Dead Node Detection using Particle Swarm Optimization

Kiran[1], Dr. Pankaj Gupta[2], Mr. Deepak Goyal[3]
(Student-M.Tech, CSE Dept.)[1], (Proff. & HOD, CSE Dept.)[2], (Associate Proff. ,CSE Dept.)[3]
Vaish College Of Engineering[1,2,3]
Rohtak , Haryana, India.
Krnjaglan38@gmail.com[1], pankajgupta.vce@gmail.com [2], deepakgoyal.vce@gmail.com [3]

***Abstract*---In computer science, particle swarm optimization (PSO) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. In PSO, simple software agents, called *particles*, move in the search space of an optimization problem. This paper deals with the dead node detection and finding an optimized path without dead nodes.**

***Keywords*- Swarm, Dead nodes, PSO, network, Energy parameter, Distance, Iteration, Inertial weight factor, PSO algorithm.**

1. INTRODUCTION

Particle swarm optimization (PSO) is a population-based stochastic approach for solving continuous and discrete optimization problems. PSO is a robust stochastic optimization technique based on the movement and intelligence of swarms. PSO applies the concept of social interaction to problem solving. It was developed in 1995 by James Kennedy (social-psychologist) and Russell Eberhart (electrical engineer). It uses a number of agents (particles) that constitute a swarm moving around in the search space looking for the best solution. Each particle is treated as a point in a N-dimensional space which adjusts its "flying" according to its own flying experience as well as the flying experience of other particles. It Combines self-experiences with social experiences.

2. BASICS OF PSO

PSO is a Collection of flying particles (swarm) where each particle keeps track of its best solution, personal best, *pbest* and the best value of any particle, global best, *gbest*. Each particle adjusts its travelling speed dynamically corresponding to the flying experiences of itself and its colleagues. In brief,

- Each particle keeps track of its coordinates in the solution space which are associated with the best solution (fitness) that has achieved so far by that particle. This value is called personal best, ***pbest***.
- Another best value that is tracked by the PSO is the best value obtained so far by any particle in the

neighborhood of that particle. This value is called ***gbest***.

- The basic concept of PSO lies in accelerating each particle toward its pbest and the gbest locations, with a random weighted accelaration at each time step.

2.1 Comments on the inertial weight factor:

A large inertia weight (*w*) facilitates a global search while a small inertia weight facilitates a local search.By linearly decreasing the inertia weight from a relatively large value to a small value through the course of the PSO run gives the best PSO performance compared with fixed inertia weight settings.

Larger w means greater global search ability and smaller w shows greater local search ability.

***w*=wMax-[(wMax-wMin)xiter]/maxIter**(2)

where,

- wMax= initial weight,
- wMin = final weight,
- maxIter = maximum iteration number,
- iter = current iteration number.

3. PSO ALGORITHM

A basic variant of the PSO algorithm works by having a population (called a swarm) of candidate solutions (called particles). These particles are moved around in the search-space according to a few simple formulae. The movements of the particles are guided by their own best known position in the search-space as well as the entire swarm's best known position. When improved positions are being discovered these will then come to guide the movements of the swarm. The process is repeated and by doing so it is hoped, but not guaranteed, that a satisfactory solution will eventually be discovered.

Formally, let $f: \mathbb{R}^n \to \mathbb{R}$ be the cost function which must be minimized. The function takes a candidate solution as argument in the form of a vector of real numbers and produces a real number as output which indicates the objective function value of the given candidate solution. The gradient of f is not known. The goal is to find a solution a for which $f(a) \leq f(b)$ for all b in the search-space, which would mean a is the global minimum. Maximization can be performed by considering the function $h = -f$ instead.

Let S be the number of particles in the swarm, each having a position $x_i \in \mathbb{R}^n$ in the search-space and a velocity $v_i \in \mathbb{R}^n$. Let p_i be the best known position of particle i and let g be the best known position of the entire swarm. A basic PSO algorithm is then:

- For each particle $i = 1, ..., S$ do:
  - Initialize the particle's position with a uniformly distributed random vector: $x_i \sim U(b_{lo}, b_{up})$, where b_{lo} and b_{up} are the lower and upper boundaries of the search-space.
  - Initialize the particle's best known position to its initial position: $p_i \leftarrow x_i$
  - If ($f(p_i) < f(g)$) update the swarm's best known position: $g \leftarrow p_i$
  - Initialize the particle's velocity: $v_i \sim U(-|b_{up}-b_{lo}|, |b_{up}-b_{lo}|)$
- Until a termination criterion is met (e.g. number of iterations performed, or a solution with adequate objective function value is found), repeat:
  - For each particle $i = 1, ..., S$ do:

- For each dimension $d = 1, ..., n$ do:
  - Pick random numbers: r_p, $r_g \sim U(0,1)$
  - Update the particle's velocity: $v_{i,d} \leftarrow \omega\ v_{i,d} + \varphi_p\ r_p\ (p_{i,d}-x_{i,d}) + \varphi_g\ r_g\ (g_d-x_{i,d})$
- Update the particle's position: $x_i \leftarrow x_i + v_i$
- If ($f(x_i) < f(p_i)$) do:
  - Update the particle's best known position: $p_i \leftarrow x_i$
  - If ($f(p_i) < f(g)$) update the swarm's best known position: $g \leftarrow p_i$

- Now g holds the best found solution.

