Stensor: A Novel Stochastic Algorithm for Placement of Sensors in a Rectangular Grid

Category: Computer Science, Team ID: eu5242

1 Introduction

The sensors are used to monitor certain parameters over a wide range of area, over an extensive period of time. For example in order to forecast the weather we need to monitor large areas, for extended periods of time. These regions are often remote and unsafe for human survey. Hence, the sensors have gained a lot of importance in various geological and earth-survey [3]. Some recent study [1] made on the birth of cyclones (e.g. Rita [2]) and the tsunamis revealed something more. It is a non-trivial task to predict the exact location of the formation of these catastrophies. The development of such turmoils in the weather are usually caused by certain basic mutually re-inforcing factors like wind speed, temperature, pressure etc. Thus it is absolutely essential that slightest fluctuations in those parameters get reported. For observation of those parameters of weather, deployment of sensors in large numbers is necessary. When large number of sensors are deployed which are meant to work in a collaborative manner we call this kind of network-*sensor network*. An extensive survey on the applications of the sensor networks can be found in [4].

The time and spatial location of formation of those catastrophies are found to be random in most cases. This randomness inhibits planned deployment of sensors. To counter the unpredictable behavior of the various forces of nature, it is important to have some stochastic strategy of placement of the sensors. Placing the sensors is not an easy task, since with the increment of the number of sensors deployed in the region, the other aspects of the sensor-networks like fault tolerance, period over which battery exhausts, etc pose several critical questions. Much of these parameters are largely affected by the physical placement of the sensor-nodes. The placement should be attempted in such an optimal way, so that the performance and efficiency is maximized.

Placement of sensors is a well studied field [5, 7]. Research till date has been primarily focussed towards deterministic placement of the sensors. As the actual placement involves lot of cost, simulation of sensor networks is inevitable to measure the expected performance. Thus, for the sake of simulation, it is important to find a suitable algorithm for the placement of sensor-nodes in an area, before physical deployment. In this paper, we present a *stochastic* algorithm-*Stensor* which addresses the following problem: *Given a rectangular field, the task is to place the sensors in a random fashion, such that it can monitor randomly generating targets consuming lesser power without compromising with the performance.*

2 Related Works

So far much of the recent research has taken place in devising the deterministic algorithms [7, 8] for optimal sensor placement. But, when the number of sensors grow as well the area becomes larger, these kind of deterministic algorithms remain no more feasible[6]. With this motivation,

the authors in [6] proposed a stochastic algorithm for the placement problem in sensor network. Their communication model is as follows. They assume that they do not know the location of the target in advance and the sensor nodes can exchange data with the sensor nodes which are within the radio transmission range. Moreover, they considered a base station in vicinity of the sensors to which the data is to be reported. Traditionally, there are three typical variations in stochastic sensor-placement.

- **Simple Diffusion:** The simplest way to distribute the sensor nodes is to scatter them from the air. This is known as the simple diffusion. This diffusion takes place centering the base station assuming that the sensor nodes would report to the base station either directly (if it is within the range of signal transmission), or else it routs the information indirectly to the base-station.

- **Constant Diffusion:** Sensor nodes are also placed so that their density is constant. Such random distributions are called as constant diffusion. But this distribution is also meaningful in the context of reporting of information through a base-station located in the vicinity.

- **R-random placement:** Ishizuka and Aida proposed , this distribution technique in [6]. In this placement technique they proposed random scattering of the sensor nodes with decreasing density radially outward, from the center. Their communication model is also the similar to that mentioned in the earlier diffusion mode.

In the following paragraph, we describe the motivation of proposing this new algortihm. As the area of application of sensor networks (in the context of weather forecasting etc.) is usually *remote*, the sensor network is supposed to remain active for longer periods of time. Hence the sensor-network should consume lesser power. However, deficit of power might cause undesirable delay in transmission of the observed changes in the environment. Moreover, this delay in observation of weather elements might lead to lack of appropriate data for prediction of weather, which in turn might have **catastrophic consequences**. None of the previous works considers this *power-factor*. Thus, these may fail in **safety-critical applications**. To address this problem we have proposed a new sensor-placement algorithm with the following functionalities.

