Region Directed Diffusion in Sensor Network Using Learning Automata: RDDLA

A. Sayyad[1], M. Shojafar[2], Z. Delkhah[1], and A. Ahmadi[1]
[1]*Msc. in Information Tech., Computer and Elec. Dept., Islamic Azad University Qazvin Branch, Iran*
[2]*Msc. in Computer Eng., Computer and Elec. Dept., Islamic Azad University Qazvin Branch, Iran*

Abstract

One of the main challenges in wireless sensor network is energy problem and life cycle of nodes in networks. Several methods can be used for increasing life cycle of nodes. One of these methods is load balancing in nodes while transmitting data from source to destination. Directed diffusion algorithm is one of declared methods in wireless sensor networks which is data-oriented algorithm. Directed diffusion deals with two fundamental problems. First, in the network the data packets traverse the invalid routes up to the central node in order to make new routes and eliminate the previous ones at a very short period of time. They are dispatched from a platform lacking any central node which itself causes the decrease of data delivery rates. Second, the reconstruction of such routes urges the application of exploration phase which comes along with the distribution of interest packets and exploratory data resulting into a great deal of outputs injected into the network. But with every motion, the application of exploratory phase of central node makes an overflow of outputs. Certainly with intense movement it enjoys high importance. Having more than one sink, network is separated to some regions and the proposed algorithm called Region Directed Diffusion Learning Automata (RDDLA) updates the rout between these sinks and finds interface node with learning automata and sends packet from source to this node and transmits data to sinks with this node. This approach decreased overall network metrics up to 22%.

Keywords: *Wireless sensor networks, ad- hoc, directed diffusion, Learning automata, Region Transmitting.*

1. Introduction

In the near future, advances in processor, memory and radio technology will enable small cheap nodes capable for wireless communication and significant computation. The addition of sensing capability to such devices will make distributed micro sensing—an activity in which a collection of nodes coordinate to achieve a larger sensing task—possible. Such technology can revolutionize information gathering and processing in many situations. Large scale, dynamically changing, and robust sensor networks can be deployed in inhospitable physical environments such as remote geographic regions or toxic urban locations. They will also enable low maintenance sensing in more benign, but less accessible, environments: large industrial plants, aircraft interiors etc [1].

Wireless sensor networks are a trend of the past few years, involved in deploying a large number of small nodes [2]. The nodes then sense environmental changes and report them to other nodes over flexible network architecture. Sensor nodes are great for deployment in hostile environments or over large geographical areas. One of the types of WSN is Ad Hoc [3]. An Ad Hoc wireless network, or simply an ad hoc network, consists of a collection of geographically distributed nodes that communicate with one other over a wireless medium [4].

Wireless ad hoc network are formed by devices that are able to communicate with each other using a wireless physical medium without having to resort to a pre-existing network infrastructure. These networks, also known as mobile ad hoc networks, can form stand-alone groups of wireless terminals, but some of these terminals could also be connected to a cellular system or to a fixed network. A fundamental characteristic of ad hoc networks is that they are able to configure them on-the-fly without the intervention of a centralized administration. Terminals in ad hoc network can function not only as end systems (executing application, sending information as source nodes and receiving data as destination nodes), but also as intermediate systems (forwarding packets from other nodes).

Therefore, it is possible that two nodes communicate even when they are outside of each other's transmission ranges because intermediate nodes can function as routers. This is why wireless ad hoc networks are also known as "multi-hop" wireless networks" [5]. The lack of a fixed infrastructure in ad hoc networks implies that any computation on the network needs to be carried out in a decentralized manner. Thus, many of the important problems in ad hoc networking can be formulated as problems in distributed computing. However, there are certain characteristics of ad hoc networks that make this study somewhat different than traditional works in distributed computing [6]. Basic problem domain that is Routing in Ad-hoc network is focused in this research. Some well-known algorithms exist that help rooting in ad hoc network implemented such as SPIN, LEACH, SPAN and Directed Diffusion. One of routing algorithms in wireless network is Ad hoc [7].

Directed Diffusion is data centric in which all communication is for named data. All nodes in a directed diffusion-based network are application-aware. This enables diffusion to achieve energy savings by selecting empirically good paths and by caching and processing data in-network (e.g., data aggregation) [8, 9].

In this Paper, Directed Diffusion is explained (one phase pull approach, two phase pull approach) in Section 2 as related works, then, the review of Learning Automata is provided in Section 3, in Section 4 the approach of Directed Diffusion is proposed. Likewise reconstruction of the whole route and maintenance of the routes in order to support the algorithm of directed diffusion are provided. Based on this Suggestion one phase called Covering Phase to Directed Diffusion is added to Support Sink Movement with the Help of Learning Automata. Section 5 provides comparison of our approach with others routing algorithm such as one phase pull approach and two phase pull approach. All related diagrams are also provided in this section 5. Finally, Conclusions of this research have been covered in section 6.

