k-covered* if every point is *within the transmission range of at least k sensors*
- a naive, expensive approach to check coverage: divide the area into a grid of very fine granularity and examine all grid points exhaustively to see if they are k-covered $\Rightarrow$ in an $s \times s$ area and with "grid resolution" ϵ , we must check $(\frac{s}{\epsilon})^2$ points (which may be very intensive for small ϵ)
- a better approach: *check if each "intersection subregion" is k-covered* (in a network with n sensors there may be $O(n^2)$ such regions and checking each of them may be difficult)

k-coverage: a distributed algorithm

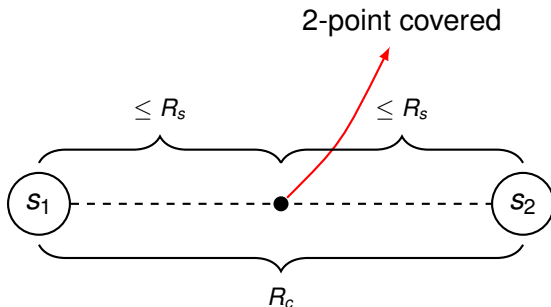
- Definition: a sensor is *k-perimeter-covered* if all points on the perimeter circle of its region are within the perimeters of at least k other sensors
- Theorem: A region is k -covered if and only if all n sensors are k -perimeter-covered
- Theorem (stronger): A region is k -covered if and only if *all intersection points of the perimeters* of the n sensors (and the perimeter of sensors and the region boundary) are covered by at least k sensors
- The distributed algorithm implied by the last theorem is quite fast (we have to only check intersection points)

A fundamental relationship between coverage and connectivity

Theorem (Wang et al):

If a convex region is k -covered by sensors with sensing range R_s and communication range R_c , the communication graph is k -connected so long as $R_c \geq 2R_s$

Intuition:



VI. Path observation metrics

- suitable primarily for *tracking moving objects* traversing a sensor network field
- the *maximal breach distance path*: the path that maximizes the distance between the moving target and the nearest sensor during the target's point of nearest approach to any sensor. This distance is called the maximal breach distance metric.
- It captures *ability of an adversary* with full knowledge of the network deployment to *avoid being detected* (i.e. it is a worst case notion)
- Algorithm: Voronoi cells + dynamic programming

Other path observation metrics

- the exposure metric: captures target observability
- the lowest probability detection metric
- a best case notion: the *maximal support distance* is the path minimizing the maximum distance between every point on the path and the nearest sensor.
- Algorithm: Delaunay triangulation + dynamic programming

Other coverage metrics

- percentage of desired points covered
- total area of k -coverage
- average coverage overlap: the average number of sensors covering each point
- maximum/average inter-sensor distance
- minimum/average probability of detection

VII. Mobile deployment

- mutual avoidance techniques are used to spread mobile nodes in order to avoid overlapping coverage and achieve load balance (completely distributed, local approach $\Rightarrow$ scales very well)
- approaches trying to ensure good connectivity: each node remains within range of k neighbors
- other approaches: incremental self-deployment: nodes move towards a new, "better" placement calculated based on the current one
- hybrid static/mobile deployments: either mobile nodes move to fill any coverage holes of a fixed deployment or a static deployment is guided by the information collected by mobile explorers.