IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 1, January 2017

Review on Compute Analysis of Routing Algorithm for Wireless Sensor Networks

Prof. S.M. Patil[1], Sheetal Ragit[2]

Assistant Professor, Electronic and Telecommunication, Government College of Engineering, Jalgaon, India [1]

Student, Electronic and Telecommunication, Government College of Engineering, Jalgaon, India [2]

Abstract: Recent significant research on wireless sensor networks (WSNs) has led to the widespread adoption of software defined wireless sensor networks (SDWSNs), which can be reconfigured even after deployment. In this paper, we propose an energy-efficient routing algorithm for SDWSNs. In this algorithm, to make the network to be functional, control nodes are selected to assign different tasks dynamically. The selection of control nodes is formulated as an NP-hard problem, taking into consideration of the residual energy of the nodes and the transmission distance. To tackle the NP-hard problem, an efficient particle swarm optimization (PSO) algorithm is proposed. Simulation results show that the proposed algorithm performs well over other comparative algorithms under various scenarios.

Keywords: SDWSNs, sensing tasks, control nodes, residual energy, transmission distance, PSO.

I. INTRODUCTION

The emergence of big data and cloud technology has driven a fast development of wireless sensor networks (WSNs). A sensor node is normally comprised of one or more sensor units, a power supply unit, a data processing unit, data storage, and a data transmission unit [1]. A wireless sensor network is a collection of wireless nodes with limited energy that may be mobile or stationary and are located randomly in a dynamically changing environment. Wireless sensor networks hold the promise of revolutionizing the way we observe and interact with the physical world in a wide range of application domains such as environmental sensing, habitat monitoring and tracking, military defense, etc. The characteristics of low-cost, low-power, and multifunctional sensor have attracted a great deal of research attention, in that sensor nodes can perform intelligent cooperative tasks under stringent constrains in terms of energy and computational resources.

However, most previous research work only considers the scenario where a WSN is dedicated to a single sensing task, and such application-specific WSN is prone to high deployment costs, low service reutilization and difficult hardware recycling [2]. A software-defined wireless sensor network (SDWSN) consists of software-defined sensor nodes that can dynamically reconfigure their functionalities and properties. by loading different programs on-demand according to real time sensing requests. SDWSNs are emerging as a compelling solution to tackle the above issues. A software-defined sensor node equipped with several different types of sensors is able to undertake a variety of sensing tasks according to deployed and activated programs. In recent years, especially due to the advent of forth coming 5G networks, a number of prototypes have been practically implemented. SDWSNs enable programmable control in network and virtualization of network equipment by decoupling the control plane and data plane [2]. In SDWSNs, control intelligence is taken out from data plane devices and implemented in a logically centralized controller (network operating system, however can be formed by distributed clusters), which interacts with data plane devices through standard interfaces. Network operators run software programs on the controller to automatically manage data plan devices and optimize network resource usage [1]. This architecture enables up-to-date control schemes to be developed and deployed so as to enable new smart sensing services, making simplified network management in WSNs, which makes the future of SDWSNs bright [3]. However, to realize the aforementioned advantages of SDWSNs is not without challenges. In a sensor network, each node acts as both a sensor and router, with limited computing and communications capabilities, and storage capacity.

However, in many WSN applications, the deployment of sensor nodes is performed in harsh environments, which makes sensor replacement difficult and expensive [4]. Thus, in many scenarios, wireless nodes must operate without battery replacement for a long period of time. Consequently, the energy constraint is vital for the design of WSNs and SDWSNs. In an SDWSN, although different virtual networks can work together on top of the same physical infrastructure, the centralized control plane may lead to high energy costs due to information collection to reach a global view, and multiple virtual networks may compete for common physical network resources. Therefore, resource utilization of the SDWSN also needs to be carefully designed. In this paper, we consider the SDWSN as illustrated in Fig. 1 which consists of a sensor control server and a set of software-defined sensor nodes. The large scales of deployed nodes that are equipped with multi-functions are able to execute multi-tasks simultaneously. For example, a software defined node can

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 1, January 2017

monitor the temperature and humidity at the same time. The sensor control server can reprogram some sensor nodes by distributing a corresponding program to them for the tasks. We divide the sensor nodes into clusters, each of which consists of a control node and a number of common nodes with different tasks, aiming at balancing the energy consumption of sensor nodes and avoiding the collision of data transmission. Like TinyOS [1], every SDWSN OS can support multi-tasking that executes independently and non preemptively.