2. Related Works

The Directed Diffusion protocol includes a family of algorithms. They are two-phase pull diffusion, one-phase pull diffusion [10, 11].

2.1. Two-Phase Pull Diffusion

If a user in the network would like to track an object in some remote sub-region, The user subscription to that particular object information, specified by a set of attributes is needed. Sensors across the network publish that information. The user’s application subscribes to data using a list of attribute-value pairs that describe a task using some task-specific naming scheme. Intuitively, attributes describe the data that are desired by specifying sensor types and possibly some geographic regions. The user’s node becomes a sink, creating an interest of attributes specifying a particular kind of data. The interest is propagated from neighbor-to-neighbor towards sensor nodes in the specified region. A key feature of directed diffusion is that every sensor node can be task aware, that is these nodes can store and interpret interests, rather than simply forwarding them along. Each sensor node that receives an interest remembers which neighbor or neighbors sent it. To each such neighbor, it sets up a gradient. A gradient represents the direction towards which data matching the interest flows, and the status of that demand, whether it is active or inactive and possibly the desired update rate. After setting up a gradient, the sensor node redistributes the interest to its neighbors. When the node can infer where potential sources might be, the interest can be forwarded to a subset of neighbors. Otherwise, it will simply broadcast the interest to all of its neighbors.

Sensors indicate what data they may generate by publishing an appropriate set of attributes. Thus, sensors become potential sources. As interests travel across the network, sensors with matching publications are triggered and the application activates its local sensors to begin collecting data. Prior to activation the node’s sensors would be in a low-power mode. The sensor node then generates data messages matching the interest. In directed diffusion, data is also represented using an attribute-based naming scheme. Data is cached at intermediate nodes as it propagates toward sinks. Cached data are used for several purposes at different levels of diffusion. The core diffusion mechanism uses the cache to suppress duplicate messages and prevent loops, and it can be used to preferentially forward interests. Cached data are also used for application specific, in-network processing. That is, data from detections of a single object by different sensors may be merged to a single response based on sensor-specific criteria. The initial source data message is marked as exploratory and is sent to all neighbors for which it has matching gradients.

The initial flooding of the interest, together with the flooding of the exploratory data, constitutes the first phase of the two-phase pull diffusion. If the sink has multiple neighbors, it chooses to receive subsequent data messages for the same interest from a preferred neighbor, that is, the one which delivered the first copy of the data message. The sink reinforces the preferred neighbor, which, in turn reinforces its preferred upstream neighbor, and so on. The sink may also negatively reinforce its current preferred neighbor if another neighbor delivers better lower latency sensor data. This negative reinforcement propagates neighbor-to-neighbor, removing gradients and tearing down and existing path if it is no longer needed. Negative reinforcements suppress loops or duplicate paths that may arise due to changes in network topology.

After the initial exploratory data message, subsequent messages are sent only on reinforced paths. The path reinforcement, and the subsequent transmission of data along reinforced paths, constitutes the second phase of the two-phase pull diffusion. Periodically the source sends additional exploratory data messages to adjust gradients in the case of network changes, due to, node failure, energy depletion, or mobility, temporary network partitions, or to recover from lost exploratory messages. Recovery from data loss is left to the application. While simple applications with transient data, such as sensors that report their state periodically, need no additional recovery mechanism, retransmission scheme for applications that transfer large, persistent data objects is being developed.

2.2. One phase Pull Diffusion

One-phase pull is a subscriber-based system that avoids one of the two phases of flooding presented in the two-phase pull. The sink sends interest messages that disseminate through the network, establishing gradients. When an interest arrives at a source it does not mark its first data message as exploratory, but instead sends data only on the preferred gradient. The preferred gradient is determined by the neighbor who was the first to send the matching interest, thus suggesting the lowest latency path. Therefore, one-phase pull does not require reinforcement messages, and the lowest latency path is implicitly reinforced.

By comparison, two-phase pull uses several control messages to forward data, which has been reduced in one phase pull. In this case, the sink sends the interest messages and accordingly the data are sent on the preferred gradient. As follows, the approach in this research is proposed that it is more applicable than One-Phase Pull.

3. Learning Automata

Learning Automata are adaptive decision-making devices operating on unknown random environments. A Learning Automaton has a finite set of actions and each action has a certain probability (unknown to the automaton) of getting rewarded by the environment of the automaton. The aim is to learn to choose the optimal action (i.e. the action with the highest probability of being rewarded) through repeated interaction on the system. If the learning algorithm is chosen properly, then the iterative process of interacting on the environment can be made to result in selection of the optimal action. Fig. 1 illustrates how a stochastic automaton works in feedback connection with a random environment. Learning Automata can be classified into two main families: fixed structure learning automata and variable structure learning automata (VSLA). In the following, the variable structure learning automata which will be used in this situation is described [12].