The parameters ω, φ_p, and φ_g are selected by the practitioner and control the behavior and efficacy of the PSO method.

4. DEAD NODE DETECTION

A Wireless Sensor network is a dynamic network with large no. of nodes. As the traffic increases over the network such type of network suffers from the problems like congestion and packet loss. Over the period these sensors loss the energy and can be dead. If the network is wireless in such case there are more chances of inclusion of some external node in the network. In such case there are more chances of Intrusion or packet loss over the network. The packet loss is acceptable up to some threshold value but as there is more packet loss we need some solution for this. The same solution is presented in this proposed work. The complete solution is defined in terms of two stages. In first stage we have to find the node that is responsible for packet loss over the network. Another stage is the development of algorithm or approach that will eliminate the node dynamically and get the reliable data transmission over the network. As we are working on broken link problem in a peer to peer network. Because of this we have to first establish a sensor network. For implementing this work we have used MATLAB editor tool. The result will be shown in command window and in figures.

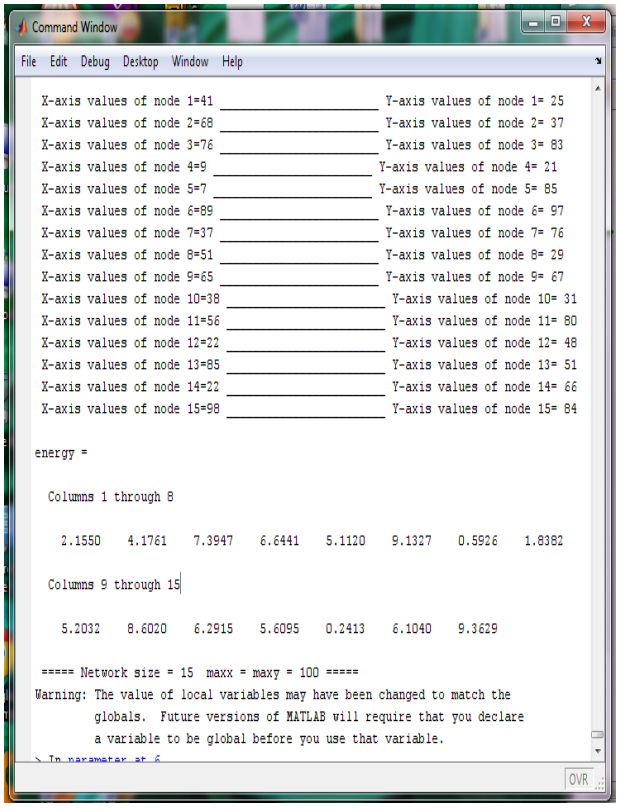

Figure 4.1 Network establishment

We had taken 15 nodes. The above figure shows their X and Y axis values assigned to them. All of these nodes have some energy values which also shown in the figure. Based on these parameter values, dead node detection will be done.

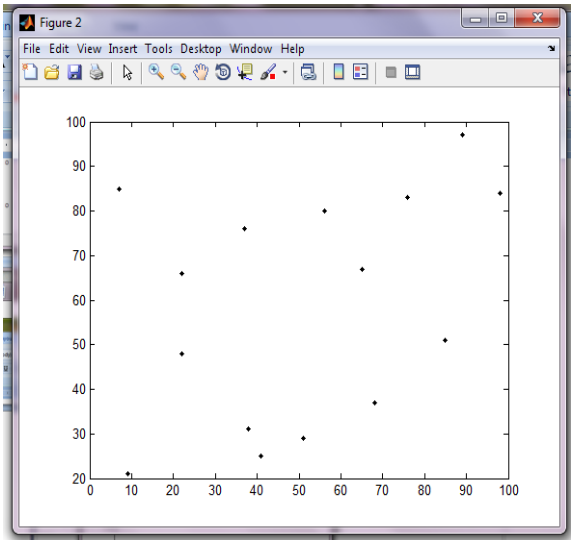

Figure 4.2 Network established having 15 nodes

The above figure has shown the network established which consists of 15 nodes. Now we apply the dead node detection algorithm on this network. According to it, a number of iteration are performed. Here the number of iteration means total number of communication performed. It includes the number of time data transmission is performed. We can take any value for it. Here we had taken its value 100 means hundred times communication will be performed. With each transmission some energy gets lost associated with each node. As the energy of a node get 0, the node is set as dead node.