Traditional routing protocols in WSNs consume more energy for multi-tasking sensor networks because of the inflexibility. Therefore, based on the above architecture, we propose a new energy-efficient routing algorithm for software defined wireless sensor networks. Traditional routing protocols in WSNs consume more energy for multi-tasking sensor networks because of the inflexibility.

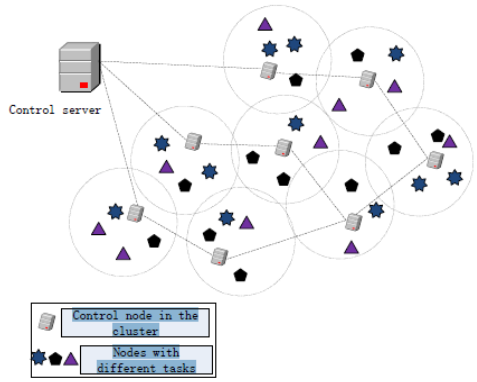

Fig. 1 An example of the software-defined sensor network with multi-tasks

Traditional routing protocols in WSNs consume more energy for multi-tasking sensor networks because of the inflexibility. Therefore, based on the above architecture, we propose a new energy-efficient routing algorithm for software defined wireless sensor networks. The control server selects the control nodes of each cluster, and the control nodes instruct the intra-cluster nodes to complete different tasks. In this project, we are motivated to investigate how to minimize the energy consumption if reprogramming by considering the control nodes' selection and multicasting routing[1]. Our main contributions are summarized as follows:

- We propose an energy-efficient routing algorithm for the multi-tasking SDWSNs. The selection of control nodes is formulated as an NP-hard problem, taking into account the residual energy of the nodes and the transmission distance; and
- To tackle the NP-hard problem, we propose an efficient particle swarm optimization (PSO) algorithm solve it.

II. OBJECTIVE

The design of Energy efficiency wireless sensor network is a challenging research since battery is consider as power source to the sensor nodes. Recharging battery is very difficult and impossible in some cases. Methods/ Analysis: In Wireless sensor networks, clustering techniques consist of partitioning the network into a various number of sensor groups called clusters. The clustering technique selects the cluster heads in the rotation manner to perform data aggregation operation.

Findings: To prolong the lifespan of a network, sensor nodes are scheduled to sleep dynamically. Sleep Scheduling (SS) mechanism is the most widely used technique for efficiently managing network energy consumption. In this paper, we provide a survey on energy-efficient scheduling mechanisms in Wireless sensor networks that has different network architecture than the traditional Wireless Sensor Networks.

III.LITERATURE SURVEY

A. RELATED WORK

1. Overview of Software-Defined Wireless Sensor Network

Overview of software-defined wireless sensor network the software-defined wireless sensor network represents a new paradigm shift that offers a significant promise to ubiquitous sensing and sensory data access through sensing as- a-service. Fig. 2 illustrates the logical view of the SDWSN.

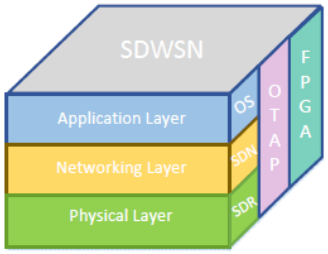

Fig. 2 A logical view of the SDWSN

There are a few pioneering investigations that have been studied in the literature. Lecointre et al. [1] propose a software-defined radio interface for wireless sensor networks. Rossi et al. [2] present a system dubbed SYNAPSE++ for over-the-air reprogramming of wireless sensor networks.

