ECE 659/493: IOT Signal Processing and Intelligent Sensor Networks

Instructor: Dr. Otman A. Basir

Group No: 2

Student ID: 20979706 Juhi Vasudev Bachani

20988007 Akashdeep Singh Khehra

20993717 Seoyoung Sim

Q1 : Transport and Application Layer Protocols

(1) What is the goal of sensor management in WSN?

- The goal of sensor management is to perform large number of tasks by providing the software operations.
- It is required for switching on the sensors or turning them off.
- Another goal is to move the nodes of the sensor, for scheduling sensor nodes, location exchange, inquiring the sensor node status.
- It is required for using location based addressing and attribute based naming in order to access nodes.

(2) **Review the following paper and summarize it using 500 (min) to 700 (max) words.**

Ans-

Sensor Management Using Relevance Feedback Learning

Chris Kreucher*, Keith Kastella and Alfred O. Hero III, IEEE Fellow

Section wise main features-

Section 1 Introduction:

The sensors are typically used to identify a group of targets and learn about their position and velocity which is termed as the kinematic state. The goals of sensor management include tracking errors, identification of accuracy, target detection probability, etc. Whereas the applications of sensor management include wireless networking and robot path planning. Many researchers have also proposed divergence measurement as an alternate method to sensor management. This is because the illustration of divergence metrics include the interactive search of database of images while searching for a desired image which is called Content Based Image Retrieval (CBIR). In case of CBIR application the goal image is a fixed entity so there is no requirement of any tracking algorithm.

There are basically two main contributions of this paper- Firstly, there is a theory based on Particle Filter based multitarget tracking that enforces the multitarget nature of the problem. Secondly, the estimation of the utility is done using the Renyi divergence method.

Section 2 (The joint multitarget probability density- JMPD)

In this section there is an examination regarding the core component of the sensor management strategy. The concept of joint multitarget probability density (JMPD) is explained in detail and the challenges associated in the numerical problems while implementing it on a grid. JMPD is a discrete hybrid system which is symmetric under permutation. The position and the velocity vectors are tracked and their description is given by the four dimension state vector. The number of grid cells needed grow as (Location)^targets where Locations- the number of discrete locations and target- the number of targets.

Section 3(The particle filter implementation of JMPD)-

In this section, there are solutions regarding the implementation of JMPD that offer a computationally tractable approach, enabling the considerations of real world scenarios. The algorithm used in this section allows the tracking of large number of objects with a relatively few particles and increases the efficiency of the particles. The approximation of joint probability density is taken which is given by- p(X|Z).

Section 4(Relevance feedback learning for sensor management)-

This section goes into detail of the sensor management method which is basically a learning technique that uses Renyi divergence as a matric. The calculation between two information densities f0 and f1 can also be done with Renyi divergence which is also called alpha divergence. On the basis of asymptotic analysis, alpha = 0.5 will cause the most discriminatory ability between similar densities. In case of extensions to non-myopic sensor management, the decision trajectories are clearly poor paths and the best path can be found using optimization techniques.

Section 5(Simulation Results)-

Section 5 provides a performance analysis of the tracker by employing sensor management for two model problems with increasing realism. There are also comparisons made in two sensor management strategies as well as a non-managed or a periodic approach. Using actual recorded target trajectories from a military battle simulation, synthetic situation is initially shown before moving to a realistic one. In both cases there is an assumption that the sensor is constrained by time, bandwidth and other physical factors, allowing it to measure just a portion of surveillance area at any given moment.

Section 6(Discussion/Conclusion)-

In conclusion, a technique named as relevance feedback learning is discussed which basically provides a method to manage agile sensors. It's working is in such a way that if a user has any particular image in his mind that has to extracted from the images that a system has learned. The system displays images from memory using distance metric called Kullback Leiber divergence. The aim of the system is to get an amount of group of targets in a surveillance region. So the user specifies the images displayed by the system and the system updates the images accordingly. This process is repeated until the goal image is found. A sensing action is also taken by the system which is called Renyi divergence in which the system updates the probability density on the number of statuses of the target after the measurement has been taken and provides the feedback accordingly.

(3) **Explain and compare, in your own words, the main attributes of the MQTT and COAP protocols. (Material on both protocols is posted on learn)**

	MQTT(MQ Telemetry transport)	CoAP(Constrained App Protocol)
Developer	Built by IBM.	Built by IEFT group.
Requirements	Required for message transferring for M2M and IOT.	A document transfer protocol that is designed for constrained devices.
Basic Components	The basic components of MQTT are client and brokers.	The basic components of CoAP are client and server.
Data Transfer	Data transfer occurs on TCP.	Data transfer occurs on UDP.

Q2. Cognitive Sensor Networks:

(1) What is cognitive sensor networks?

Ans- The sensor network that use a distributed sensor actuator, computing and a communication system for the purpose of learn, reason to act are called cognitive sensor networks.

They are also called self aware networks which can also make mission and environment specific configurations.

(2) What makes them different from traditional sensor networks?

Ans- The difference between Cognitive sensor networks (CWSN) from traditional sensor network is that in case of CSWN, the nodes adjust their broadcast and reception settings in response to the radio environment. Also, the availability of spectrum sensing operation in CSWN also distinguishes it from the WSN which does not have this operation.

(3) Why are we interested in them?

Ans- We are interested in it because of its-

- Efficient spectrum utilization.
- Multiple channel utilization.
- Energy efficiency.
- Global operability.