Figure 4.3 Network with dead nodes

In the above figure, the red dots denote the dead nodes which are also shown with green color cloud callout. Here two nodes are shown as dead nodes. It means after the communications performed, the energy of these two nodes got zero. So these are declared as the dead nodes.

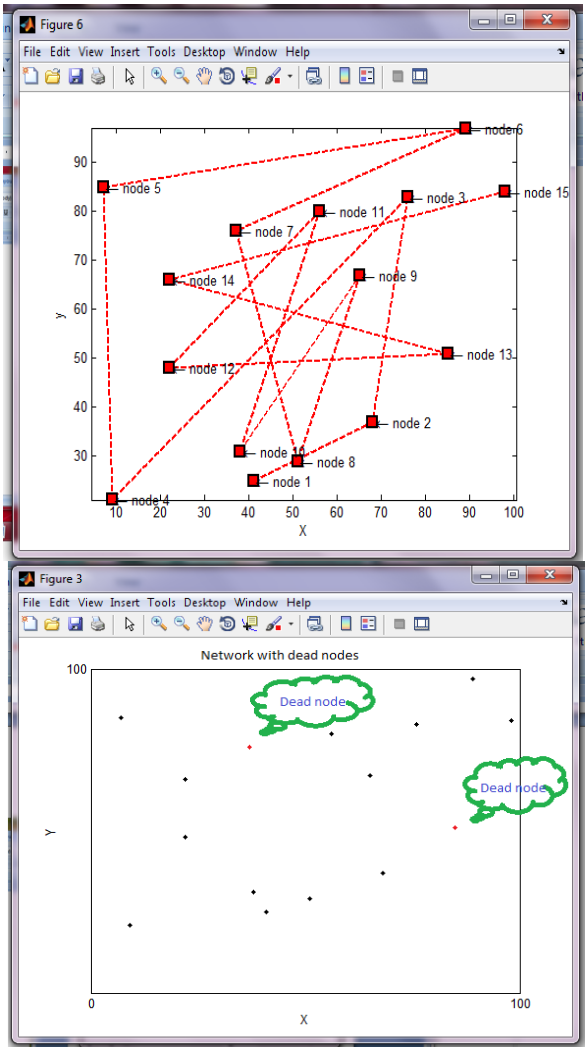

Figure 4.5 Communication path between 15 nodes

The above figure shows the communication path between the 15 nodes in the network, when there no optimization is performed. It includes both the live and dead nodes. All of fifteen nodes are shown separately with naming given as node1, node2…and so on. In this communication path, maximum distance is covered while traversing the nodes. As dead nodes are also included so whenever this path will be followed for communication, it will consume a lot of time and energy. Whenever we reach to dead node, the communication will be stopped. There will be no use of finding this kind of communication path including dead nodes.

5. ANALYSIS OF IMPLEMENTED WORK

We will do analyses of our work with respect to the distance covered with and without dead nodes. Both of these cases will be understand with help of graph shown below.