In [3], a die-hard sensor network is described, which can automatically monitor a disasterhit region by scattering many sensor nodes in the region. A TinyOS-based SDN framework that allows for multiple controllers within the WSN is presented in [4], which is hardware independent. Huang et al. propose a SDWSN prototype to improve the adaptability of WSNs for environmental monitoring applications, taking account of some constraints. The above existing works have demonstrated the feasibility of the SDWSN. Different from the above studies, this project focuses upon the energy efficiency in the network layer of the SDWSN, as shown in Fig. 2.1, as well as a particular emphasis on multi-task scheduling.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 1, January 2017

2. WSN Routing Algorithm

Routing is important in the WSN in determining the optimum routing paths of data packets, and there have been a great number of popular routing algorithms for the WSN. Ad hoc On-demand Distant Vector (AODV) [5] was proposed in 1999, and became an IETE standard. It is a routing algorithm in consideration of the distance between the nodes. Its quick adaption to link conditions, low memory usage and low network utilization make the ADOV algorithm popular.

However, the number of flooding messages increases significantly thanks to the increasing routing request messages. Clustering protocols can aid in data aggregation through efficient network organization. Low-energy adaptive clustering hierarchy (LEACH) [8] is one of the most well-known WSN hierarchical routing algorithms, which selects the cluster headers (CHs) based on a predetermined probability in order to rotate the CH role among the sensor nodes and to avoid fast depletion of the CH's energy. LEACH operates in two phases, i.e., the cluster setup phase and the steady phase. In the cluster setup phase, the cluster heads are selected and then broadcast to other nodes. In the steady state phase, actual transmission of data occurs.

However, the study of LEACH considers only energy consumption in receiving the advertisements from the CHs at each sensor node during the setup phase. The number of the cluster heads varies and the CHs do not have a good distribution. Furthermore, LEACH requires the transmission between the cluster heads and the sink to be completed in a single hop, which consumes a large quantity of energy and disrupts the energy balancing of nodes if the CHs are located far away from the sink. In [7], DF-LEACH is proposed as an improvement of LEACH, which takes into account the distance of the CH to the sink node, and thus saves communications energy. In [8], a hybrid energy-efficient distributed clustering approach (HEED) is proposed. The initial probability for each sensor to become a cluster head is dependent of its residual energy, and the performance results are fairly good. Hausdorff uses a greedy algorithm to select the cluster heads based on residual energy and location information, and this method can significantly prolong the network lifetime. In [9], an unequal cluster-based routing protocol is proposed, which focuses on load balancing in order to address the hot-spot issue. Mottola et al. [10] propose an adaptive energy-aware multi-sink routing algorithm, which is expressly designed for many-to-many communications. In [11], the authors address the issue of load balancing through considering different hop distances for the clusters. EDIT [13] is proposed to select the cluster head based on not only energy but also delay. The traditional routing algorithms are unable to adapt to the flexibility of SDWSNs. Consequently, we propose a new routing algorithm for the SDWSN, which can accommodate the SDWSN's conditions flexibly and helps achieve better results.

IV. SYSTEM DEVELOPMENT

A. NETWORK MODEL

In this paper, we consider the network architecture as shown in Fig. 1. G = (V;L) denotes the directed graph representing the network. V is the vertex set, including one control sever and a number of sensor nodes distributed within the monitoring field randomly. L is the set of directed links. The following assumptions on the sensor network and sensor nodes under consideration in this paper are made:

- We consider a set of _ sensing targets, e.g., temperature, humidity, and so on, which are randomly distributed within the same region of the SDWSN [11].
- The resources in a sensor node should be managed, controlled and allocated in an orderly manner in support of various sensing tasks. Besides, to complete different tasks, corresponding programs are stored on the sensor nodes [1], and the sensor node shall allow application programmers to adjust the sensor functionalities via invoking different programs.
- Each sensor node has the same ability to operate either in the sensing mode to perceive the environmental parameters or in the communications mode to send data among each other, or directly to the control server, and each node can gather data packets from a cluster member when acting as the control node. And each sensor node is assigned a unique identifier (ID).
- The sensor nodes and control server are stationary after deployment, which is typical for sensor network applications.
- Initial energy is fair to each sensor node, and the network is considered homogeneous;
- All the nodes are left unattended without battery replacement after deployment.
- Nodes are location-unaware, i.e., not equipped with GPS capable antennae or other similar equipment, and each node is assigned a number according to its location.
- The links between the nodes are symmetric. A node can estimate the distance to another node based only on the received signal power.
- The control server is externally powered.