- *Constrained randomness*: The placement is inherently random but bound to certain constraints attempting nearly even distribution.

- *Power Efficiency*: The sensors placed according to our methodology consume lesser power compared to the state-of-the-art aproaches. This enhances the *active network-lifetime*.

- *Easy Replenishment Option:* Even if the network stops from performing (due to power deficit), it is easier to replenish the network by deploying minimal number of additional sensors in the region, compared to other approaches discussed in next section.

The experimental results also show that the algorithm is not compromising with the performance.

3 Communication and Deployment Model

The context of application of any algorithm must be clearly specified. In our case, first, we describe the communication model and the relevant assumptions for which the algorithm has been devised.

3.1 The Communication model

The communication model is as follows.

1. This algorithm considers those applications where we do not know the location of the target beforehand. This is practical when we want to sense some parameter of an unknown area.
2. The ratio of sensing radius to the radio-transmission radius is small.

3.2 The Deployment Model

The deployment model is as follows.

1. We target applications where, a large number of sensors are deployed over a wide area. Moreover, (for discrete analysis) the area over which the sensor nodes are distributed can be further classified into small cells (or pixels).
2. *Each cell cannot host more than one sensor node in it.*
3. The ratio of number of sensors to the number of cells in the region is pretty *small.*
4. We follow random distribution, as it is not possible for somebody to place the sensors in pre-determined locations.

We want to simulate this situation in our proposed model. In our case we tried to simulate with *poisson distribution.* In the following subsection, we present an intuitive reasoning behind the choice of the distribution.

3.3 Why Poissan Distribution?

Let us denote N as the number of cells in the rectangular grid. Also, let the number of sensor-nodes be λ. It is reasonable to assume that $\lambda \ll N$, since the number of sensor-nodes are usually fewer compared to the whole area. The probability (p) for each cell containing a sensor is given by,

$$p = \frac{\lambda}{N} \approx 0 \quad (1)$$

$$\Rightarrow Np = \lambda \quad (2)$$

Therefore, the fact that although N is a large value and p is having a very small value, their product ($= \lambda$) is moderate, hints that the distribution follows a **Poisson Distribution**.

4 The Stochastic Algorithm

From our assumption it is clear that the sensor-nodes are distributed more or less uniformly in the region. We try to model this situation using the *Poisson Distribution.* We state the algorithm, followed by an example and then the analysis.

4.1 Algorithm

Divide the whole region into $\lfloor\sqrt{\lambda}\rfloor$ strips. Next, consider that there are $\lfloor\sqrt{\lambda}\rfloor$ buckets, where the λ sensor-nodes are to be inserted. Denote a random variable X_j for each strip. X_j denotes the

number of the sensor-nodes in j^{th} strip. Here, the distribution of X_j is governed by the Poisson Distribution. Therefore,

$$X_j \sim P(\sqrt{\lambda})$$

Therefore, the *probability distribution function* (p.d.f.) is given by,

$$\Pr(X_j = x) = \frac{e^{-\sqrt{\lambda}} \lambda^{\frac{x}{2}}}{x!}$$

We distribute the sensor-nodes in the buckets according to the *p.d.f.*. Virtually the sensor-nodes are divided in the $\sqrt{\lambda}$ partitions. Now, let j^{th} partition contains x_j sensor-nodes, where suppose, $\lfloor\sqrt{x_j}\rfloor$>1. We again formulate the problem of the distribution of those x_j sensor-nodes inside the j^{th} strip, as earlier problem. So, again divide the j^{th} strip into $\lfloor\sqrt{x_j}\rfloor$ partitions and repeat the process for each partition until $\lfloor\sqrt{x_j}\rfloor > 1$ holds in the recursively subdivided partitions. Lastly, place the sensor-nodes randomly in the allotted strips. Next we present the formal algorithm.

