Global Journal of Pure and Applied Mathematics.
ISSN 0973-1768 Volume 12, Number 2 (2016), pp. 1413-1432

http://www.ripublication.com

Optimized Reinforcement Learning Based Adaptive Network Routing for MANETs

Mr. Rahul Desai
Research Scholar, Sinhgad College of Engg
Asst Professor, Army Institute of Technology, Pune, Maharashtra, India.

Dr. B P Patil
Professor, Army Institute of Technology, Pune, Maharashtra, India.

Abstract

In this paper, optimized reinforcement learning based adaptive network routing is investigated. Shortest Path routing is most suitable network routing algorithm for wired network but not suitable for any wireless network. In high traffic conditions, shortest path routing algorithm will always select the shortest path (in terms of number of hops) between source and destination thus all packets from source to the destination will follow the same path and thus generates more congestion. There could be some alternate path which may not be shortest in the number of hops but packet might reaches to the designation in shortest amount of time. Thus proposed method is an adaptive network routing algorithm, where the path from source to the destination is selected based on actual traffic present on the network. Thus they guarantee the least delivery time to reach the packets to the destination. Analysis is done on a 6 by 6 irregular grid and an ad hoc network. Various performance parameters used for judging the network are packet delivery ratio and delay shows optimum results using the proposed method.

Keywords: Ad Hoc Network, Ad Hoc On Demand Distance Vector Routing (AODV), Destination Sequenced Distance Vector (DSDV), Dynamic Source Routing (DSR), Confidence Based Routing (CBQ) , Dual Reinforcement Q Routing (DRQ), Q Routing (QR)

Introduction

Information is transmitted in the network in form of packets. Routing is the process of transmitting these packets from one network to another. While transmitting the packets from source to the destination, a number of intermediate hops came in picture.

Various performance parameters are used to judge the quality of routing such as delay, packet delivery ratio, control overhead, throughput, jitter etc. Some of the most important parameter used for judging the quality is the delay and packet delivery ratio. Different routing algorithms such as shortest path routing, bellman ford algorithms are used. The most simplest and effective policy used is the shortest path routing. In shortest path routing the path with minimum number of hops is selected to deliver the packet from source to the destination. In shortest path routing, cost table and neighbor tables are present to store the appropriate information and tables are exchanged frequently for adaptation purpose.

The shortest path routing policy is good and found effective for less number of nodes and less traffic present on the network. But this policy is not always good as there are some intermediate nodes present in the network that are always get flooded with huge number of packets. Such routes are referred as popular routes. In such cases, it is always better to select the alternate path for transmitting the packets. This path may not be shortest in terms of number of hops, but this path definitely results in minimum delivery time to reach the packets to the destination because of less traffic on those routes. Such routes are dynamically selected in real time based on the actual traffic present on the network. Hence when the more traffic is present on some popular routes, some un-popular routes must be selected for delivering the packets. This is the main motivating factor for designing and implementing various adaptive routing algorithms on a network.

Learning such effective policy for deciding routes online is major challenge, as the decision of selecting routes must be taken in real time and packets are diverted on some unpopular routes. The main goal is to optimize the delivery time for the packets to reach to the destination and preventing the network to go into the congestion. There is no training signal available for deciding optimum policy at run time, instead decision must be taken when the packets are routed and packets reaches to the destination on popular routes.

Preliminary

Existing Routing Protocols for MANET

Ad Hoc networks are infrastructure less networks. These are consisting of mobiles nodes which are moving randomly. Figure 1 shows an ad hoc network where multiple hops are used to deliver the packets to the destination. Routing protocols for an ad hoc network are generally classified into two types - Proactive and On Demand. A proactive protocol maintains consistent, up to date routing information in a network. Updates are exchanged among all nodes throughout the network.

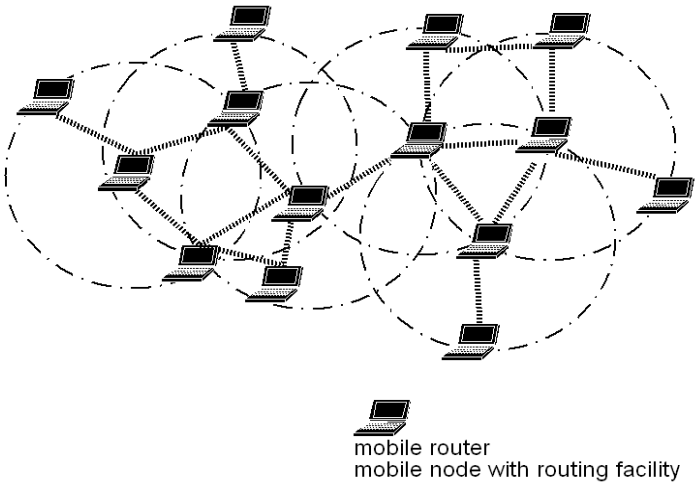

Figure 1: Example of Mobile Ad Hoc Network

These protocols always find the optimum routes to reach to every destination node. Destination sequenced Distance Vector (DSDV) is one of older protocol used for an ad hoc networks. It is based on distance vector algorithm and uses sequence numbers to avoid count to infinity problem. Every node communicates and finds out their neighbors by sending hello messages and exchanges their routing tables with them. Periodic full updates and small updates are also transmitted to maintain routing tables up to date. Optimized link state routing protocol (OLSR) is another proactive routing protocol based on link state algorithm. Here, every node broadcasts link state updates to every other node present in the network and thus creates link tables from which routing tables are designed. In order to reduce the overheads, multipoint relay concept is widely used.

Second type of routing protocol on an ad hoc network is on demand routing protocols which are also known as reactive routing protocols. These routing protocols maintain routes whenever required. In on demand routing protocols, route to the destination is obtained only when there is a need. When source nodes want to transmit data packets to the destination nodes, it initiates route discovery process. Route request (RREQ) messages float over the network and finally the packet reaches to the destination, Destination nodes replies with route replay message (RREP) and unicast towards the source node. All nodes including the source node keeps this route information in caches for future purpose. Dynamic Source Routing Protocol (DSR) is thus characterized by the use of source routing. The data packets carry the source route in the packet header. When the link or node goes down, existing route is no longer available; source node again initiates route discovery process to find out the optimum route. Route Error packets and acknowledgement packets are also used. Ad Hoc on Demand Distance Vector Routing (AODV) is also on demand routing protocol. AODV uses traditional routing tables, one entry per destination. This is in contrast to DSR, where DSR maintains multiple route cache entries for each destination.