B. LEACH PROTOCOL

LEACH Protocol is a typical representative of hierarchical routing protocols. It is self adaptive and self-organized. LEACH protocol uses round as unit, each round is made up of cluster set-up stage and steady-state stage[8], for the purpose of reducing unnecessary energy costs, the steady state stage must be much longer than the set-up stage. The process of it is shown in Figure 3.

At the stage of cluster forming, a node randomly picks a number between 0 to 1, compared this number to the threshold values **t(n)** , if the number is less than **t(n)** , the it become cluster head in this round, else it become common node.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 1, January 2017

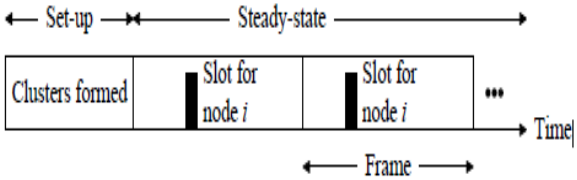

Fig.3. LEACH Protocol Process

Threshold **t(n)** is determined by the following:

$$t(n) = \begin{cases} \dfrac{p}{1 - p * (r \bmod \frac{1}{p})} & if\ n \in G \\ 0 & if\ n \notin G \end{cases}$$

Where p is the percentage of the cluster head no des in all nodes, r is the number of the round, G is the collections of the nodes that have not yet been head nodes in the first 1/P rounds. Using this threshold, al l nodes will be able to be head nodes after 1/P rounds. The analysis is as follows: Each node becomes a cluster head with probability p when the round begins, the nodes which have been head nodes in this round will not be head nodes in the next 1/P rounds, becausee the number of the nodes which is capable of head node will gradually reduce, so, for these remain nodes, the probability of being head nodes must be increased. After 1/P-1 round, all nodes which have not been head nodes will be selected as head nodes with probability 1, when 1/P rounds finished, all nodes will return to the same starting line. When clusters have formed, the nodes start to transmit the inspection data. Cluster heads receive data sent from the other nodes, the received data was sent to the gateway after fused. This is a frame data transmission. In order to reduce unnecessary energy cost, steady stage is composed of multiple frames and the steady stage is much longer than the set-up stage.

C. ROUTING ALGORITHM BASED ON PSO

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behaviour of bird flocking or fish schooling.PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. The detailed information will be given in following sections. Compared to GA, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust. PSO has been successfully applied in many areas: function optimization, artificial neural network training, fuzzy system control, and other areas where GA can be applied [12].

1. PSO Algorithm

As stated before, PSO simulates the behaviours of bird flocking. Suppose the following scenario: a group of birds are randomly searching food in an area. There is only one piece of food in the area being searched. All the birds do not know where the food is. But they know how far the food is in each iteration. So what's the best strategy to find the food? The effective one is to follow the bird which is nearest to the food. PSO learned from the scenario and used it to solve the optimization problems. In PSO, each single solution is a "bird" in the search space. We call it "particle". All of particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles[9]. The particles fly through the problem space by following the current optimum particles. PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In the every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called gbest[10]. When a particle takes part of the population as its topological neighbours, the best value is a local best and is called lbest. After finding the two best values, the particle updates its velocity and positions with following equation (a) and (b).

v[] = v[] + c1 * rand() * (pbest[] - present[]) + c2 * rand() *(gbest[]-present[])…..(a)

present[]=persent[]+v[]..... (b)

v[] is the particle velocity, persent[] is the current particle (solution). pbest[] and gbest[] are defined as stated before. rand () is a random number between (0,1). c1, c2 are learning factors. Usually c1 = c2 = 2[12]. The pseudo code of the procedure is as follows.