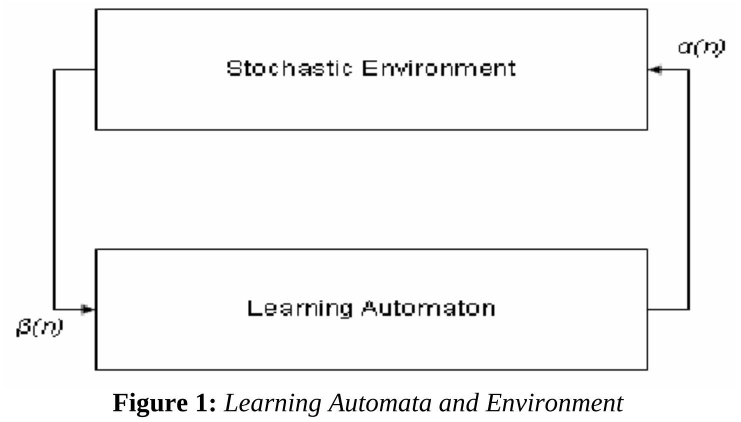


Figure 1: *Learning Automata and Environment*

A VSLA is a quintuple $(\alpha, \beta, p, T(\alpha, \beta, p))$ where a, β, p are an action set with s actions, an environment response set and the probability set p containing s probabilities, each being the probability of performing every action in the current internal automaton state, respectively. If the response of the environment takes binary values learning automata model is P-model and if it takes finite output set with more than two elements that take values in the interval [0, 1], such a model is referred to as Q-model, and when the output of the environment is a continuous variable in the interval [0, 1], it is refer to as S-model. The function of T is the reinforcement algorithm, which modifies the action probability vector p with respect to the performed action and received response. Assume β € [0, 1]. A general linear schema for updating action probabilities can be represented as follows. Let action i be performed then (1), (2) shows as follows:

$$p_j(n+1) = p_j(n) + \beta(n)[b/(r-1) - bp_j(n)] \\ -[1-\beta(n)]ap_j(n) \qquad \forall j \qquad j \neq i \tag{1}$$

$$p_i(n+1) = p_i(n) + \beta(n)[bp_i(n)] + \\ [1-\beta(n)]a[1-p_i(n)] \qquad \forall i \tag{2}$$

Where a and b are reward and penalty parameters. When a=b, the automaton is called LRP. If b=0, the automaton is called LRI and if 0<b<<a<1, the automaton is called LR ε P [13].

4. Region Directed Diffusion Learning Automata (DDLA)

At first, dividing the whole sensor network to some regions using bully algorithm [14] (Figure 2). In Bully algorithm, the nodes that are very close to each other are being in one region. Rejoining has two main purposes. First, overlapping nodes do not answer the request synchronically; because, this reduce nodes and network energy (prevent wasting energy). Second, coordinator of the region was used for increasing the speed of data transmitting (coordinator is a node that its energy is more than other nodes in region).

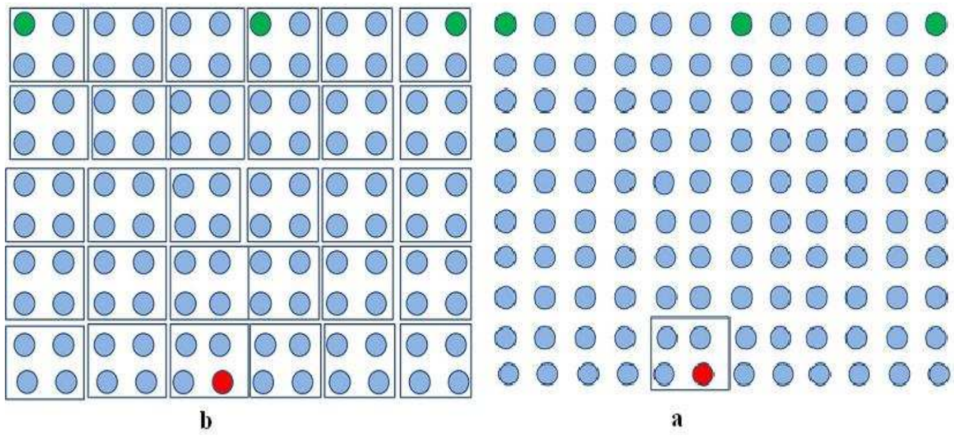


Figure 2: *a: before rejoining and b: after rejoining*

The approach in this research would be used in problems with one source and more than one sink in network. At first, the path that delivered the packets to sinks with directed diffusion was constructed (Figure 3).