(4) Briefly discuss challenges one faces in trying to realize cognition in IOT and WSN.

Ans- Security, False alarm, Detection, Miss detection-Probability and Privacy are one of the major challenges trying to realize cognition in IOT and WSN's.
Cognition in IOT devices and their increased usage has led to a huge demand of wireless bandwidth in a geographical region. Whereas the practical challenges include Fault intolerance, installation cost, topology modifications, channel selection, scalability to name a few.

(5) Explain what is meant by spectrum sensing

Ans- It is an important part of the cognitive radio's learning process and it can be defined as the procedure of monitoring the particular frequency band in order to detect the presence or absence of primary or key users. It is used for improving spectrum utilization and preventing harmful interference with licenced users.

(6) Identify three spectrum sensing techniques. Summarize each one of them briefly. If you are to combine two of them in one implementation to achieve improved performance, which two of the three you have identified would you choose? Justify why.

Ans-
The spectrum sensing techniques are:-

1. Transmitter detection or Non-corporative sensing or Local sensing.
2. Cooperative sensing
3. Interference based sensing

1. Local spectrum sensing
   It is basically a local secondary user based observation technique that detects whether the frequency band consists of a Primary user signal or not.
   Local spectrum sensing can be further divided into-
      a) Matched filter detection
      b) Energy detection
      c) Cyclostationary feature detection

2. Cooperative spectrum sensing
   The spectrum sensing technology that employs many detectors and combines their result to make an appropriate conclusion is called corporative spectrum sensing.
   It is further divided into two types-
   a) Centralized cooperative spectrum sensing
   b) Distributive cooperative spectrum sensing

3. Interference based spectrum sensing
   In this type, the secondary users are allowed to coexist with the primary users, they can also transmit lower power compared to primary users.
   In this technique, the interference is avoided by the interference temperature level.

In order to achieve enhanced performance, In order to achieve enhanced performance, cooperative spectrum sensing and interference based spectrum sensing can be combined in order prevent cognitive radio users to cause interference to the primary users. This would also result in the increase in the throughput and the quality of service.
Apart from this, Energy detection can also be combined with centralized cooperative sensing as energy detection suffers from noise uncertainty, so when combined with centralized cooperative spectrum sensing, it’s limitation will be compensated as multiple nodes transmit sensing information to a single centralized unit which will prevent interference between the nodes.

Q3. Time Synchronization

(A) Refer to TDOA localization in a three-dimensional space. Assume that five reference nodes are known at Node 1: (0,3,0), Node 2: (6,0,0), Node 3: (3,4,0), Node 4: (4, 3, 0), and Node 5: (0, 0, 8) respectively. Also, for node U, we know the following propagation times: U11 = 0s, U12 = 1s, U13 = 0.7s, U14 = 0.7s, & U15 = 1.7s. The velocity of propagation is v.

(i) Find the unknown location (xU , yU , zU).

(ii) Now assume that the propagation speed is known to be 8.7 m/s. Find the unknown location (xU , yU , zU)

- Handwritten notes for this question are attached at the bottom of each section.

Q-3) Time Synchronization

a) TDOA localization in 3D

5 Nodes : 1 (0,3,0)
2 (6,0,0)
3 (3,4,0)
4 (4,3,0)
5 (0,0,8)

For Node U, the propogation times :

$U_{11} = 0\,s$

$U_{12} = 1\,s$

$U_{13} = 0.7\,s$

$U_{14} = 0.7\,s$

$U_{15} = 1.7\,s$

(i) velocity of propogation = v

$$Aw = b$$

$$A = \begin{bmatrix} x_1 & y_1 & z_1 & -t_{11} & t_{11}^2/2 \\ x_2 & y_2 & z_2 & -t_{12} & t_{12}^2/2 \\ x_3 & y_3 & z_3 & -t_{13} & t_{13}^2/2 \\ x_4 & y_4 & z_4 & -t_{14} & t_{14}^2/2 \\ x_5 & y_5 & z_5 & -t_{15} & t_{15}^2/2 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 3 & 0 & 0 & 0 \\ 6 & 0 & 0 & -1 & 0.5 \\ 3 & 4 & 0 & -0.7 & 0.245 \\ 4 & 3 & 0 & -0.7 & 0.245 \\ 0 & 0 & 8 & -1.7 & 1.445 \end{bmatrix}$$

$$r_i^2 = x_i^2 + y_i^2 + z_i^2$$

$$b = \frac{1}{2}\begin{bmatrix} r_1^2 \\ r_2^2 \\ r_3^2 \\ r_4^2 \\ r_5^2 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 9 \\ 36 \\ 25 \\ 25 \\ 64 \end{bmatrix} = \begin{bmatrix} 4.5 \\ 18 \\ 12.5 \\ 12.5 \\ 32 \end{bmatrix}$$

$$w = \begin{bmatrix} x_u \\ y_u \\ z_u \\ V_{si} \\ V^2 \end{bmatrix}$$

$$A w = b$$

$$\begin{bmatrix} 0 & 3 & 0 & 0 & 0 \\ 6 & 0 & 0 & -1 & 0.5 \\ 3 & 4 & 0 & -0.7 & 0.245 \\ 4 & 3 & 0 & -0.7 & 0.245 \\ 0 & 0 & 8 & -1.7 & 1.445 \end{bmatrix} \begin{bmatrix} x_u \\ y_u \\ z_u \\ V_{si} \\ V^2 \end{bmatrix} = \begin{bmatrix} 4.5 \\ 18 \\ 12.5 \\ 12.5 \\ 32 \end{bmatrix}$$