Algo. 4.1 PlaceSensor

PlaceSensor(N, λ)
begin
1: if ($\lfloor\sqrt{\lambda}\rfloor$=1) randomly place the sensors and return.
2: else partition the area into $\sqrt{\lambda}$ partitions and let, N_j be the no. of cells in j^{th} partition.
3: for each j^{th} partition
3.1: Obtain a random number (r) such that, $r \in [0, 1]$
3.2: Assign $X_j \leftarrow x$, such that, $\Pr(X_j \leq x) \leq r < \Pr(X_j \leq x + 1)$.
3.3: PlaceSensor(N_j, X_j).
3.3: endfor
end

4.2 Computational Aspect of Stochastic Allotment

In this paragraph, we describe how to use the pseudo-random number generator to obtain the value of x_j. Suppose, the pseudo-random number generator generates a value(=r) within [0, 100]. Therefore, we get a value within [0, 1] by dividing the obtained random number by 100. Let this number be the cumulative frequency such that,

$$\Pr(X_j \leq x) \leq \frac{r}{100} < \Pr(X_j \leq x + 1)$$

In this case, we assign $X_j = x$.

With reference to the Fig 1, we find the obtained random cumulative value is between $\Pr(X_j \leq 8)$ and $\Pr(X_j \leq 9)$. So the strip gets the random value of 8 here.

4.3 An Example

Let us consider this example where, $\lambda = 16$ and $N = 1000$. So $\sqrt{\lambda} = 4$ and $p = 0.016$. Hence, $X_j \sim P(4)$. So the whole plot is distributed into 4 strips. We apply the first round of iteration to get 4 sensors (say), in the strip 1. Next, the strip 1 is further subdivided into 2 partitions and final random placement is shown in Fig 2.

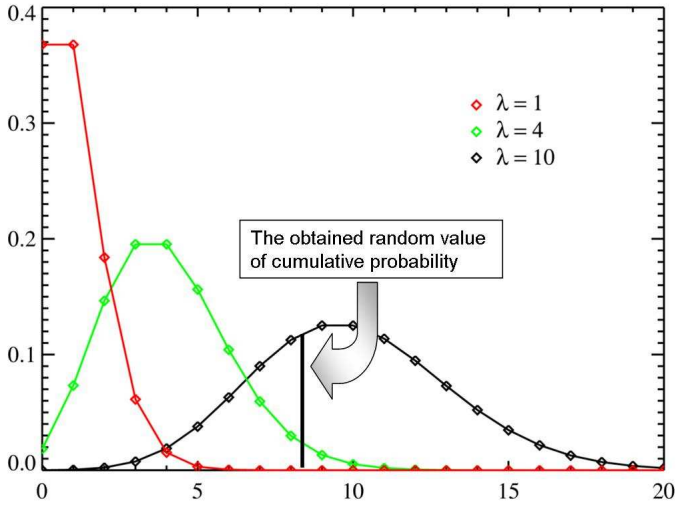

Figure 1: Computing the value of x_j

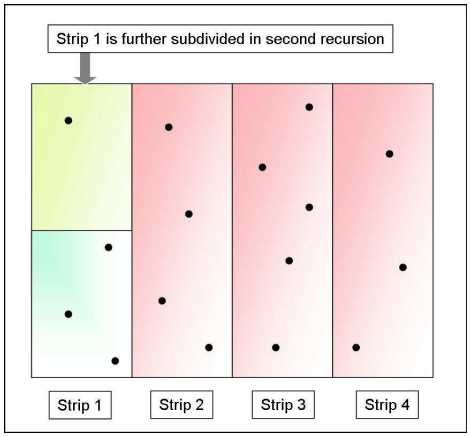

Figure 2: Random allotment of 16 sensors

4.4 Complexity Analysis of the Algorithm

In this subsection, we analyze the algorithm. Observe that, $X_j \sim P(\sqrt{(}\lambda))$, for the j^{th} partition, in any recursive step. So the expectation of each strip is given by

$$E(X_j) = \sqrt{\lambda}\ \forall j = 1 \text{ to } \sqrt{\lambda}, \text{ by Poisson Distribution}$$

Therefore,

$$\sum_{j=1}^{\sqrt{\lambda}} E(X_j) = \sqrt{\lambda}.\sqrt{\lambda} = \lambda \tag{3}$$

At each recursive depth this holds, implying the expected uniform distribution. For this reason, the area is partitioned into $\sqrt{\lambda}$ regions so that we can mimic the situation, where the distribution is uniform though random. Thus we simulate a fairly uniform placement (distributed over the whole area) of the sensor-nodes retaining the randomness. Next we present the complexity analysis of the algorithm.