Introduction to Reinforcement Learning

Reinforcement learning is learning where the mapping between situations to actions is carried out so as to maximize a numerical reward signal [1,2]. Fig 2 shows agent's interaction with the system. An agent checks the current state of system, chooses one action from those available in that state, observes the outcome and receives some reinforcement signal [3-5].

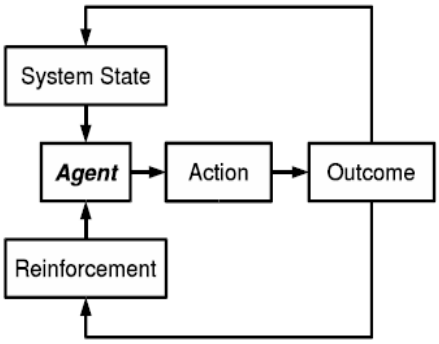

Figure 2: Reinforcement Learning Approach

Q Routing is one of the best reinforcement based learning algorithm. In this, each node contains reinforcement learning module which dynamically determines the optimum path for every destination [6-8]. Fig 3 illustrates the basic process of reinforcement learning. Let Qx(y, d) be the time that a node x estimates it takes to deliver a packet P to the destination node d through neighbor node y including the time that packet would have to spend in node x's queue. Upon sending packet to y, x gets back y's estimate for the time remaining in the trip. Upon receiving this estimate, node x computes the new estimate [9-10].

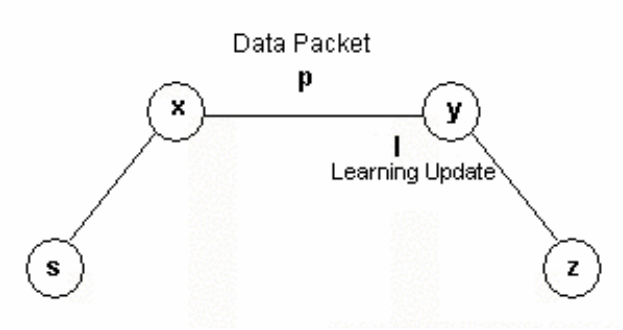

Figure 3: Reinforcement Learning – Q Learning

Fig 4 shows an algorithm for Packet Send and Packet Receive for standard Q Routing. Fig 5 shows Q routing forward exploration.

Algorithm 1: PacketSend(X)

1 Receive the packet from Packet Queue
2 Find out the best neighbor Y = min(Q_x(Y, D))
3 Forward Packet to the neighbor Y
4 Receive Estimate (Q_y(Z, D) + q_y) from node Y.
5 Update Q value Q_x(Y, D).

Algorithm 2: PacketReceive(Y)

1 Receive a packet from neighbor X
2 Calculate best estimate for node D; Q_y(Z, D) and send back to node X.
3 Get ready for receiving next packet

Figure 4: Reinforcement Learning – Q Learning

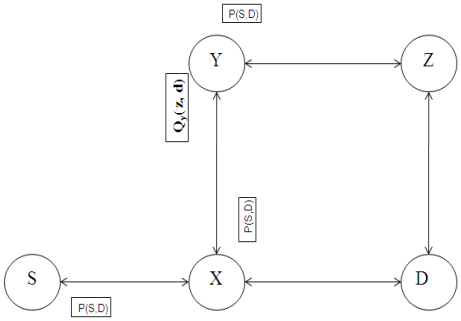

Figure 5: Q routing Forward Exploration

Flow chart for Packet Send function at node X and packet receive function at Node Y for Q routing is as illustrated in Fig 6 and Fig 7 respectively.

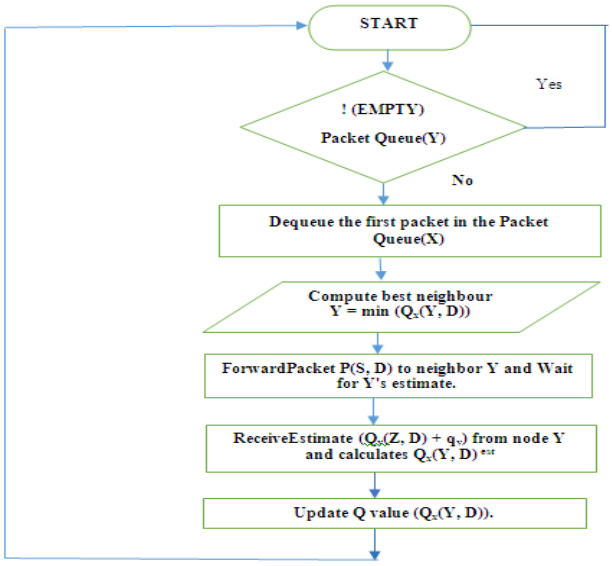

Figure 6: Flowchart for Packet Send at Node X for Q routing

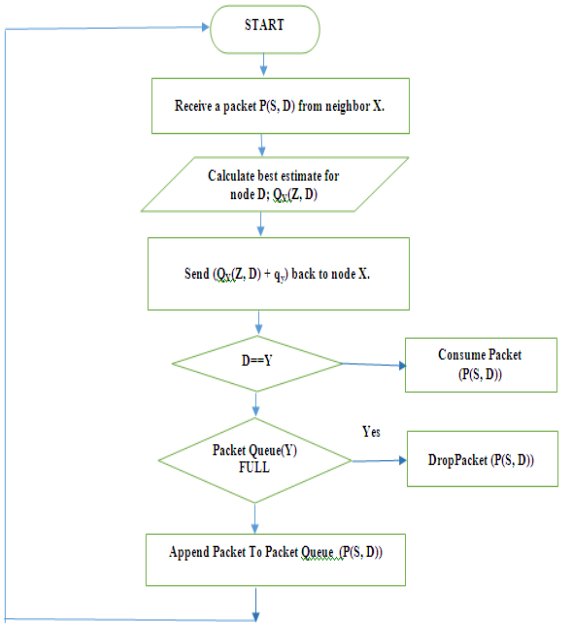

Figure 7: Flowchart for Packet Receive at Node Y for Q routing

In another optimized form, Confidence Based Q Routing (CBQ), each Q value is associated with confidence value (real number between 0 and 1). This value essentially specifies the reliability of Q values. All Intermediate nodes along with Q value, also transmits C values which will be updated in confidence table. Fig 8 shows Confidence based Q routing forward exploration. [9-11].