$$3y_u = 4.5 \quad , \boxed{y_u = 1.5} \quad - (1)$$

$$6x_u - V_{si} + 0.5\,V^2 = 18 \quad - (2)$$

$$3x_u + 4y_u - 0.7(V_{si}) + 0.245\,V^2 = 12.5 \quad - (3)$$

$$4x_u + 3y_u - 0.7(V_{si}) + 0.245\,V^2 = 12.5 \quad - (4)$$

$$8z_u - 1.7(V_{si}) + 1.445\,V^2 = 32 \quad - (5)$$

From eq^n (3) and (4)

$$\begin{array}{l} 3x_u + 4y_u \cancel{- 0.7(V_{si})} \cancel{+ 0.245V^2} = \cancel{12.5} \\ 4x_u + 3y_u \cancel{- 0.7(V_{si})} \cancel{+ 0.245V^2} = \cancel{12.5} \\ - \quad\;\; - \quad\;\; + \quad\quad\quad - \quad\quad\; - \\ \hline -x_u + y_u = 0 \end{array}$$

$$\boxed{x_u = y_u = 1.5}$$

Multiplying 0.7 in eq^n ② and subtracting with eq^n ③

$$4.2\,xu - \cancel{0.7\,vsi} + 0.35v^2 = 12.6$$
$$-\;4yu + 3\,xu - \cancel{0.7\,vsi} + 0.245v^2 = 12.5$$
$$-4yu + 1.2xu + 0.105v^2 = 0.1$$
$$-4(1.5) + 1.2(1.5) + 0.105v^2 = 0.1$$
$$0.105\,v^2 = 0.1 - 1.8 + 6$$
$$v^2 = 40.952$$
$$\boxed{v = 6.40}$$

Substituting value of v in eq^n ②

$$6(1.5) - 6.40\,si + 0.5\,(40.95) = 18$$
$$6.40\,si = 9 + 20.475 - 18$$
$$\boxed{si = 1.80}$$

Keeping value of v, and si in eq^n ⑤

$$8\,zu - 1.7\,(6.40)(1.80) + 1.445v^2 = 32$$
$$8zu = -7.591$$
$$\boxed{zu = -0.948}$$

So the position (location) of Node U is $(xu, yu, zu) = (1.5, 1.5, -0.948)$

(ii) $v = 8.7$ m/s

$v^2 = 75.69$ m/s

$$Aw = b + dv^2 \qquad w = \begin{bmatrix} xu \\ yu \\ zu \\ vsi \end{bmatrix}$$

$$A = \begin{bmatrix} x_1 & y_1 & z_1 & -t_{11} \\ x_2 & y_2 & z_2 & -t_{12} \\ x_3 & y_3 & z_3 & -t_{13} \\ x_4 & y_4 & z_4 & -t_{14} \\ x_5 & y_5 & z_5 & -t_{15} \end{bmatrix} = \begin{bmatrix} 0 & 3 & 0 & 0 \\ 6 & 0 & 0 & -1 \\ 3 & 4 & 0 & -0.7 \\ 4 & 3 & 0 & -0.7 \\ 0 & 0 & 8 & -1.7 \end{bmatrix}$$

$$d = \frac{1}{2}\begin{bmatrix} t_{11}^2 \\ t_{12}^2 \\ t_{13}^2 \\ t_{14}^2 \\ t_{15}^2 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.5 \\ -0.245 \\ -0.245 \\ -1.445 \end{bmatrix}$$

$$Aw = b + dv^2$$

$$\begin{bmatrix} 0 & 3 & 0 & 0 \\ 6 & 0 & 0 & -1 \\ 3 & 4 & 0 & -0.7 \\ 4 & 3 & 0 & -0.7 \\ 0 & 0 & 8 & -1.7 \end{bmatrix} \begin{bmatrix} x_u \\ y_u \\ z_u \\ V_{si} \end{bmatrix} = \begin{bmatrix} 4.5 \\ 18 \\ 12.5 \\ 12.5 \\ 32 \end{bmatrix} + \begin{bmatrix} 0 \\ -37.84 \\ -18.54 \\ -18.54 \\ -109.37 \end{bmatrix}$$

$$\begin{bmatrix} 3y_u \\ 6x_u - V_{si} \\ 3x_u + 4y_u - 0.7V_{si} \\ 4x_u + 3y_u - 0.7V_{si} \\ 8z_u - 1.7V_{si} \end{bmatrix} = \begin{bmatrix} 4.5 \\ -19.84 \\ -6.04 \\ -6.04 \\ -77.37 \end{bmatrix} \begin{matrix} -(1) \\ -(2) \\ -(3) \\ -(4) \\ -(5) \end{matrix}$$

$$3y_u = 4.5$$

$$\boxed{y_u = 1.5}$$

From (3) and (4)

$$3x_u + 4y_u \cancel{- 0.7V_{si}} = \cancel{-6.04}$$

$$4x_u + 3y_u \cancel{- 0.7V_{si}} = \cancel{-6.04}$$

$$- \quad - \quad + \quad +$$

$$-x_u + y_u = 0$$

$\boxed{xu = yu = 1.5}$

Will substitute value of xu in eqn ②

$6\,xu - Vsi = -19.84$

$6(1.5) + 19.84 = Vsi$

$\boxed{Vsi = 28.84}$

Substituting value of Vsi in eqn ⑤

$8\,Zu - 1.7\,Vsi = -77.37$

$8\,Zu = -77.37 + 1.7(28.84)$

$8\,Zu = -28.342$

$\boxed{Zu = -3.542}$

So the location of Node U,
when ~~V is~~ V = 8.7 m/s is
$(xu, yu, Zu) = (1.5, 1.5, -3.542)$

(B) Node A sends a synchronization request to node B at 3150 (on node A's clock). At 3250, node A receives the reply from node B with a time stamp of 3120.