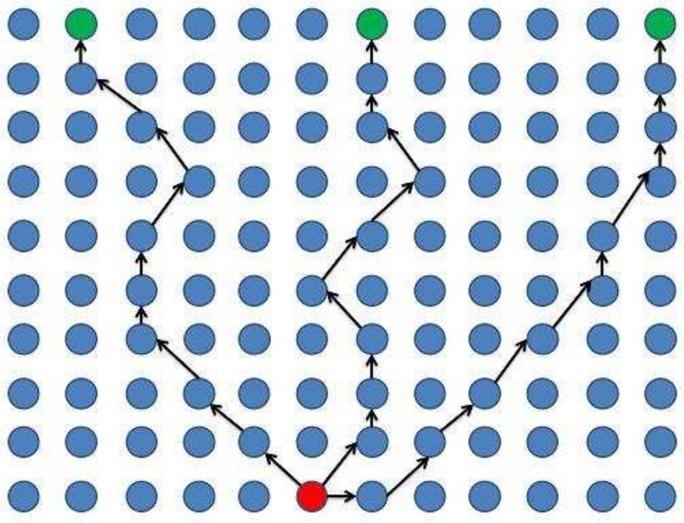


Figure 3: *Three paths and three sink with one source illustrated*

The goal in this approach is finding optimum path depended on current path for transmitting data to sinks. For this purpose, interface node was found using learning automata. Here, interface node is a node closer to sinks and often has high energy and is in delivery region. Delivery region is a region that packets transmit from source to sink from. This node can be supposed as a virtual sink which means after finding virtual sink, packets can transmit from source align one of the paths to this node (Virtual Sink) and then VS can transmit packets to real sinks.

In the proposed algorithm, each datum on packet has identical number called ID which is made by the source. ID was used for specifying the data type and each node can transmit various data and can understand which data has been passed. Any region has HOP. HOP is number of steps of each region from source. ID and HOP fields are added to packets that each region sends to its neighbors. The proposed algorithms have two actions and two perceptions from environment (network).

a is called the prize, grown to reach clearly to the goal and b is called the punishment rate that are between 0 and 1([0, 1]). Assume that a=0.1 and b=0.05 for the first. figure 4 illustrated our algorithm clearly.

As it is seen in Figure 5 distance(s, y) is equal to distance(y, x) and x cannot be selected so transmitting follows on x AND y continued on separate path to their sinks.

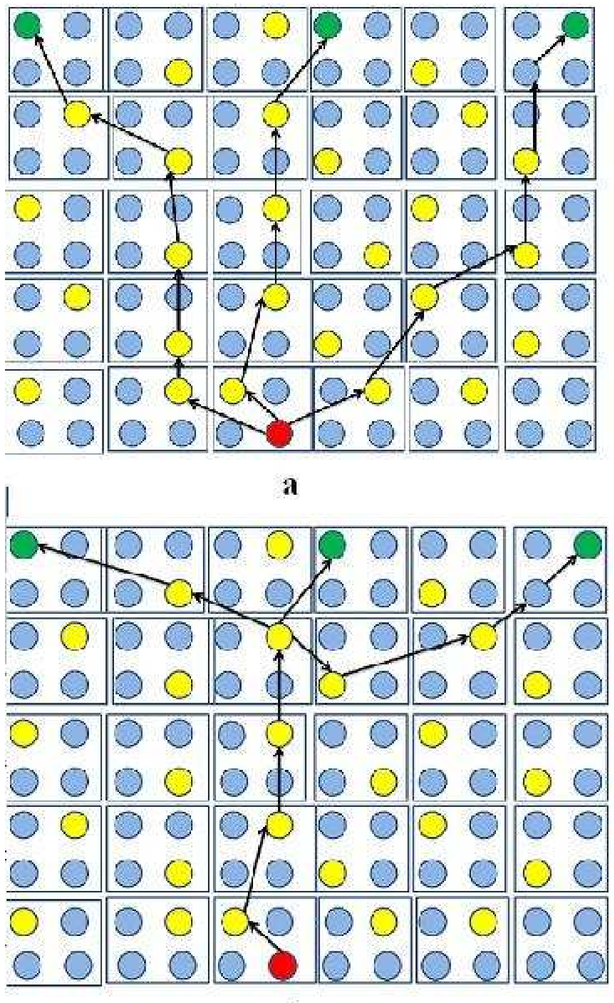


Figure 4: *a: illustrated paths before using RDDLA, b: illustrated paths after using RDDLA*