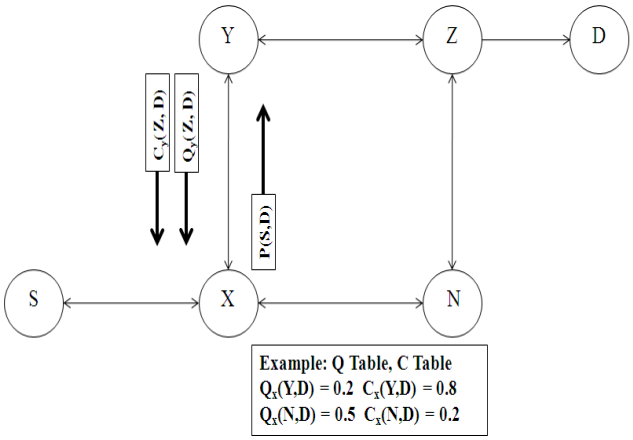

Figure 8: Confidence Q routing

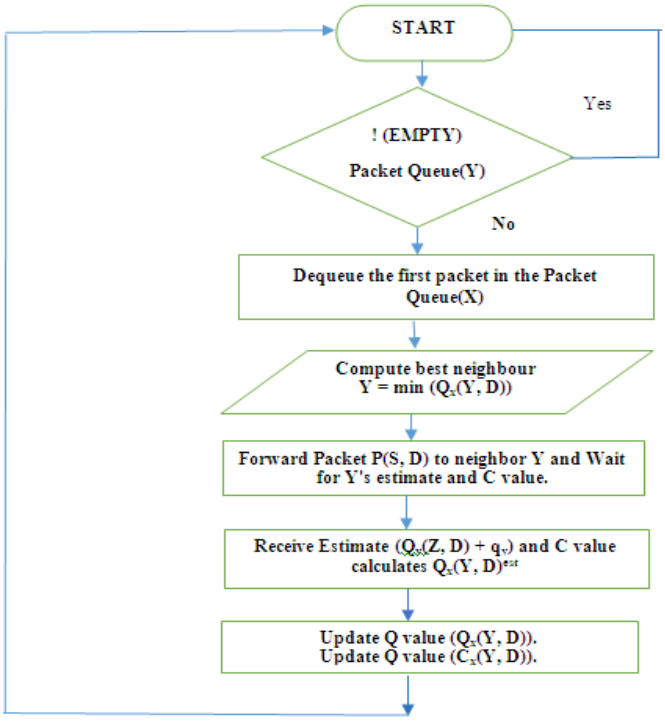

Figure 9: Flowchart for Packet Send at Node X for CBQ routing

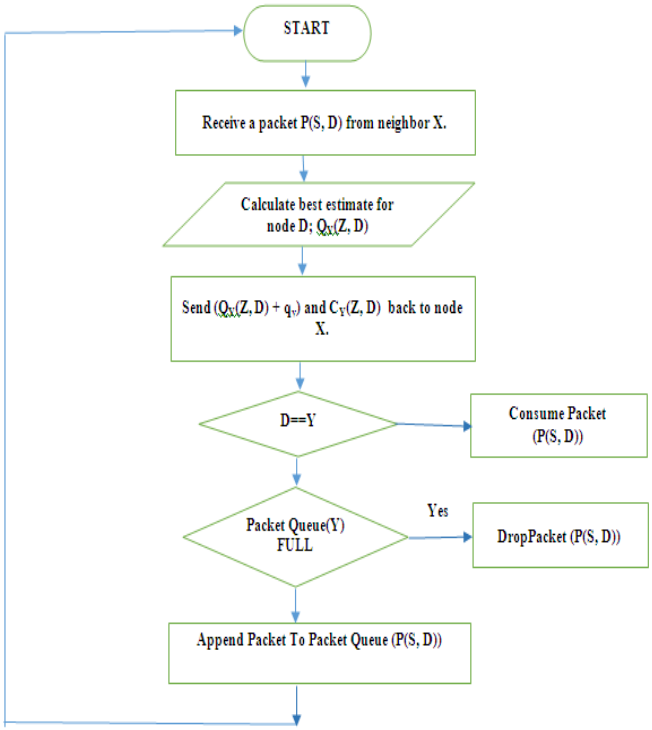

Figure 10: Flowchart for Packet Receive at Node Y for CBQ routing

Flow chart for Packet Send function at node X and packet receive function at Node Y for Confidence Based Q routing is as illustrated in Fig 9 and Fig 10 respectively. Fig 11 shows an algorithm for Packet Send and Packet Receive for CBQ Routing.

Algorithm 1: PacketSend(X)

1. Receive the packet from Packet Queue
2. Find out the best neighbor Y = min(Q_x(Y, D))
3. Forward Packet to the neighbor Y
4. Receive Estimate (Q_y(Z, D) + q_y) and C_y(Z, D) from node Y.
5. Update Q value Q_x(Y, D) and C_y(Z, D) value.

Algorithm 2: PacketReceive(Y)

1. Receive a packet from neighbor X
2. Calculate best estimate for node D; Q_y(Z, D) and send back to node X.
3. Find the corresponding confidence value C_y(Z, D) and send back to node X.
4. Get ready for receiving next packet

Figure 11: Reinforcement Learning – CBQ Learning

Dual reinforcement Q Routing (DRQ) is another optimized version of the Q Routing, where learning occurs in both ways. Performance of DRQ routing almost doubles as learning occurs in both directions. Fig 12 shows forward and backward exploration involved in Q learning process.

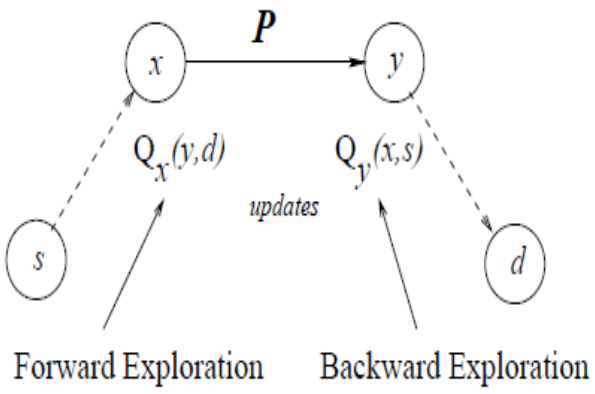

Figure 12: Forward and Backward Exploration

Fig 13 shows DRQ routing which involves backward exploration.