(i) What is node A's clock offset with respect to the time at node B (you can ignore any processing delays at either node)?

(ii) Is node A's clock going too slow or too fast?

(iii) How should node A adjust its clock?

Q-3> b> Two Nodes

A, $t_1 = 3150$ $\xrightarrow{\text{SYNC}}$ B, $t_2 = 3120$

A, $t_4 = 3250$ $\xleftarrow{\text{Reply}}$ B, $t_3 = 3120$

(i) Node A clock offset w.r.t time at node B

$$\text{Clock drift} = \Delta = \frac{(T_2 - T_1) - (T_4 - T_3)}{2}$$

$$= \frac{(3120 - 3150) - (3250 - 3120)}{2}$$

$$= \frac{-30 - 130}{2} = \frac{-160}{2}$$

$$= -80$$

So Node A's clock offect is - 80 sec.

(ii) Is Nodes A's clock going too slow or too fast?

- It is going slow as the offset value is - 80 Sec. so it is slow by 80 sec.

(iii) How should Node A adjust its clock?

- Here if it add's 80 second to it's clock, it will adjust the clock as it is slow by 80 sec.

(C) **Two nodes A and B use RBS to receive periodic acoustic synchronization signals from a reference node. Node A's clock shows 10 s when it receives the last synchronization beacon, while node B's clock shows 15 s. Node A detects an event at time 15 s, while node B detects the same event at time 19.5 s. Assume that node A is 100 m away from the synchronization source and node B is 400 m away from the synchronization source. Which node detected the event sooner and by how much? Assume a signal speed of 300 m/s.**

Q-3) c)

- At the time of synchronization, Node B's clock showed 15 sec and Node A's clock showed 10 sec

So we can say,

clock of B = clock of A + 5 sec

B = A + 5

- B detects an event at 19.5 sec so according to that A should detect same event at:

$B = A + 5$

$A = B - 5$

$= 19.5 - 5$

$= 14.5$ sec

But A detected same event at 15 sec

$= 15 - 14.5$

$= 0.5$ sec

So actually A is detecting ~~0.5~~ 0.5 sec later

- B received 0.5 sec earlier than A.

Now, A is 100 m away from Sync
B is 400 m away from Sync

So Sync has to travel additional 300 m for Node B.

This takes the sound $= \frac{300}{300}$

$= 1$ sec

~~Will subtract~~ ~~earlier detecte~~d ~~this time by 1 sec~~

Will subtract ~~this~~ time by earlier detected time

$= 1$ sec $- 0.5$ sec
$= 0.5$ sec

So Node B heard 0.5 sec earlier than Node A.

(D) TOA requires that all the reference nodes and the receiver have precise synchronized clocks and the transmitted signals be labeled with time stamps. TDOA measurements remove the requirement of an accurate clock at the receiver. Assume that five reference nodes have known positions (0, 0),(−1, −1),(0, 1),(3, 1), and (1, 4) respectively. We choose (0, 0) as the reference sensor for differential time-delays which are defined as $t_{1r} = t_1 - t_r = r_{s1} - r_{s2}\ v$ where v is the velocity of propagation, r_{si} is the distance between the unknown node and the i th node. Further assume that $t_{12} = -1.4$s, $t_{13} = 0.4$s, $t_{14} = -1.6$s, and $t_{15} = -2.6$s (a) Find the unknown location (x_t, y_t). (b) Now assume that the propagation speed is known as 1.8 m/s. Find the unknown location (x_t, y_t).

Q-3) d) TDOA ~~1(0,0)~~ Reference Node 1 (0, 0)

Node 2 (−1, −1)

3 (0, 1)

4 (3, 1)

5 (1, 4)

$$t_{1r} = t_1 - t_r = \frac{r_{s1} - r_{s2}}{v}$$

$t_{12} = -1.4s$ $\quad t_{13} = 0.4s$ $\quad t_{14} = -1.6s$

$t_{15} = -2.6s$

(i) Find unknown location (x_t, y_t)

$$A = \begin{bmatrix} x_2 & y_2 & -t_{12} & -t_{12}^2/2 \\ x_3 & y_3 & -t_{13} & -t_{13}^2/2 \\ x_4 & y_4 & -t_{14} & -t_{14}^2/2 \\ x_5 & y_5 & -t_{15} & -t_{15}^2/2 \end{bmatrix} \quad W = \begin{bmatrix} x_t \\ y_t \\ v s_i \\ v^2 \end{bmatrix}$$

$$A = \begin{bmatrix} -1 & -1 & 1.4 & -0.98 \\ 0 & 1 & -0.4 & -0.08 \\ 3 & 1 & 1.6 & -1.28 \\ 1 & 4 & 2.6 & -3.38 \end{bmatrix}$$