In 1^{st} recursive-depth, the expected time-complexity = $O(\lambda^{\frac{1}{2}})$.
In 2^{nd} recursive-depth, the expected time-complexity = $O(\lambda^{\frac{1}{4}})$. (in each partition)
Similarly, in i^{th} recursive-depth, the expected time-complexity = $O(\lambda^{\frac{1}{2}^i})$. (in each partition)

Therefore, the total expected time-complexity is given by,

$$\prod_{i=1}^{\infty} O(\lambda^{\frac{1}{2^i}}) = O(\prod_{i=1}^{\infty} \lambda^{\frac{1}{2^i}}) \quad (4)$$

$$= O(\lambda^{\frac{1}{2}}.\lambda^{\frac{1}{4}}.\lambda^{\frac{1}{8}}\ldots) \quad (5)$$

$$= O(\lambda) \text{ (Since } \lim_{k\to\infty} \sum_{i=1}^{k} \frac{1}{2^k} = \frac{\frac{1}{2}}{1-\frac{1}{2}} = 1) \quad (6)$$

5 Results

In the following section we present the obtained result. In this discussion we refer to our scheme as the *Stensor Placement* for differentiating from the Constant Diffusion and R-random Placement. The following figures depict the alive sensors, before and after the simulation, in the square field of 1000 metres.

Observe that the Constant Diffusion and Stensor Placement achieve nearly equal sparse distribution of the sensors in the field. On the other hand, in the R-random Placement, the sensors are mainly concentrated in the center of the field. Interestingly, both the Constant Diffusion and R-random Placement fail to cover the whole region. Both of them have covered a circular space surrounding the base-station. But the Stensor Placement is able to cover almost the whole of the rectangular field due to its partition-based distribution strategy.

Also, note that after the simulation is over, both the Constant Diffusion and the R-random placement have a handful of sensors which retain power to function further. In fact observe that those handful of sensors are active only in the periphery of the square field. The sensors near the base-station are exhausted of the power quickly in all the cases because of the packet-traffic. This particular pattern of power exhaustion effectively reduces the number of active *carrier* sensors in the field. As the sensors lying around the base-station are more likely to run quickly out of energy, the packets fail to reach the base-station even when other sensors are *alive* elsewhere. Hence, from the observed result it makes sense to distribute the sensors evenly throughout the field. However, note that, the Stensor Placement has a number of *alive sensors* near the base-station.

5.1 Target Generation

We experimented with the placement of total 250 sensors. From the algorithm 4.1 (*PlaceSensor*), it is evident that some of the sensors may fall outside the specified region. Moreover, in case of R-random placement, the proportion of sensors falling outside the region to those falling inside is lower than that in Stensor placement. We have assumed 1000 targets whose arrival times follow an exponential distribution with the mean as 72 minutes ($\frac{1}{\lambda} = 72$, hence, $\lambda = 0.139$). The cumulative p.d.f of the distribution is given by,

$$\Pr[T \leq t] = 1 - e^{-\lambda t},$$

where, T is the random variable denoting the arrival time of a sensor. Fig 6 shows a particular example of the arrival times of the targets versus their locations. The X & Y-axes locate the