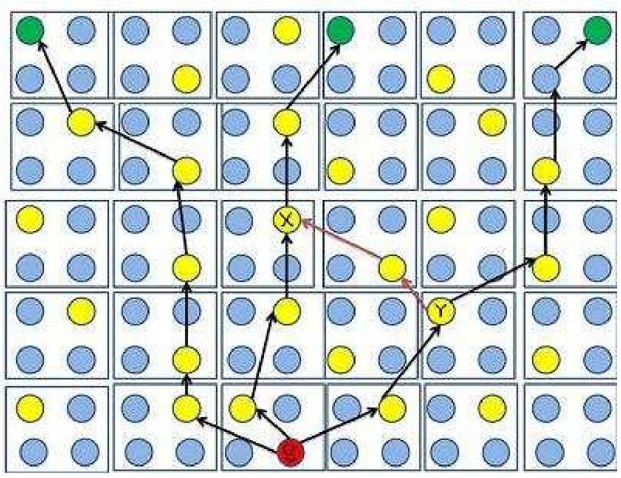
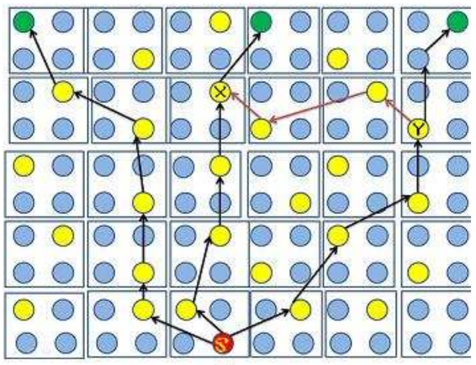


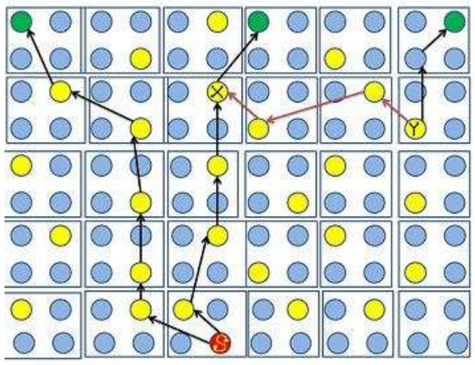
Figure 5: *Distance(s, y) = distance(y, x) = 2, so x would not selected as interface node*

Action one (α_1) was sending packet from node (i) to all its neighbors and the neighbors that are in delivery region and have the same ID (maybe each region Crossing different packets with different ID, in this case the focus is on one ID and the algorithm is used for distinct ID) reply its HOP to node (i). j is supposed a step movement from current region. Each sending to neighbor was increased j+1. a and b were increased 0.1 and 0.05. if the neighbor node was not in a delivery region it sent packets to its neighbors. In this case, a and b were increased 0.04 and 0.1 respectively, i.e (a+0.04) (b+0.1). This operation recursively run up to HOP (Current Region) became smaller than j and b>a. It is known that all sinks were in different ways and the interface node could not be found. Perception (β_1) was HOP and j of each region that called x return to Current Position. Action Two (α_2) was selecting x that its HOP was bigger than other neighbors and its j is smaller than HOP (Current Position) and had the same ID. Then all regions would be eliminated from source to current position and the new route was all regions from source to region(x) and all region between x to current position that it was j steps. Both a and b were increased 0.1. The reason of eliminating region from source to current position was that j region could be used instead of this region. Because, both of the paths are exists and we want to decrease some region that they are extra using for transmitting. The example of action two is illustrated in Figure 6 and Figure 7. Perception (β_2) was eliminating i region and highlighting j plus HOP region from source to x. the proposed algorithm was as follows: Find_MAX_Neighbor (y: current position, var X: our new region for transmitting stead of y) function was used for selecting best neighbor for sending packet in network.

```
Find_MAX_Neighbor(y, var x) {
Send data and bind fields (ID, HOP) to neighbors(y)
x:=neighbors(y);
a=a+0.1;
If x in delivery region and have same ID then a+0.1 and b+0.05
Else increase a+0.04 and increase b+0.1
If a>b then Find_MAX_Neighbor(x, neighbors(x)) and distance (x, y)>distance (source, y) then
{eliminate all region from source to y
temp:=source;
source:= Find_MAX_Neighbor(x, neighbors(x))
a:=a+0.1;
b:= b+0.1;
}
If a<b then NOP; b=b+0.2; //we cannot find interface node
If a> a/ (a+b) then
{Interface node=x;
Send packets from source to all regions between source to interface node and send these packets to all sink.
}
//here, we find interface node
}//end function
```