$$b = \frac{1}{2} \begin{bmatrix} r_2^2 \\ r_3^2 \\ r_4^2 \\ r_5^2 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 2 \\ 1 \\ 10 \\ 17 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.5 \\ 5 \\ 8.5 \end{bmatrix}$$

$$AW = b$$

$$W = A^{-1} b$$

$$A^{-1} = \begin{bmatrix} 0.49 & 1.54 & 0.63 & -0.42 \\ -1.85 & -3.86 & -0.94 & 0.98 \\ -3.66 & -9.82 & -1.89 & 2.01 \\ -4.87 & -11.67 & -2.38 & 2.29 \end{bmatrix}$$

$$w = A^{-1} b$$

$$\begin{bmatrix} \cancel{xt} \\ yt \\ vsi \\ v2 \end{bmatrix} = \begin{bmatrix} 0.49 & 1.54 & 0.63 & -0.42 \\ -1.85 & -3.86 & -0.94 & 0.98 \\ -3.66 & -9.82 & -1.89 & 2.01 \\ -4.87 & -11.67 & -2.38 & 2.29 \end{bmatrix} \begin{bmatrix} 1 \\ 0.5 \\ 5 \\ 8.5 \end{bmatrix}$$

$$\begin{bmatrix} xt \\ yt \\ vsi \\ v2 \end{bmatrix} = \begin{bmatrix} 0.84 \\ -0.15 \\ -7.97 \\ -3.14 \end{bmatrix}$$

So, the location of unknown $(xt, yt) = (0.84, -0.15)$

(ii) $v = 1.8\ m/s$

$v^2 = 3.24$ $\cancel{m/s}$

$$w = \begin{bmatrix} xt \\ yt \\ vsi \end{bmatrix}$$

$$\cancel{Aw = b + dv^2}$$

$$w = (A^T A)^{-1} A^T \times b + (A^T A)^{-1} A^T d\, v^2$$

$$A = \begin{bmatrix} x_2 & y_2 & -t_{12} \\ x_3 & y_3 & -t_{13} \\ x_4 & y_4 & -t_{14} \\ x_5 & y_5 & -t_{15} \end{bmatrix} = \begin{bmatrix} -1 & -1 & 1.4 \\ 0 & 1 & -0.4 \\ 3 & 1 & 1.6 \\ 1 & 4 & 2.6 \end{bmatrix}$$

$r_i^2 = x_i^2 + y_i^2$ $\cancel{+ \cdots}$

$$b = \frac{1}{2}\begin{bmatrix} r_{12}^2 \\ r_{13}^2 \\ r_{14}^2 \\ r_{15}^2 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 2 \\ 1 \\ 10 \\ 17 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.5 \\ 5 \\ 8.5 \end{bmatrix}$$

$$A = \begin{bmatrix} -1 & -1 & 1.4 \\ 0 & 1 & -0.4 \\ 3 & 1 & 1.6 \\ 1 & 4 & 2.6 \end{bmatrix}$$

$$A^T A = \underset{3\times 4}{\begin{bmatrix} -1 & 0 & 3 & 1 \\ -1 & 1 & 1 & 4 \\ 1.4 & -0.4 & 1.6 & 2.6 \end{bmatrix}} \underset{4\times 3}{\begin{bmatrix} -1 & -1 & 1.4 \\ 0 & 1 & -0.4 \\ 3 & 1 & 1.6 \\ 1 & 4 & 2.6 \end{bmatrix}}$$

$$= \underset{3\times 3}{\begin{bmatrix} 11 & 8 & 6 \\ 8 & 19 & 10.2 \\ 6 & 13 & 11.44 \end{bmatrix}}$$

$$(A^T A)^{-1} = \begin{bmatrix} 0.13 & -0.02 & -0.05 \\ -0.04 & 0.14 & -0.10 \\ -0.01 & -0.15 & 0.23 \end{bmatrix}$$

$$(A^T A)^{-1} A^T = \underset{3\times 3}{\begin{bmatrix} 0.13 & -0.02 & -0.05 \\ -0.04 & 0.14 & 0.10 \\ -0.01 & 0.15 & 0.23 \end{bmatrix}} \times$$

$$\underset{3\times 4}{\begin{bmatrix} -1 & 0 & 3 & 1 \\ -1 & 1 & 1 & 4 \\ 1.4 & -0.4 & 1.6 & 2.6 \end{bmatrix}}$$

$$= \begin{bmatrix} -0.18 & 0 & 0.29 & -0.08 \\ -0.24 & 0.18 & -0.14 & 0.26 \\ 0.482 & -0.242 & 0.188 & -0.012 \end{bmatrix}_{3\times 4}$$

$(A^TA)^{-1} A^T \times b =$

$$= \begin{bmatrix} -0.18 & 0 & 0.29 & -0.08 \\ -0.24 & 0.18 & -0.14 & 0.26 \\ 0.482 & -0.242 & 0.188 & -0.012 \end{bmatrix}_{3\times 4} \begin{bmatrix} 1 \\ 0.5 \\ 5 \\ 8.5 \end{bmatrix}_{4\times 1}$$

$$\cancel{\begin{bmatrix} x_t \\ y_t \\ v_{si} \end{bmatrix} = \begin{bmatrix} 0.59 \\ 1.36 \\ 1.999 \end{bmatrix}_{3\times 1}} = \begin{bmatrix} 0.59 \\ 1.36 \\ 1.99 \end{bmatrix}_{3\times 1}$$