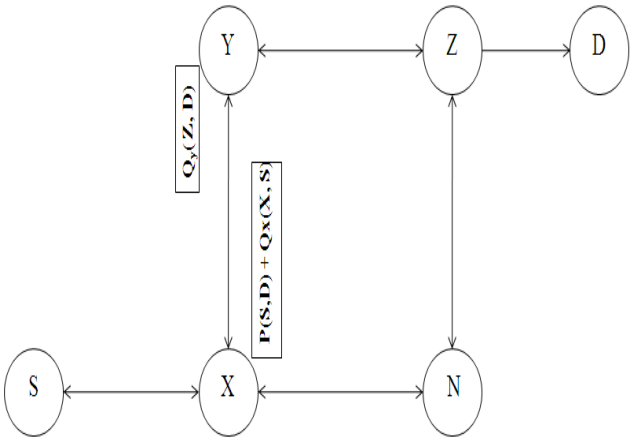

Figure 13: Backward Exploration

Flow chart for Packet Send function at node X and packet receive function at Node Y for DRQ routing is as illustrated in Fig 14 and Fig 15 respectively. Fig 16 shows an algorithm for Packet Send and Packet Receive for DRQ Routing.

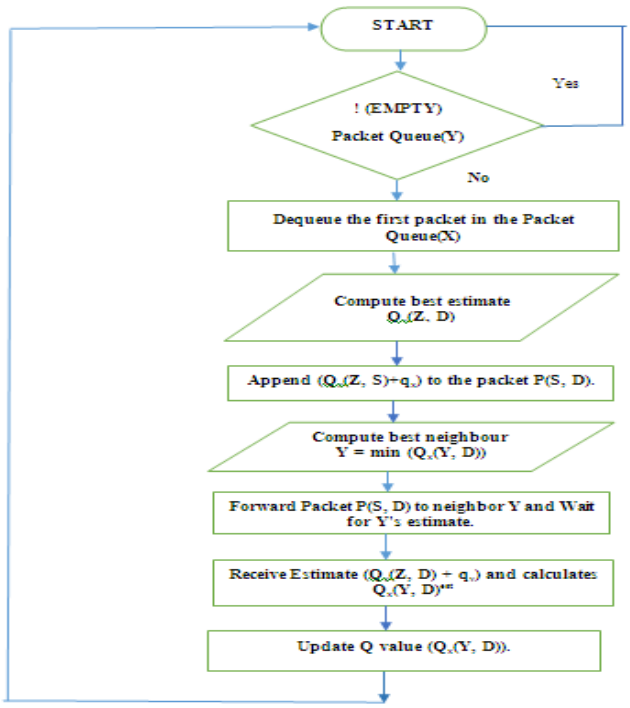

Figure 14: Flowchart for Packet Send at Node X for DRQ routing

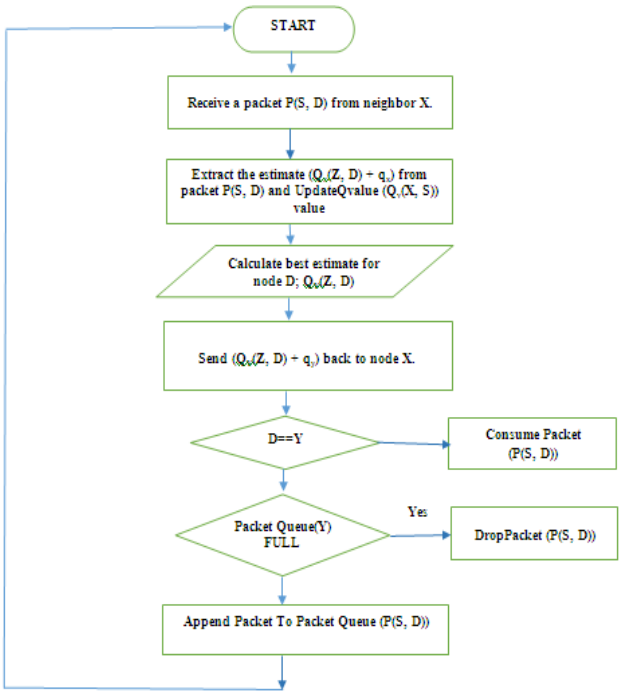

Figure 15: Flowchart for Packet Receive at Node Y for DRQ routing

Algorithm 1: PacketSend(X)

1. Receive the packet from Packet Queue
2. Find out the best estimate $Q_x(Z, d)$
3. Append $(Q_x(Z, S)+q_x)$ and $C_x(Z, S)$ to the packet P(S, D).
4. Find out the best neighbor $Y = \min(Q_x(Y, D))$
5. Forward Packet to the neighbor Y
6. Receive Estimate $(Q_y(Z, D) + q_y)$ and $C_y(Z, D)$ from node Y.
7. Update Q value $Q_x(Y, D)$ and C value $C_x(Y, D)$.

Algorithm 2: PacketReceive(Y)

1. Receive a packet from neighbor X
2. Using the received estimate $Q_x(Z, D) + q_x$ and $C_x(Z, D)$ Update Q value $Q_y(X, S)$ and $C_y(X, S)$.
3. Calculate best estimate for node D; $Q_y(Z, D)$ and $C_y(Z, D)$, send back to node X.
4. Get ready for receiving next packet.

Figure 16: Reinforcement Learning – DRQ Learning

Proposed Method – Optimization of Reinforcement Learning

Mostly, a packet has multiple possible routes to reach to its destination. The decision of selecting best route is very important in order to reach the packets to the destination having a least amount of time and without packet loss. This selection has three main challenges, first, coordination and proper communication among nodes in a network

is always required. Second, link and node failure cases should be handled gently. Third and very most important in dynamic environment, routes must be able to change dynamically according to the state of the network [4-6].