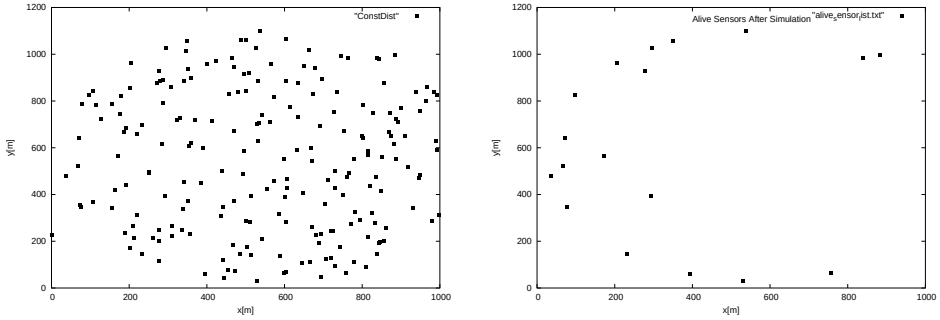

Figure 3: Constant Diffusion before and after simulation

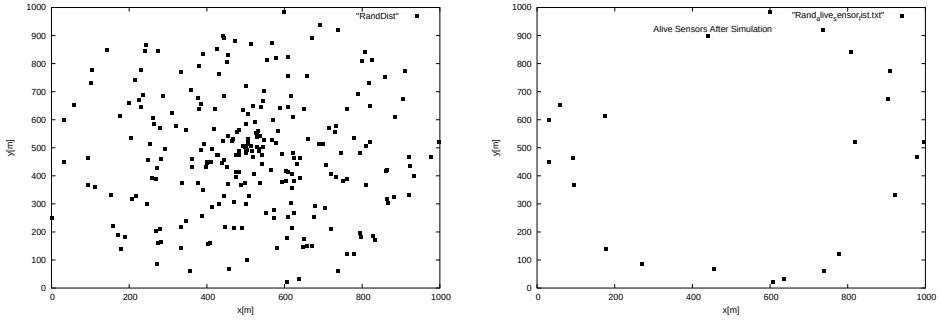

Figure 4: R-random Placement before and after simulation

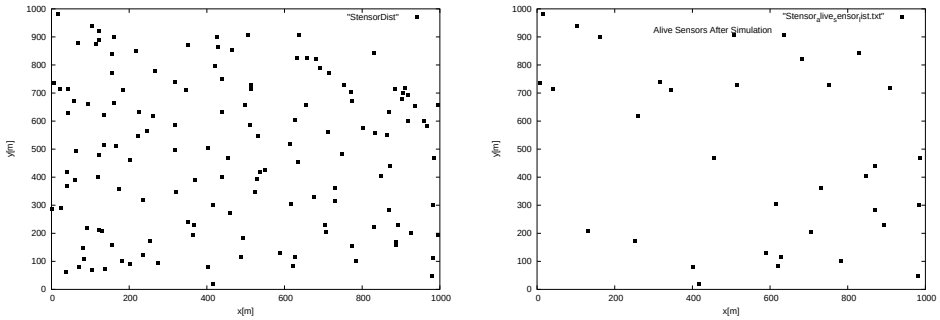

Figure 5: Stensor Placement before and after simulation

position of the target and the Z-axis represents the time in minutes. The position in the field is randomly chosen.

5.2 Assumptions

In this experiment, we have made certain assumptions. The assumptions are the following.

- *Buffer Limits:* It is a realistic idea to set an upper limit to the number of packets each sensor can recieve and transmit in unit time. We have assumed the maximum number of packets transmitted and recieved per simulation unit time, as 200 and 400 respectively.

$$\texttt{max\_trans\_per\_sim\_cycle} = 200 \tag{7}$$

$$\texttt{max\_recvd\_per\_sim\_cycle} = 400 \tag{8}$$

- *Activity Radii:* The sensors are able to sense a target within a given range of distance. Besides, for transmitting a given packet the reciever should be within a certain radius. We