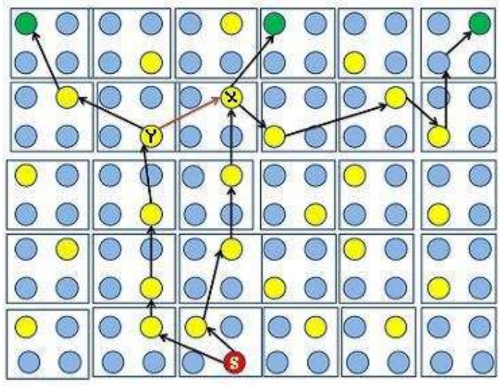


a

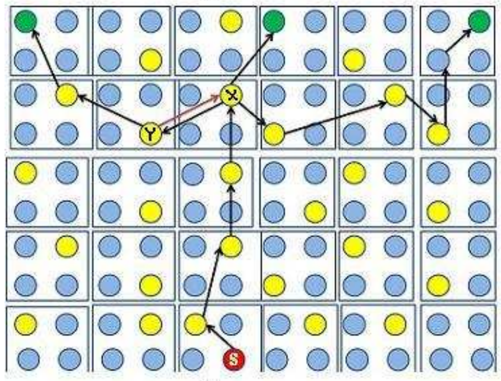


b

Figure 6: *a: Distance(s, y)=4 > distance(y, x) = 3, so x would selected as Interface node b: eliminate all region from s to y*



a



b

Figure 7: *a: Distance(s, y) =4 > distance(y, x) = 1, so x would selected as Interface node b: eliminate all region from s to y*

5. Performance Evaluation

In this section the comparison between basic directed diffusion and RDDLA has been illustrated. NS2 was used for simulating. Send energy rate was 0/660j and 0.395 j for data receiving. In RDDLA method, construct and repair routing tree were used and the overhead produced for packets into two phase and one phase would be minor. The proposed approach improved Basic Directed Diffusion for optimum path guarantee by finding interface node and sending packets to several sinks with interface node. The provided evaluation metrics were:

1) Computing network Traffic or Overhead Rate
2) Computing packet loss
3) Computing packet Average delay
4) Computing Remaining Energy

Overhead depended on time is illustrated in Figure 8. Two Phase Pull has 3 Phases thus has most overhead compare with others. One Phase Pull in Explorer Phase is rapidly from two phases Pull. So it had less Overhead. But, at first in RDDLA because of constructing Regions and running LA the overhead increased and after finding optimum path, overhead rate was decreased rapidly.

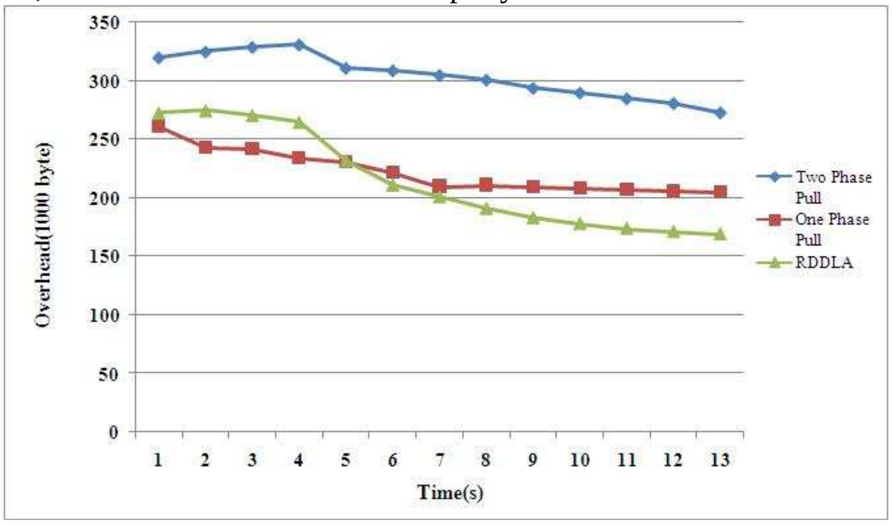


Figure 8: *Overhead Rate in Time*

First, Packet Delivery Delay in the proposed approach was more than two and one phase pull algorithms. Computing in RDDLA was more than others, after elapsing time and increasing the packet number in network, the effect of first delay rather decreased up to vanished. Event accurate time was the time that a packet was sent to sinks and average delay was the time it took that packet receive to sink. Packet delivery delay in time was illustrated in Figure 9.

The main reason of dropping packets in directed diffusion (one phase, two phases) was flooding interested packets and explorer data that it was avoiding receiving data to sink nodes. as it is shown in Figure 10, a few packets was dropped in RDDLA. It was less than 60%. It means more than 40% of packets reached clearly to sinks by increasing time. decreasing explorer data in RDDLA caused this rather than other algorithms.