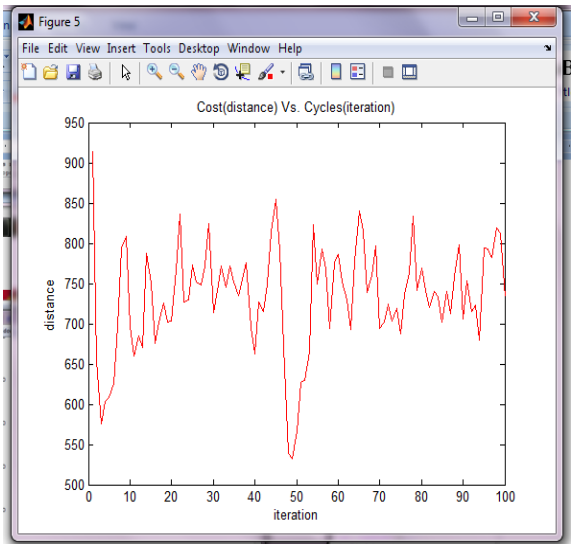

Figure 5.1 With dead nodes

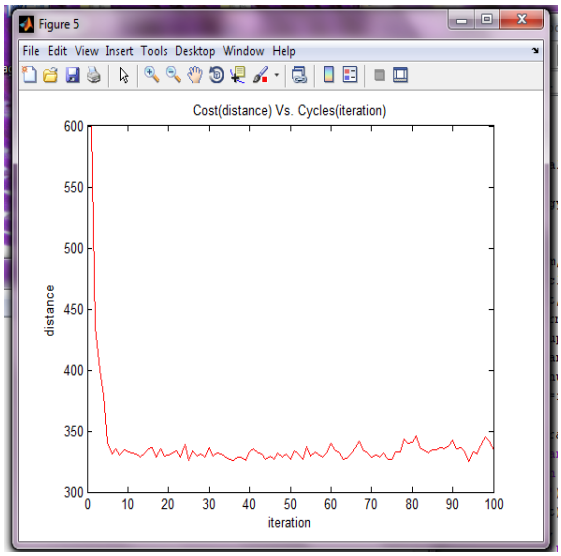

Figure 5.2 Without dead nodes

Figure 5.1 and 5.2 show the graph between distance covered during each communication is performed. The X axis shows the no. of cycles or iterations or no. of communication is performed. While the Y axis shows the distance covered with respect to each iteration. Figure 5.1 stands for network with dead and live both nodes. As shown in it, distance covered is large. Because no optimization is performed, due to it there will be only wastage of time and energy. Figure 5.2 stands for the network with dead node detection and after discarding these dead nodes. It clearly shows that the distance covered is much less than the previous figure. Optimization is performed here. The cost factor used for optimization is distance covered. In this way swarm optimization is performed for dead node detection and then discarding of these dead nodes.

6. FUTURE WORK

In this work, we had generated values of X and Y axis for nodes randomly. Also the energy given to each node is randomly generated with each new running task of the work. In this work we can also change the no. of nodes taken and no. of iteration performed. For the future work, we can change the optimization factor and then perform this optimization again so that with the distance covered in communication path, also the time taken for the optimized path also become optimum.

References

[1] Kennedy, J. "The particle swarm: Social adaptation of knowledge", Proceedings of the 1997 International Conference on Evolutionary Computation, IEEE Service Center, Piscataway, NJ. Pp.303-308(1997).

[2] Brandstatter, B. and Baumgartner, U., "Particle swarm optimization-mass-spring system analogon," *IEEE Transactions on Magnetics*, vol. 38, no. 2, pp. 997-1000, (Mar.2002).

[3] El-Gallad, El-Hawary, Sallam, and Kalas, "A Particle swarm optimizer for constrained economic dispatch with prohibited operating zones", *Canadian Conference on Electrical and Computer Engineering, 2002*, pp. 78-81(2002).

[4] Hu, X. and Eberhart, R. C., "Multiobjective optimization using dynamic neighborhood particle swarm optimization",. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2002), Honolulu, Hawaii USA*. pp. 1677-1681, (2002)

[5] Cedeno, W. and Agrafiotis, D. K., "Using particle swarms for the development of QSAR models based on K-nearest neighbor and kernel regression," *Journal of Computer-Aided Molecular Design*, vol. 17, no. 2-4, pp. 255-263, (Feb.2003).

[6] Mostaghim, S. and Teich, J. r. "Strategies for finding local guides in multi-objective particle swarm optimization (MOPSO)", *Proceedings of the IEEE Swarm Intelligence Symposium 2003 (SIS 2003), Indianapolis, Indiana, USA*. pp. 26-33, (2003).

[7] Zheng, Y., Ma, L., Zhang, L., and Qian, J. "Empirical study of particle swarm optimizer with an increasing inertia weight", *Proceedings of IEEE Congress on Evolutionary Computation 2003 (CEC 2003), Canbella, Australia.* pp. 221-226, (2003).

[8] Zhang, X., Yu, L., Zheng, Y., Shen, Y., Zhou, G., Chen, L., Xi, L., Yuan, T., Zhang, J., and Yang, B., "Two-stage adaptive PMD compensation in a 10 Gbit/s optical communication system using particle swarm optimization algorithm," *Optics Communications*, vol. 231, no. 1-6, pp. 233-242, (Jan.2004).

[9] X. Wang, S. Wang, and J. J. Ma, "An improved co-evolutionary particle swarm optimization for wireless sensor networks with dynamic deployment," *Sensors*, vol. 7, pp. 354–370, (2007).

[10] T. P. Hong and G. N. Shiu, "Allocating multiple base stations under general power consumption by the particle swarm optimization," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS)*, pp. 23–28,(2007).

[11] A. Boukerche, H. A. B. Oliveira, E. F. Nakamura, and A. A. F. Loureiro, "Localization systems for wireless sensor networks," *IEEE Wireless Commun. Mag.*, vol. 14, no. 6, pp. 6–12, (December 2007).