~~So, the unknown location is~~

For
$(A^TA)^{-1} A^T \times dv^2$

$$d = \frac{1}{2} \begin{bmatrix} t_{12}^2 \\ t_{13}^2 \\ t_{14}^2 \\ t_{15}^2 \end{bmatrix} = \begin{bmatrix} -0.98 \\ -0.08 \\ -1.28 \\ -3.38 \end{bmatrix}$$

$$dv^2 = \begin{bmatrix} -3.17 \\ -0.25 \\ -4.14 \\ -10.95 \end{bmatrix}$$

$$(A^T A)^{-1} A^T \times dv^2 = \begin{bmatrix} -0.18 & 0 & 0.29 & -0.08 \\ -0.24 & 0.18 & -0.14 & 0.26 \\ 0.482 & -0.242 & 0.188 & -0.012 \end{bmatrix}$$

$$\times \begin{bmatrix} -3.17 \\ -0.25 \\ -4.14 \\ -10.95 \end{bmatrix}$$

$$= \begin{bmatrix} 0.24405 \\ -1.5525 \\ -2.1134 \end{bmatrix}$$

$$W = (A^T A)^{-1} A^T \times b + (A^T A)^{-1} A^T \times dv^2$$

$$= \begin{bmatrix} +0.59 \\ +1.36 \\ 1.99 \end{bmatrix} + \begin{bmatrix} 0.24 \\ -1.55 \\ -2.11 \end{bmatrix}$$

$$\begin{bmatrix} x_t \\ y_t \\ V_{si} \end{bmatrix} = \begin{bmatrix} 0.83 \\ 0.19 \\ -0.92 \end{bmatrix}$$

So unknown location is

$$(x_t, y_t) = (0.83, 0.19)$$

Q5. Coverage Control

1. **(CTR/topology for connectivity:6 points) A network of wireless IOT devices is to be deployed uniformly in a (1,000 meters by 1,000 meters) area to collect data and to control the switching of a set of lights..**

(A) Make the necessary assumptions and determine the minimum connectivity transmission range.

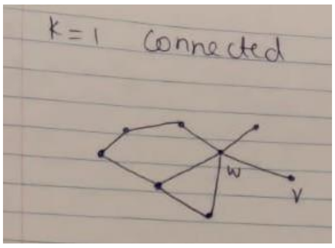
Suppose n nodes are placed in a certain region R = $[0,1]^2$. The deployment region is unit square and n points are distributed uniformly at random in R, then the CTR (Critical Transmitting Range) for connectivity is $r_c = \sqrt{\frac{\log n + f(n)}{n\pi}}$ where f(n) is an arbitrary function such that $\lim_{n\to\infty} f(n) = +\infty$. As n gets larger, the CTR decreases. If we calculate the minimum connectivity transmission range in theoretical way, $f(n) = \log(\log n)$ so $r_c = \sqrt{\frac{\log n + \log(\log n))}{n\pi}}$. So, the minimum connectivity transmission range is $\lim_{n\to\infty} r_c = \lim_{n\to\infty} \sqrt{\frac{\log n + \log(\log n))}{n\pi}}$. Therefore, if we assume n is 1000, r_c =0.1045km, if n goes to infinity, r_c can be 0 in theoretically.

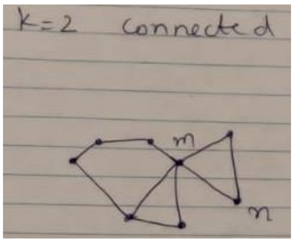
(B) Explain the network attribute that it is k-connected.

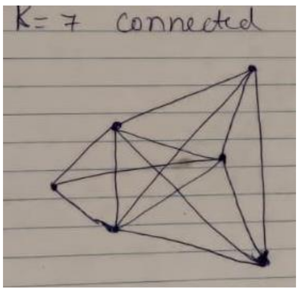
The attribute of a k-connected network is that all nodes in the network are still connected to each other when some nodes (that should be lower than k) are removed.

(C) Draw the k-connectivity probability profile for the network. What is the probability that the network is 1-connected, 2-connected, and 7-connected?

Assume n nodes, each with transmitting range r_0, are distributed uniformly at random in a very large area A. Then, the probability that the minimum node degree in the communication graph is at least k for some $1 \le k < n$, is closely approximated by $P(deg_{min} \ge k) \approx (1 - \sum_{i=0}^{k-1} \frac{(\rho\pi r_0)^i}{i!} e^{-\rho\pi r_0^2})^n$, where $\rho = \frac{n}{A}$ which is the number of nodes per unit of area. If we assume n=1000, the probability of 1-connected network is 0.033725, 2-connected network is 0.059925, 7-connected network is 0.0612303.







2. Topology for coverage

(A) What is the difference between the detection optimal coverage and estimation optimal coverage?

In the detection optimal coverage, the detection probability measures its sensing quality of sensors. The detection probability of a space point by a single sensor is related to the distance between them while it is also related to other factors.
Otherwise, in the estimation optimal coverage, estimation errors measure its sensing quality of sensors. The estimation error of a space point by a single sensor is related to the distance between them while it is also related to other factors.
So, the main difference between the detection optimal coverage and estimation optimal coverage is the method to represent its sensing quality. Detection optimal coverage uses detection probability, while estimation optimal coverage uses estimation errors.