For each particle
  Initialize particle
END
Do
  For each particle
    Calculate fitness value
If the fitness value is better than the best fitness value (pBest) in history set current value as the new pBest
End
Choose the particle with the best fitness value of all the particles as the gBest For each particle
Calculate particle velocity according equation (a)
Update particle position according equation (b)
End
While maximum iterations or minimum error criteria is not attained [13].

Particles' velocities on each dimension are clamped to a maximum velocity Vmax. If the sum of accelerations

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 1, January 2017

would cause the velocity on that dimension to exceed Vmax, which is a parameter specified by the user. Then the velocity on that dimension is limited to Vmax.

Particle swarm optimization (PSO) [5] is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995, inspired by the social behavior of birds flocking or fish schooling. PSO begins with a group of random particles (random solutions), aiming at finding out the optimum solution through an iterative process.

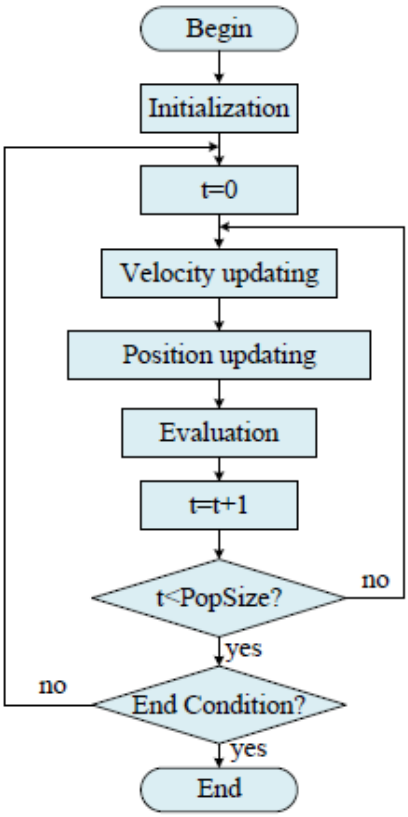

Fig. 4. Flowchart of the PSO Algorithms

Each particle has a fitness value, which will be evaluated by the fitness function to be optimized in each generation [1]. During the search process, each particle in the population consists of a d-dimensional vector including the velocity vector vi = [vi1; vi2; :::; vid], the current position vector (pBest) xi = [xi1; xi2; :::; xid], and the previous best position vector pi = [pi1; pi2; :::; pid], where d is the dimensionality of the search space. What's more, the whole population maintains a global best-so-far population vector pg = [Pg1; pg2; :::; pgd] [3]. The flowchart of PSO is shown in Fig. 4. As can be seen from the figure, during each iteration of the evolutionary process in PSO, each particle learns from its own search experience pBest and the swarm's search experience gBest to update its velocity vi and position xi [12, 13]. During the iterations, the velocity of the particle is updated according to the following

$$v_{id}(t+1) = wv_{id}(t)+c1(p_{id} -x_{id}(t))+c2\_(pgd -x_{id}(t))\ldots\ldots(5)$$

The position of the particle is updated as follows

$$x_{id}(t + 1) = x_{id}(t) + vij(t)\ldots..(6)$$

where the representation of v_{id} is similar to that of x_{id}; P_{id} and Pgd are the d^{th} dimension of the i^{th} particle's velocity. Coefficients _ and _ are two randomly generated values within the range of [0; 1] for the d^{th} dimension. c1 and c2 are two acceleration parameters which are commonly set to 2.0 or adaptively controlled according to the evolutionary states. Factor w is the inertial weight, which plays the role of controlling the impact of the previous velocity of a particle on the current one so as to balance between global search (large inertial weight) and local search (small inertial weight). However, PSO exhibits poor local search ability and often leads to premature convergence, especially in complex multipeak search problems. To tackle this issue, this paper proposes a method which adapts itself nonlinearly as follows

$$w = (w_{max}- w_{min} \square d_1)* e^{1/1+d_2*t/K} \ldots\ldots(7)$$

where w_{max} and w_{min} represent the maximum and minimum inertial weights and are always set to 0.9 and 0.4, respectively. K is the maximum number of allowed iterations while t represents the current iteration. d1 and d2 are two control factors which control the value of w between w_{min} and w_{max}. The execution of the algorithm is comprised of two phases, i.e., the control nodes' selection phase and the data transmission phase. The two phases are performed in each round of the network operation and repeated periodically. We elaborate on how to use the non-linear weight particle swarm optimization algorithm (NWPSO) to select the control nodes in the next section.