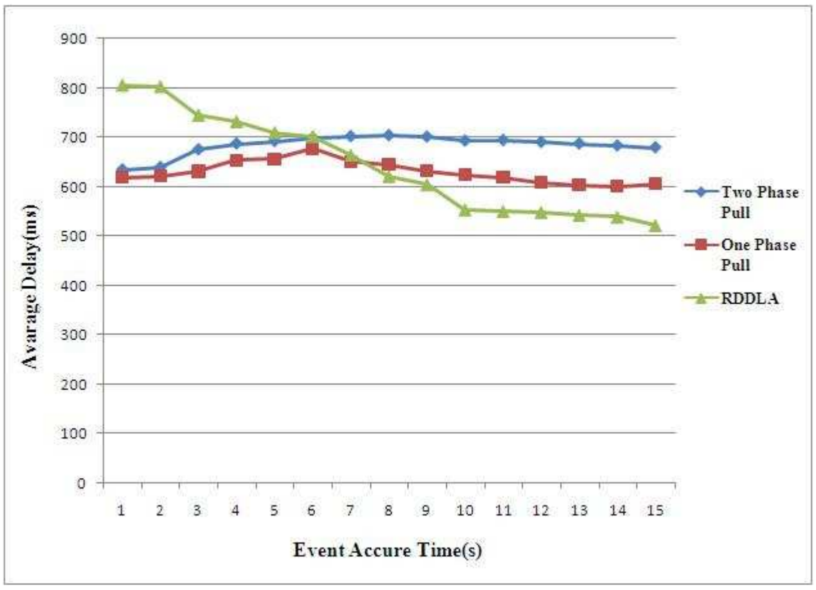


Figure 9: *Packet Average delay in time*

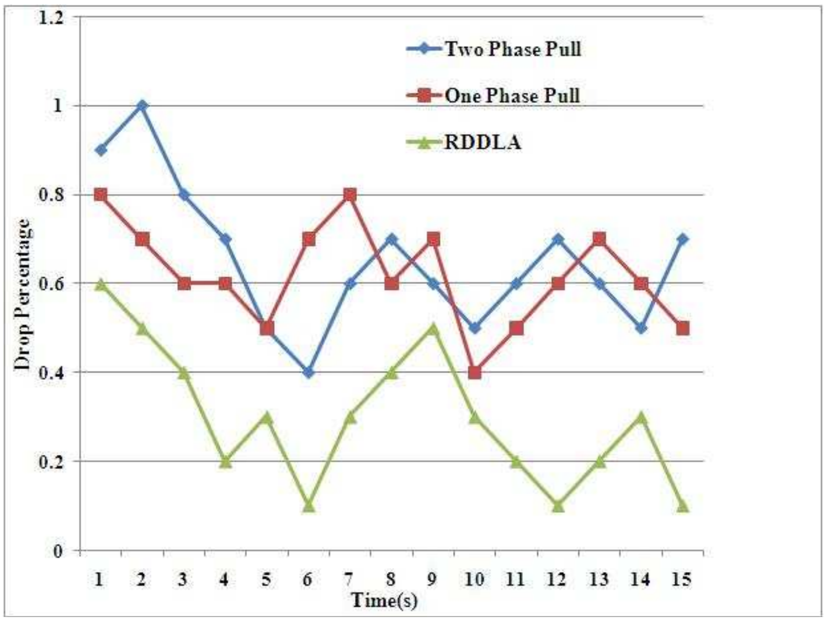


Figure 10: *Packet loss in network*

The remaining energy of network depending on time is shown Figure 11. Two phases and one phase pull have several paths to deliver packets. So, because of engaging more nodes the energy consumed in network is high, but, in the proposed approach extra regions were eliminated and energy saving occurred up to near 50 percent.