(B) In the class we studied one scenario of optimal sensor placement, as follows: A generalization of the ILP problem is to minimize the network cost if different sensor nodes have different costs and coverage capabilities. Sometimes, we may have other constraints such as the distance between any pair of sensor nodes should not be less than a certain min distance.
For example, in the scenario of placing different types of sensors, suppose that we have A types of sensors, each type with cost c_a and coverage distance r_a, $a = 1, \ldots, A$. Normally, a larger coverage range corresponds to a larger cost. Let $D_a(i)$ denote the set of targets that can be covered by a type a sensor placed at site i, i.e.,

$$D_a(i) = \{j \mid d(i,j) \le r_a\}, i = 1, \ldots, I$$

Where $d(i,j)$ is the Euclidean distance between a site i and a target j. Again, we use $x_i^a = 1$ to denote that a type a sensor been placed at site i and $x_i^a = 0$ otherwise. Furthermore, the coverage requirement is to cover with at least k sensors. The objective is to minimize the total sensor placement cost
Minimize :

$$\sum_{i=1}^{I}\sum_{a=1}^{A} c_a x_i^a$$

Subject to :

$$\sum_{a=1}^{A}\sum_{j\in D_a(i)} x_i^a \ge k, \qquad i = 1, \ldots, I$$

$$\sum_{a=1}^{A} x_i^a \leq 1, i = 1, \ldots, I$$

As the given information, $c_1 = 300$, $c_2 = 170, c_3 = 65$, and there are 17 targets. These targets are randomly and uniformly positioned within the boundaries of the surveillance area (500mx500m).

Answer-

As we are given in the above problem that are 17 targets. In addition to that we are given the surveillance area from which we get that Layout width is 500, the Layout height is also 500.

i. Coverage is 2 Nos.

ii. For Sensors –

   a) S1 type sensors– 7 Nos.

   b) S2 type sensors– 17 Nos.

   c) S3 type sensors – 77 Nos.

In case of Optimization parameters-

a) Initial temperature is taken as 85

b) Final temperature is taken as 5

c) Value of alpha is taken as 0.65

For the senor model, in order to define its parameters, the code is as follows-

File Edit View Run Kernel Settings Help

```python
class Type_sens(enum.Enum):
    sens_1 = 1  # Cost of type S1 is 300
    sens_2 = 2  # Cost of type S2 is 170
    sens_3 = 3  # Cost of type S3 is 65


class Range_sens(enum.Enum):
    sens_1 = 100
    sens_2 = 70
    sens_3 = 30


class Cost_sens(enum.Enum):
    sens_1 = 300
    sens_2 = 170
    sens_3 = 65


class Target:
    def __init__(self, width, height, targetId):
        self.pos_x = random.randint(0, width)
        self.pos_y = random.randint(0, height)
        self.target_id = targetId
        self.coverage = 0
        self.coverage_list = []
        pass


class Sensor:

    def __init__(self, width, height, sensorType, Range_sens, Cost_sens, sensorId):
        self.sensor_type = sensorType
        self.sensor_id = sensorId
        self.sensor_cost = Cost_sens
        self.pos_x = random.randint(0, 500)
        self.pos_y = random.randint(0, 500)
        self.range = Range_sens
        self.coverage = 0
        self.coverage_list = []
        self.sensor_present = True
        self.sensor_area = round((math.pi * ((Range_sens)**2)), 2)
        self.used_status = True
        pass
```

In order to generate 17 sensors around the square area of (500*500)

File Edit View Run Kernel Settings Help

```python
    def target_gen(self):
        for a in range(0, self.target_count):
            target = Target(self.width, self.height, a)
            target_pos = [target.pos_x, target.pos_y]
            target_dict[a] = tuple(target_pos)
            target_dict_type[a] = target
            target_data.append(target)
```