2. Program

```
clc;
clear all;
close all;
swarm_size = 64;
maxIter = 50;
inertia = 1.0;
correction_factor = 2.0;
a = 1:8;
[X,Y] = meshgrid(a,a);
C = cat(2,X',Y');
D = reshape(C,[],2);
swarm(1:swarm_size,1,1:2) = D
swarm(:,2,:) = 0;
swarm(:,4,1) = 1000;
plotObjFcn = 0;
objfcn =  @(x)(x(:,1) - 20).^2 + (x(:,2) - 25).^2;
for iter = 1:maxIter
   swarm(:, 1, 1) = swarm(:, 1, 1) + swarm(:, 2, 1)/1.3;
swarm(:, 1, 2) = swarm(:, 1, 2) + swarm(:, 2, 2)/1.3;
   x = swarm(:, 1, 1);
   y = swarm(:, 1, 2);
   fval = objfcn([x y]);
   for ii = 1:swarm_size
      if fval(ii,1) < swarm(ii,4,1)
```

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 1, January 2017

```
        swarm(ii, 3, 1) = swarm(ii, 1, 1);
swarm(ii, 3, 2) = swarm(ii, 1, 2);
swarm(ii, 4, 1) = fval(ii,1);
      end
   end
 [~, gbest] = min(swarm(:, 4, 1));
swarm(:, 2, 1) = inertia*(rand(swarm_size,1).*swarm(:, 2,
1)) + correction_factor*(rand(swarm_size,1).*(swarm(:, 3,
1) ... - swarm(:, 1, 1))) +correction_factor*
(rand(swarm_size,1).*(swarm(gbest, 3, 1) - swarm(:, 1,
1)));
   swarm(:, 2, 2) = inertia*(rand(swarm_size,1).*swarm(:,
2, 2)) + correction_factor*(rand(swarm_size,1).*(swarm(:,
3, 2) .. - swarm(:, 1, 2))) + correction_factor*
(rand(swarm_size,1).*(swarm(gbest, 3, 2) - swarm(:, 1,
2)));
clf;plot(swarm(:, 1, 1), swarm(:, 1, 2), 'bx');      axis([-2 40
-2 40]);
 pause(.1);
disp(['iteration: ' num2str(iter)]);
end
```

3. Simulation Result of PSO

Figure 5 and 6 shows the result of PSO algorithm with multiple and single partical.