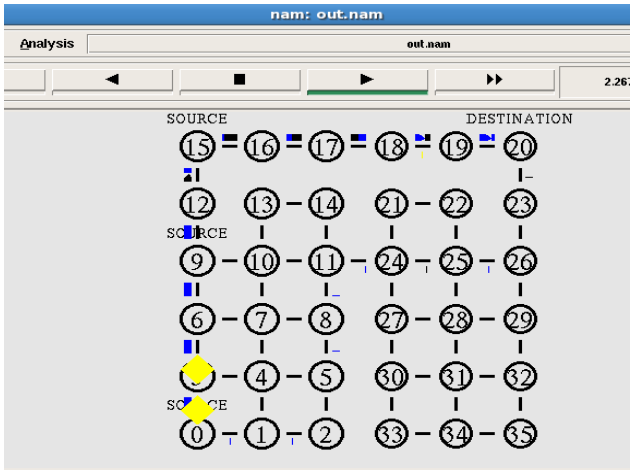

Figure 17: Limitation of Shortest Path Algorithms

For example, in order to demonstrate limitation of shortest path algorithms (fig 17), consider that Node 0, Node 9 and Node 15 are simultaneously transferring data to Node 20. Route having nodes 15-16-17-18-19-20 gets flooded with huge number of packets and then it starts dropping the packets. Thus shortest path routing is non-adaptive routing algorithm that does not take care of traffic present on some popular routes of the network.

In CBQ routing algorithm, confidence values of non-selected nodes are updated with decay constant value – λ. Also it is necessary that learning rate should not be constant and should change based on different network conditions. If Q value changes then it will select a new node having minimum Q value. As illustrated in fig 18(a), Node A has two neighbours, one neighbour who is selected for transmission as next hop to some destination, while other one is non-selected node. The node having Q value not selected for long duration, corresponding C value decays and thus corresponding Q value becomes unreliable[13].

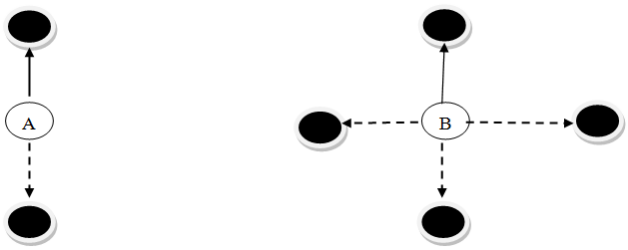

Figure 18: Node A and Node B with two and four node connections respectively.

As illustrated in fig 18(b), where B has four nodes, 3 nodes whose Q value not selected for long time, thus unselected nodes becomes more unreliable to transmit packets. Node A may not need a high learning rate as compared to node B as node A is updating its Q values more frequently compared to node B[13]. A simple solution is by introducing the Variable of Decay Constant Approach. For selected nodes, decay constant will be λ while for non-selected nodes decay constant will be $\lambda^{\{1-(n-1)\}}$ n >=2, where n is the number of node connections. Thus update rule for selected connection will be

$C_x(Y, D)^{new} = C_x(Y, D)^{old} + \max (C_Y(Z, D), 1 - C_x(Y, D)^{old}) * (C_Y(Z, D) - C_x(Y, D)^{old})$

The update rule for non-selected connection where Z is non-selected node will be

$C_x(Z, D)^{new} = \lambda^{\{1-(n-1)\}} C_x(Z, D)^{old}$

In CBQ routing only the confidence value of selected nodes but in this optimized version of confidence based Q routing, confidence value of non selected nodes are also updated. Confidence value decides the reliability of Q values and they are never used for selecting routing policy in CQ routing. C value of 1 indicates 100% reliability while C value of 0 indicates absolute no reliability. If the confidence value is 0.85, indicates that corresponding Q value is 15% less reliable as compared with its most reliable value. Using this approach, the non-selected Q values are also updated using the equation more competitive for selection and more exploration will occur[13].

These non-selected Q values will be updated as follows:

$Q_A(Z, D)^{new} = Q_A(Z, D)^{old} – \{(1- C_A(Z, D)^{new}) \} Q_A(Z, D)^{old}$

Fig 19 shows an algorithm for Packet Send and Packet Receive for proposed method. Flow chart for Packet Send function at node X and packet receive function at Node Y for proposed method is as illustrated in Fig 20 and Fig 21 respectively.

Algorithm 1: PacketSend(X)

1. Receive the packet from Packet Queue
2. Find out the best estimate $Q_x(Z, d)$
3. Append $(Q_x(Z, S)+q_x)$ and confidence value to the packet P(S, D).
4. Find out the best neighbor Y = min($Q_x(Y, D)$)
5. Forward Packet to the neighbor Y.
6. ReceiveEstimate $(Q_y(Z, D) + q_y)$ and C value from node Y.
7. UpdateQvalue ($Q_x(Y, D)$) for selected and non-selected nodes.
8. UpdateCvalue ($C_x(Y, D)$) for selected and non-selected nodes.
9. Get ready to send next packet (goto 1).

Algorithm 2: PacketReceive(Y)

1. Receive a packet P(S, D) from neighbor X
2. Extract the estimate $(Q_x(Z, D) + q_x)$ from packet P(S, D).
3. UpdateQvalue ($Q_y(X, S)$) value.
4. Calculate best estimate for node D; $Q_y(Z, D)$
5. Send $(Q_y(Z, D) + q_y)$ and Confidence value back to node X.
6. UpdateCvalue ($C_x(Y, D)$) for selected and non-selected nodes.
7. If (D = Y) then ConsumePacket(Y) else goto 5.
8. If (PacketQueue(Y) is FULL) then DropPacket (P(S, D)) else goto 8.
9. AppendPacketToPacketQueue$_y$ (P(S, D))
10. Get ready for receiving next packet (goto 1).