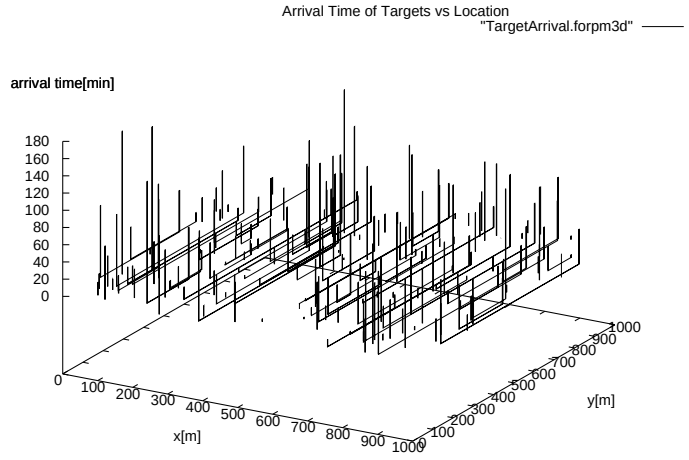

Figure 6: Target Arrival vs their Location

have assumed, the sensing radius and transmitting radius as 60m and 100m respectively.

$$\texttt{sense\_radius} = 60m \tag{9}$$

$$\texttt{trans\_radius} = 100m \tag{10}$$

- *Energy Consumption Rates:* Energy is spent in sensing and broadcasting the packets (we have assumed that the information regarding the packets are broadcasted within the `trans_radius`). We have assumed the following.

$$\texttt{sense\_consumption} = 1.9\mu\text{-J} \tag{11}$$

$$\texttt{trans\_consumption} = 3.3\mu\text{-J} \tag{12}$$

- *Intelligence:* The modern sensors are intelligent.
  - We have assumed that any sensor, discards a packet which has outlived a certain amount of time. In our case, this time is estimated by the number of hops. If the number of hops exceeds the number of sensors, the sensor currently possesing the packet discards it.
  - Our routing strategy is *flooding*. A sensor never returns a packet to its last *host* (here *host* means sensor hosting the packet).
  - If a particular sensor (say s_i) already posseses a packet(p) which has originated from target (say t_j), it does not accept any other packet originating from t_j, unless the packet p is transmitted to its neighbors.

5.3 Performance Analysis

Here we analyze the comparative performance of all the placement strategies. We have performed experiments in two separate approaches.

In the first case, the maximum energy content in each sensor is low. (`max_energy` = 2 mJ for each sensor). The total simulation time is 100 minutes. The Set 1 results in Table 1 show

	Constant Diffusion			R-random Placement			Stensor Placement		
	% Sensed	% Power Left	% Alive	% Sensed	% Power Left	% Alive	% Sensed	% Power Left	% Alive
Set1	14.8	2.08	11.11	13.6	2.02	9.2	14.9	5.78	24.11
	15.2	1.92	10.88	12.2	1.9	8.7	16.3	4.9	22.98
Set2	79.12	0.11	10.8	71.2	1.64	14.00	60.8	19.41	51.06
	76.8	0.2	8.8	69.8	0.98	12.08	56.5	18.6	48.3

Table 1: The Comparative Performance of Various Placement Strategies

the comparative performances of each of the placement algorithms. It is easy to percieve that the R-random Placement strategy is not that good either in terms of %-age of sensors sensed or the %-age of sensors *alive*. The Stensor Placement and the Constant Diffusion strategies faired well because of the goodness of their placement of sensors. With less amount of energy, the *core* group of sensors in R-random get exhausted and thereby obstructing the path of other *alive* sensors to reach the base-station.

In the second case, we allow much higher energy content per sensor (`max_energy` = 100 mJ for each sensor). The simulation time is also increased to 150 minutes. Here, the R-random network performed better than the other placement strategies. Set 2 in Table 1 displays the comparative results. The rationale behind this is the following. Each sensor has large energy content. Thus the *core* sensors in R-random strategy are *alive* for greater extent of time. Multiple active sensors in the *core* implies greater number of routes to the base-station. This results in enhanced efficiency of the network in handling greater traffic of packets. While the other approaches (with lesser number of *core* members) fail to provide those multiple routes and hence clogging of packets takes place, making the network slow for packet transmission. The end-result of all the algorithms run in Set 2 are shown in Fig 7. Certainly, Stensor Placement has a lesser loss rate of sensors compared to other methodologies. *However, placing larger number of sensors with high energy content (in Set 2, it is 50X than in Set 1) largely increases the cost, which may not be affordable in many practical scenarios.* This observation renders that our algorithm is also *cost efficient* and *practical*.