For simulated Annealing the following code has been used-

File Edit View Run Kernel Settings Help

Code

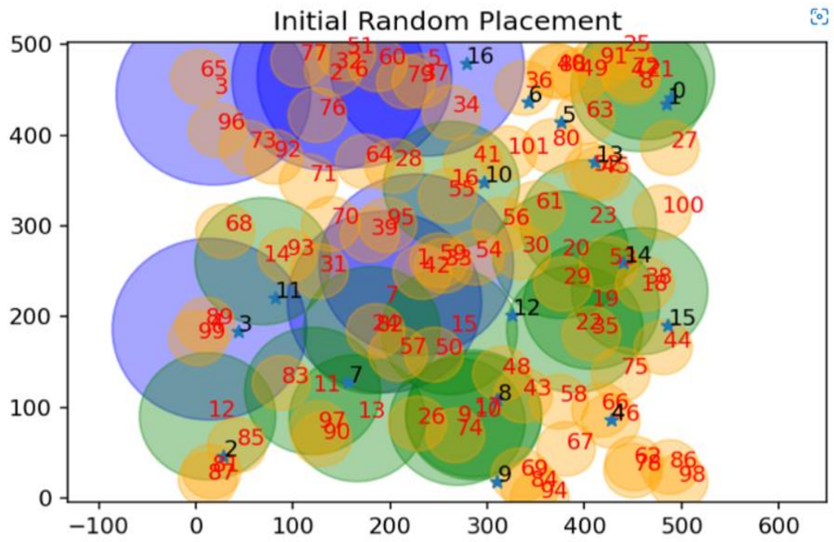
```python
    def optmize_sa(self, sensor_dictionary, targetvalue):
        init_temp = 85
        final_temp = 5
        no_of_iterations = 0
        alpha = 0.65
        current_temp = init_temp
        cost_trend = []
        atleast = 0
        while current_temp > final_temp:
            no_of_iterations += 1
            self.calc_coverage(sensor_dictionary, targetvalue, False)
            prev_cost = self.cost_function(sensor_dictionary, targetvalue)
            random_sensor = random.choice(list(sensor_dictionary.keys()))
            if sensor_dictionary[random_sensor].used_status == True:
                sensor_dictionary[random_sensor].sensor_present = False
                cov_list = self.calc_coverage(
                    sensor_dictionary, targetvalue, True)
                if (len(cov_list)/self.target_count)*100 >= 99:
                    new_cost = self.cost_function(sensor_dictionary, targetvalue)
                    cost_difference = (prev_cost - new_cost)
                    coverage_condition = True
                    for a in targetvalue.items():
                        if len(a[1].coverage_list) <= self.k:
                            pass
                        else:
                            coverage_condition = False
                    if coverage_condition:
                        if cost_difference > 0:
                            atleast += 1
                            print("No of Iter_"+str(no_of_iterations) +
                                  "----"+str(new_cost))
                            cost_trend.append(new_cost)
                        else:
                            if random.uniform(0, 1) < math.exp(-cost_difference / current_temp):
                                sensor_dictionary[random_sensor].sensor_present = True
                        current_temp -= alpha
                    else:
                        sensor_dictionary[random_sensor].sensor_present = True
                        if ((no_of_iterations > 10000) & (atleast == 0)):
                            break
                else:
                    sensor_dictionary[random_sensor].sensor_present = True
        if atleast == 0:
            print("Unable to optimize for coverage : "+str(self.k))
```

Code

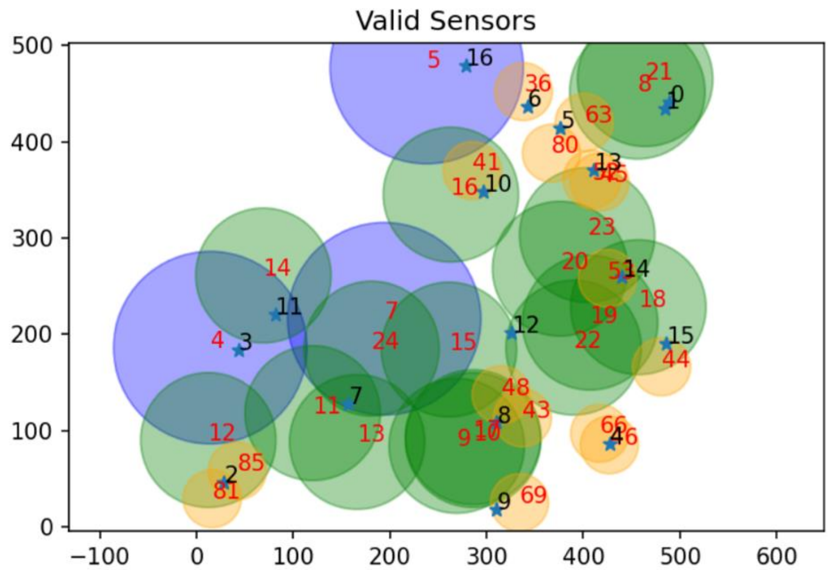
```python
        if atleast == 0:
            print("Unable to optimize for coverage : "+str(self.k))
        else:
            saName = "After_SA"
            generate_graph(cost_trend, 'SA_Swap_Trend', 'SA Swap Trend', True)
            generate_map(sensor_dictionary, targetvalue, saName, True, 'SA Placement')
            print("Final Cost "+"----" +
                  str(self.cost_function(sensor_dictionary, targetvalue)))
        pass
```

Graphs-

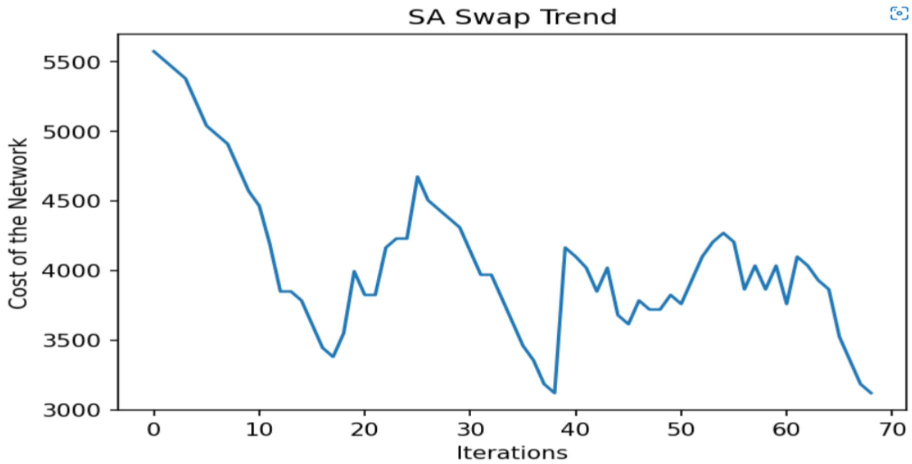
Initially the random placement of sensors is displayed as-



After removing the invalid sensors-



After simulated annealing-



The final optimal node placement is displayed as-