Figure 19: Reinforcement Learning – DRQ Learning

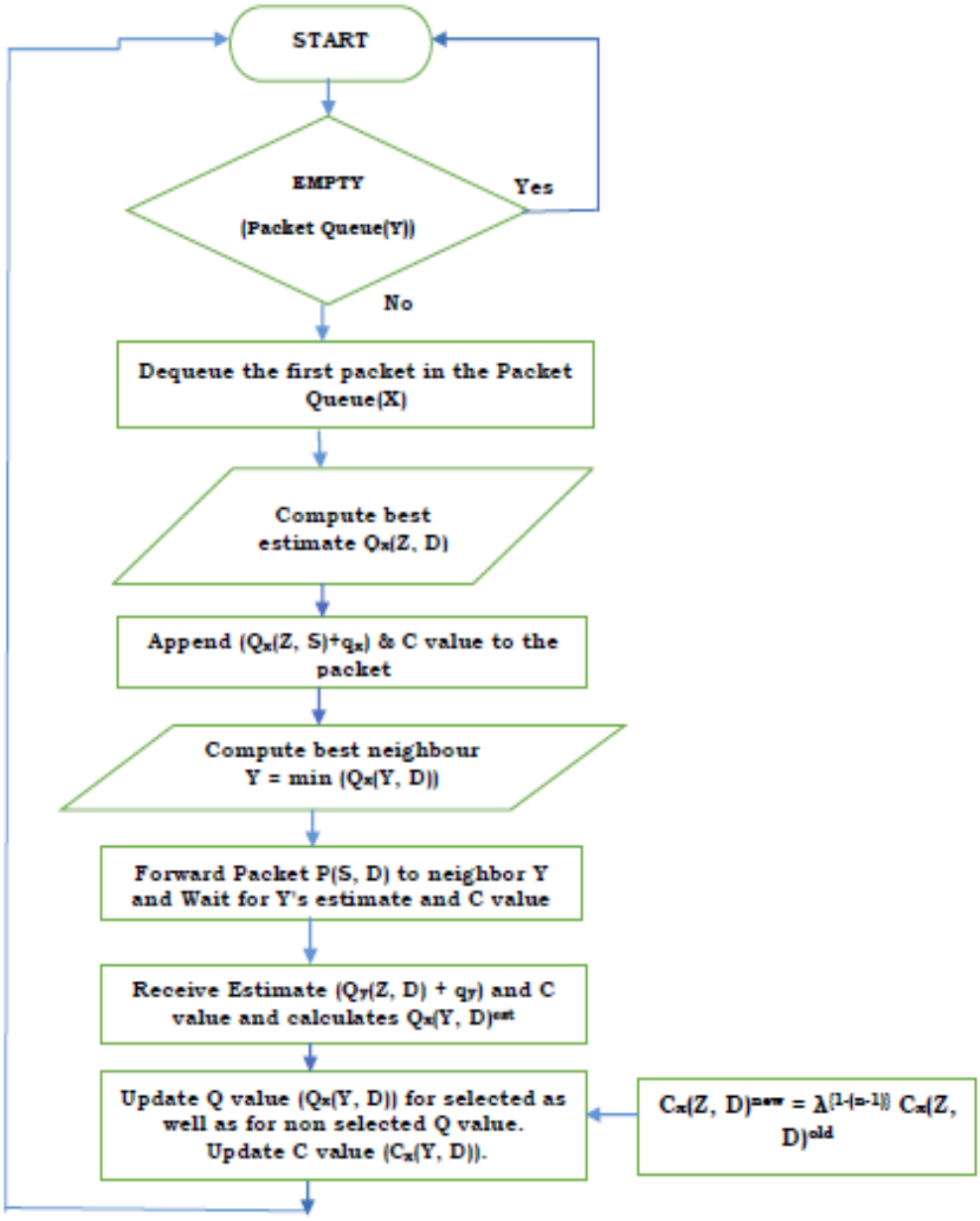

Figure 20: Flowchart for Packet Send at Node X for Proposed Method

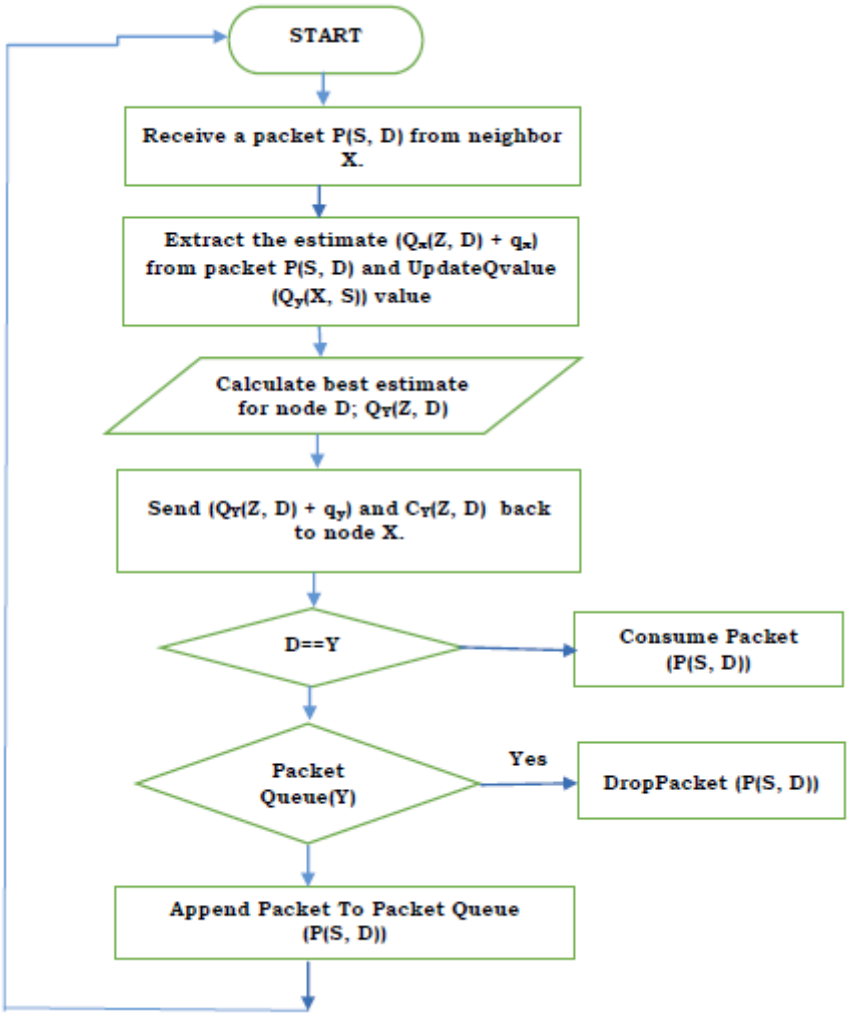

Figure 21: Flowchart for Packet Received at Node Y for Proposed Method

Results and Analysis

Three different experiments are performed to judge the quality of reinforcement learning algorithms using different performance parameters, in first and second experiment 6×6 irregular grid is used to test the performance of reinforcement learning for random traffic. Third experiment is performed on dynamic environment i.e. ad hoc network consisting of 10 to 100 nodes with random mobility of nodes and random traffic generated on the network. In first and second experiment, the network topology used is the 6×6 irregular grid shown in the fig 22.