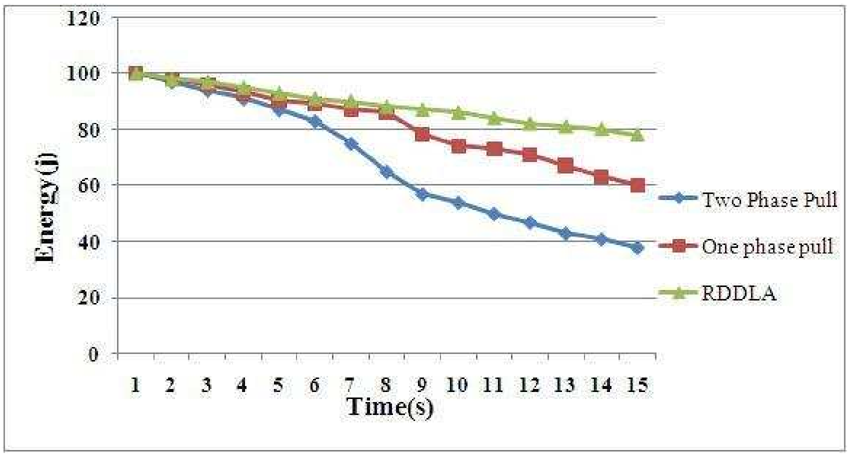


Figure 11: *Energy remaining in time*

6. Conclusions

In this paper, the most advantage of the proposed approach was preventing to build long multipath that causes wasting energy by finding interface node and selecting optimum path for sending packets to sinks. Another advantage of this algorithm was increasing fault tolerance because the proposed workbench had less nodes and it made less error in sending and less fault for nodes and network failure. Another benefit of RDDLA was rejoining. rejoining causes the nodes that are very close to each other silent which made saving energy in network. If both the number of sinks and the number of same sources increase, RDDLA decrease all metrics that because of overlapping regions. Assuming implementation of this algorithm for 10 sinks, improvement of RDDLA would be 51% that it is fantastic. The overall network metrics in DDLA compared with basic DD increased 22 percent to optimum state.

One of the disadvantages of the proposed approach was discharging interface node neighbors energy rapidly. Also, with existence decreasing overhead of routing in DD, in RDDLA used the same path for transmitting data (the path between interface node and source node) which caused discharging the path nodes between interface and source rapidly.

References

1) Ch. Intanagonwiwat, R.Govindan, D. Estrin, J. Heidemann, F. Silva, (2003). Directed Diffusion for Wireless Sensor Networking. Supported by DARPA (DABT63-99-1-0011), Vol. 11.Iss. 1, ISSN: 1063-6692, pp. 2-16.
2) Ch. Intanagonwiwat, R.Govindan, D. Estrin, (2000). Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. ACM MOBICOM 1-58113-197-6/00/08, MA, USA.
3) J. Li, Ch. Blake, D.S.J. Decouto, H.L. Lee, R.Morris. (2001) Capacity of Ad Hoc Wireless Networks. NTT-MIT Research, ISBN: 1-58113-422-3, Rome, Italy, pp. 61-69.
4) W, Chunfeng. (2004). An Introduction on Ad-hoc Networks. Mita Lab,Keio University.
5) D, Remondo. (2004) Tutorial on Wireless Ad Hoc Networks. HET-NETs 04, Second International Working Conference in performance Modeling and Evaluation of Heterogeneous Networks, Ilkley, U.K.
6) G. Jayakumar, G.Gopinath. (2007). Ad Hoc Mobile Wireless Networks Routing Protocols- a Review. Journal of computer Science 3(8):574-582, ISSN: 1549-3636, Science Publications.
7) A. D. Parker. (2004). Micro Diffusion: One-Phase-Pull in EmStar. CS213 Project, Advanced Topics in Distributed Systems.
8) B. Yu,P. Scerri,K. Sycara,Y. Xu, M. Lewis. (2006). Scalable and Reliable Data Delivery in Mobile Ad Hoc Sensor Networks. AAMAS '06, ISBN: 1-59593-303-4,Hakodate, Japan, pp. 1071-1078.
9) A. Förster, A. L. Murphy, J. Schiller, K. Terfloth. (2008). An Efficient Implementation of Reinforcement Learning Based Routing on Real WSN Hardware. Proceedings of the 4th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications(womb) ,Avignon, France, pp. 247-252.
10) I, Stojmenovic. (2005). Handbook of Sensor Networks algorithms and architectures, Wiley series on Parallel and distributed Computing, Wiley& Sons Publication, ISBN-13 978-0-471-68472-5, USA.
11) Y. Zhang,Y. Liu, F. Zhao. (2006). Information-Directed Routing in Sensor Networks Using Real-Time Reinforcement Learning. Journal of COMBINATORIAL OPTIMIZATION .DORDRECHT, Vol. 18,Springer US,ISSN:1388-3011,DOI:10.1007/0-387-29026-5-11, Netherlands, pp. 259-288.
12) Hamid Beigy. (2004). Intelligent Channel Assignment in Cellular Networks: A learning Automata Approach, A Dissertation for Doctor of Philosophy in CS Amirkabir Uni., Tehran, Iran.
13) A.S. Poznyak, K, Najim. (1997). Learning Automata and Stochastic Optimization, ISBN 3-540-76154-3 Springer-Verlag Berlin Heidelberg New York.
14) H. Garcia-Molina. (1982). Election in a Distributed Computing system. IEEE Transaction on Computers, C-31(1), pp.48-59.