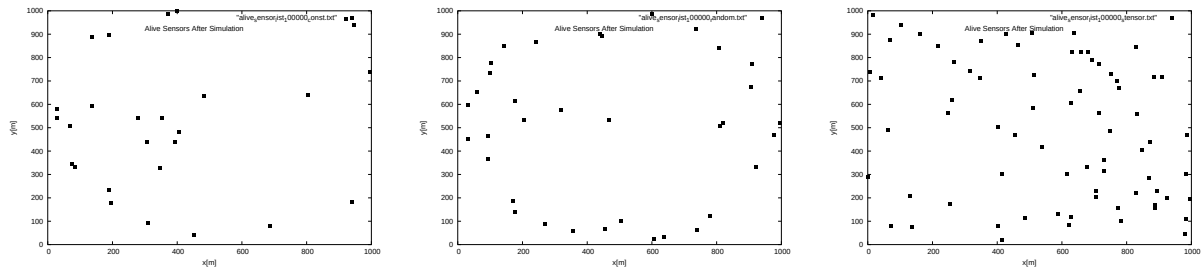

Figure 7: Alive sensors, left: Constant Diffusion, middle: R-random Placement, right: Stensor Placement

6 Conclusion

In this paper we have come up with a novel randomized placement algorithm for placement of sensors in a rectangular field. The location of the target-points which are to be sensed by the sensors are unknown apriori. Moreover, their appearance times vary following an exponential distribution. We have done a comparative study of the performances between the existing algorithms (e.g. Constant Diffusion and R-random Placement) and our proposed placement scheme. From those results we coclude that, the new placement algorithm works well when the maximum energy content is practical. Besides we also observed that performance of Stensor Algorithm is not up to the mark when sufficient energy is allowed to the sensor-points. However, it is not realistic to provide the sensors with surplus energy, since providing too much energy to the sensor-nodes will shoot up the cost of monitoring. We briefly summarize the merits and the limitations of this new placement algorithm.

- **Merits:**
  1. The placement strategy yields better results when energy content of the sensors are limited which is practical.
  2. The loss rate for sensors is less since they are evenly distributed. The low loss rate makes it suitable for easy replenishment of the sensors, if required. On the other hand R-random will necessitate fresh supply of sensors since its core contains a large share of the sensors, which unfortunately runs out of power soon.
  3. This placement strategy attempts to cover the whole region. This is essential since we have shown that targets can be generated anywhere at any point of time.
- **Demerits:**
  1. The core is not likely to be densely populated enhancing the chances of *clogging* of packets after sufficient simulation time.

References

[1] State Climate Office of North Carolina http://www.nc-climate.ncsu.edu/climate/hurricane.php

[2] Hurricane Rita Impact Studies, http://coastal.er.usgs.gov/hurricanes/rita/

[3] "*The Network Weather Service: A Distributed Resource Performance Forecasting Servce for Meta-computing*", R Wolski, N Spring, J Hayes - Journal of Future Generation Computing Systems, 1999

[4] "*A Survey of Sensor Network Applications*", Ning Xu

[5] G.Hoblos, M.Staroswiecki, and A.Aitouche. "*Optimal Design of Fault Tolerant Sensor Networks*", In Proc. IEEE International Conference on Control Applications, pp. 462-472

[6] Mika Ishizuka and Masaki Aida, "*Performance study of node placement in sensor networks*", ICD-CSW'04

[7] M. Bhardwaj, T. Garnett, and P. Chandrakasan. "*Upper bounds on the lifetime of sensor networks*", In Proc. IEEE ICC'01. pp785-790, June 2001

[8] G. Hoblos, M. Staroswiecki and A. Aitouche. "*Optimal design of fault tolerant sensor networks*". In Proc. IEEE International Conference on Control Applications, pp462-472, September 2000.