In first experiment, proposed method is compared with CBQ routing. Average packet delivery time is used as an performance parameter. In second experiment proposed method is compared with shortest path routing, simple reinforcement and CBQ routing. In third experiment, existing routing protocols on an ad hoc network such as DSDV, AODV and DSR are compared with proposed method.

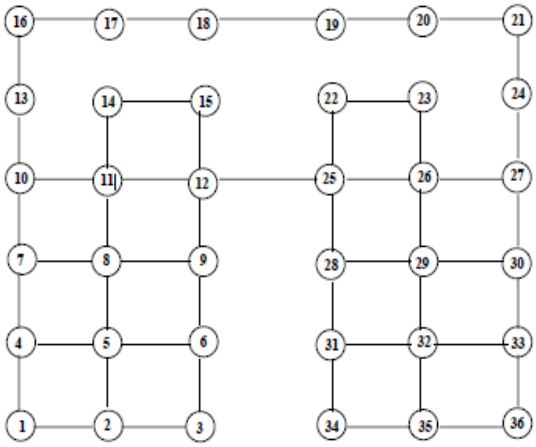

Figure 22: The 6×6 Irregular Grids

Simulation parameters used in experiment 1 and experiment 3 are listed in Table 1 and Table 2 respectively.

Table 1: Simulation Parameters Used for Experiment 2.

Parameter	**Value**
Type of Network	Fixed Network – Static Environment
Number of nodes	36 Nodes
Packet Size	2000 bytes
Simulation time	200 s , 5000 Simulation steps
Interval	0.6 to 1.0 increment by 0.1
Routing protocols analysed	Shortest path Routing, Reinforcement routing, CBQ routing, Proposed method
Analysed Parameters	Packet Delivery Ratio and Delay

Table 2: Simulation Parameters Used for Experiment 3.

Parameter	**Value**
Type of Network	Dynamic Environment – Ad Hoc Network
Number of nodes	10 Nodes to 100 Nodes
Mobility model	Random Way Point Mobility Model (default)
Simulation time	200 s
Initial Energy	100 Joules
Topology Size	1000×1000
Routing protocols analysed	DSDV, AODV, DSR and Proposed Method
Analysed Parameters	Packet Delivery Ratio and Delay

In first experiment, average numbers of packets generated at each node are 0.1 for low network traffic and 0.5 for large network traffic. The CBQ routing is compared with proposed method. At low network traffic load (fig 23), the average packet delivery times (APDT) shows that proposed method performs far better than CBQ routing. The maximum value recorded for CBQ routing and proposed methods are 825 and 2210 respectively for low network loads.

At high network traffic load, (fig 24), the average packet delivery time shows that proposed method slightly better than CBQ routing algorithm. The maximum value recorded for CBQ routing and proposed methods are 2250 and 2680 respectively for low network loads. Overall the proposed routing algorithm out performs the CBQ routing algorithm.

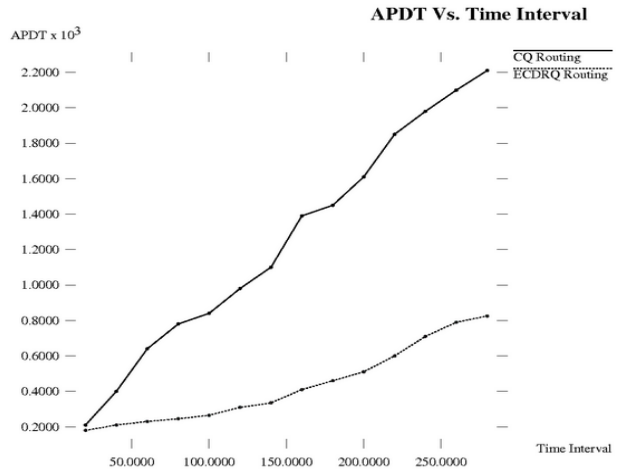

Figure 23: APDT vs. Simulation Time Interval for Low Load.

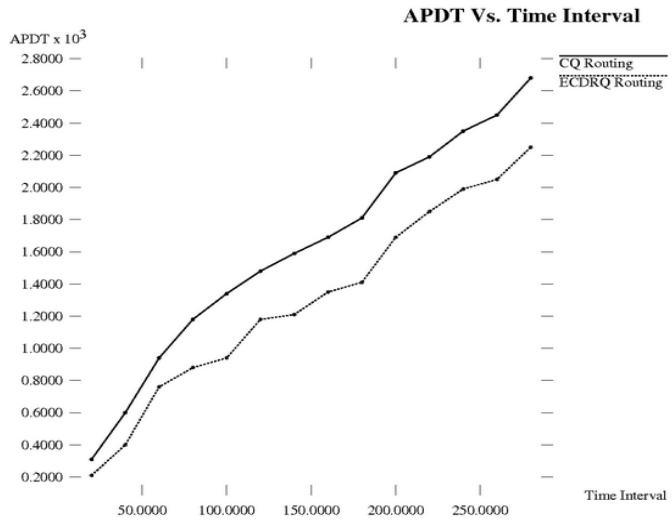

Figure 24: APDT vs. Simulation Time Interval for High Load.

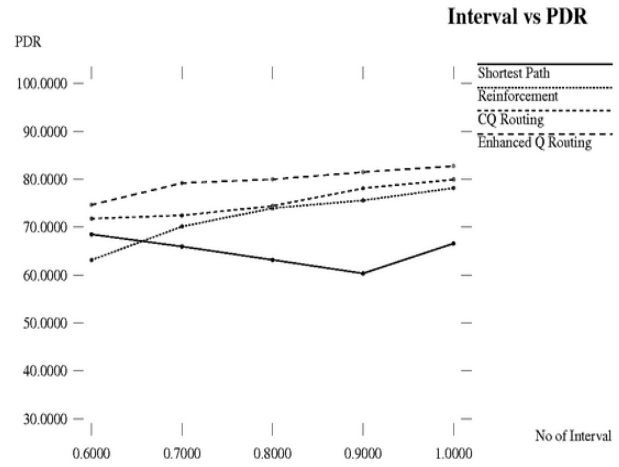

Figure 25: Interval vs. PDR

In fig 25, shortest path, standard Q routing, CBQ routing and proposed method is compared by varying the intervals between successive packets. CBQ routing performance is better than reinforcement routing as it also includes confidence values for reliability of Q values. Proposed method provides the good results and PDR lies somewhere at 85% and remains throughout constant.