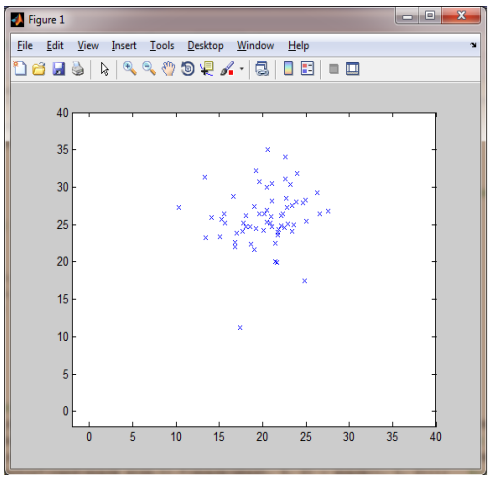

Fig.5 simulation result of PSO for swarm node

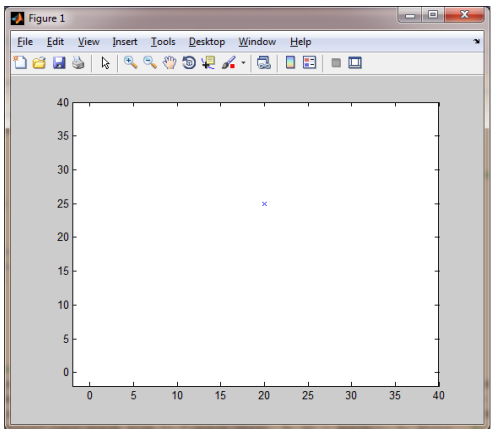

Fig. 6 Output of Single Node Based on Velocity and Position

REFERENCES

[1]. Wei Xiang, Senior Member, IEEE, Ning Wang, and Yuan Zhou, Member, IEEE "An Energy-efficient Routing Algorithm for Software-defined Wireless Sensor Networks" Citation information: DOI 10.1109/JSEN.2016.2585019, IEEE Sensors

[2]. N. Lavanya* and T. Shankar "A Review on Energy-Efficient Scheduling Mechanisms in Wireless Sensor Networks" Indian Journal of Science and Technology, Vol 9(32), DOI: 10.17485/ijst/2016/v9i32/86910, August 2016.

[3]. M. Chen, Y. Zhang, Y. Li, M. Hassan, and A. Alamri, "AIWAC: affective interaction through wearable computing and cloud technology," IEEE Wireless Communications, vol. 22, no. 1, pp. 20–27, Feb. 2015.

[4]. Poojarini Mitra,Sinthia Roy " A Reliable and Energy Efficient Enhancement of Data MULEs Protocol for Wireless Sensor Network" International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 5, May 2015.

[5]. Rajat Kandpal*, Rajesh Singh, H L Mandoria "Energy Efficient Routing in Random Deployment of Wireless Sensor Networks" International Journal of Emerging Research in Management &Technology ISSN: 2278-9359 (Volume-4, Issue-9) September 2015.

[6]. D. Zeng, P. Li, S. Guo, and T. Miyazaki, "Minimum-energy reprogramming with guaranteed quality-of-sensing in software-defined sensor networks," in 2014 IEEE International Conference on Communications (ICC), Sydney, Australia. IEEE, June 2014, pp. 288–293.

[7]. K.-B. Lee and J.-H. Kim, "Multiobjective particle swarm optimization with preference-based sort and its application to path following footstep optimization for humanoid robots," IEEE Transactions on Evolutionary Computation, vol. 17, no. 6, pp. 755–766, Dec. 2013.

[8]. Chunyao FU1, Zhifang JIANG1, Wei WEI and Ang WEI " An Energy Balanced Algorithm of LEACH Protocol in WSN" IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 1, No 1, January 2013.

[9]. Stefanos A. Nikolidakis 1, Dionisis Kandris 2, Dimitrios D. Vergados 1 and Christos Douligeris 1, "Energy Efficient Routing in Wireless Sensor Networks Through Balanced Clustering" algorithms ISSN 1999-4893 www.mdpi.com/journal/algorithms 2013.

[10]. A. De La Piedra, F. Benitez-Capistros, F. Dominguez, and A. Touhafi, "Wireless sensor networks for environmental research: A survey on limitations and challenges," in IEEE EUROCON, Zagreb. IEEE, July. 2013, pp. 267–274.

[11]. Stefanos A. Nikolidakis 1, Dionisis Kandris 2, Dimitrios D. Vergados 1 and Christos Douligeris 1, "Energy Efficient Routing in Wireless Sensor Networks Through Balanced Clustering" algorithms ISSN 1999-4893 www.mdpi.com/journal/algorithms 2013.

[12]. T. Luo, H.-P. Tan, and T. Q. Quek, "Sensor openflow: Enabling software-defined wireless sensor networks," IEEE Communications Letters, vol. 16, no. 11, pp. 1896–1899, Nov. 2012.

[13]. WANG Jin-wei, 2, SUN Hua-zhi, SUN De-bing, Research on the Number of Optimal Cluster Heads of Wireless Sensor Networks Based on Energy Consumption. Journal of Computer Research and Development, 2008, 03.