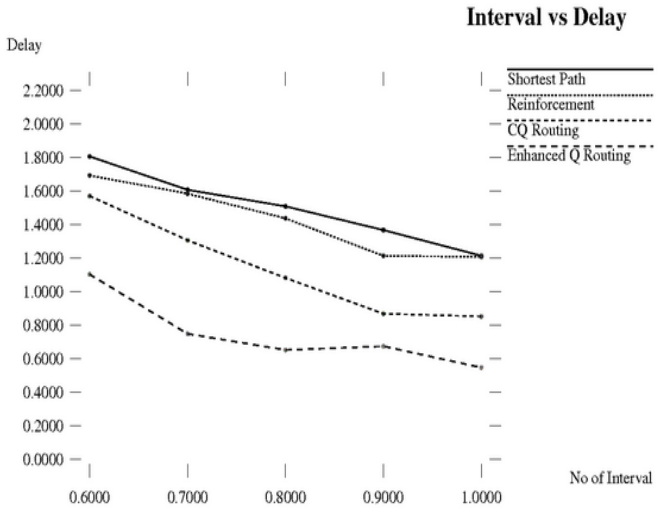

Figure 26: Interval vs. Delay

End-to-end Delay is the time taken by a data packet to reach to the destination [14-15]. The result of end to end delay for experiment 2 is illustrated in Fig 26. Proposed method will provide minimum delay for the packets to reach to the destination.

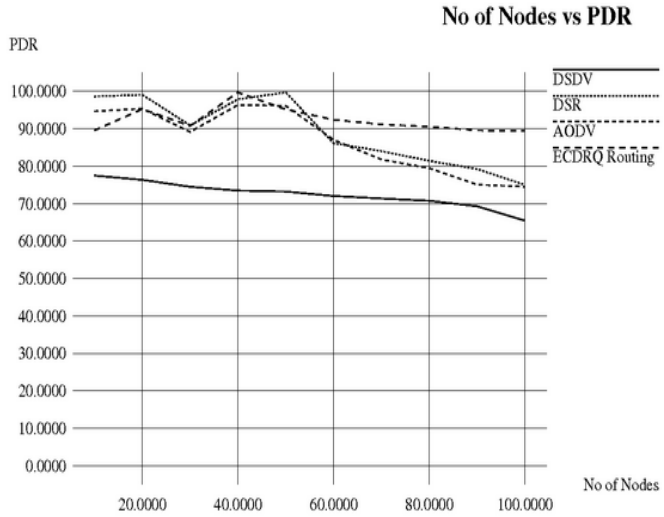

Figure 27: No of Nodes Vs PDR.

The result of Packet delivery ratio is illustrated in Fig 27. Instead of static environment, dynamic environment is considered here. No of nodes are varying from 10 to 100. Existing protocols DSDV, AODV and DSR are compared with proposed method. It is observed that when the network size increases beyond 60 nodes, AODV or DSR protocols starts dropping packets. But proposed method maintains consistent ratio throughout the network irrespective of the network size. Thus almost 85% to 90% PDR is obtained.

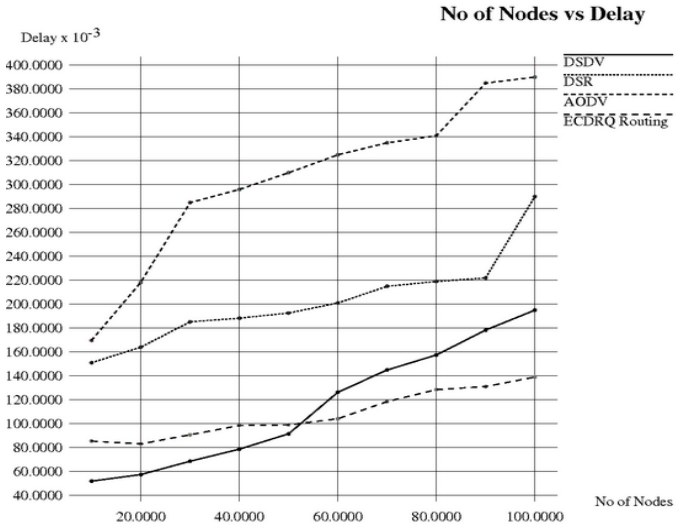

Figure 28: No of Nodes vs. Delay

The result of end to end delay for experiment 3 is illustrated in Fig 28. DSDV is the proactive routing protocol which always provides minimum delay, AODV and DSR protocols are on demand routing protocols hence they took more delay for route discovery process. Proposed method provides alternate path whenever required and thus takes care that packet reaches to the destination in minimum amount of time.

Conclusion

In this paper, various reinforcement learning algorithms were presented. Confidence based reinforcement and dual reinforcement routing are showing prominent results as compared with shortest path routing for medium and high load conditions. At high loads, dual reinforcement Q routing performs more than twice a fast as Q-Routing. For an ad hoc network, AODV protocol gives good performance at low loads but at high mobility and heavy load situations, it does not provide optimum results. In DRQ routing, as backward exploration is involved including confidence measure, less time is required in order to settle down the Q values thus they more accurately predict the state of network at run time. It is found that, though mobility rate changes at high rate as well as high traffic, dual reinforcement routing obtains more accurate result as compared with Q routing. This paper presents optimization of Reinforcement Learning and compares the performance with existing routing protocols. This research study compares DSDV, AODV and DSR protocols with proposed method for a static network as well as for dynamic network. Average packet delivery time is the basic parameter used for static network while PDR and delay are used to decide the reliability of proposed method. In our simulation environment PDR and delay in proposed method outperforms AODV and DSR routing protocols with almost 90-95% without packet loss with lower delay.

References

[1] Fahimeh Farahnakian. "Q-learning based congestion-aware routing algorithm for onchip network", 2011 IEEE 2nd International Conference on Networked Embedded Systems for Enterprise Applications, 12/2011

[2] Milos Rovcanin, Eli De Poorter, Ingrid Moerman, Piet Demeester,"A reinforcement learning based solution for cognitive network cooperation between co-located, heterogeneous wireless sensor networks" Ad Hoc Networks, Volume 17, June 2014, Pages 98-113

[3] Amr El-Mougy, Mohamed Ibnkahla,"A cognitive framework for WSN based on weighted cognitive maps and Q-learning", Ad Hoc Networks, Volume 16, May 2014, Pages 46-69

[4] Jianjun Niu, Zhidong Deng,"Distributed self-learning scheduling approach for wireless sensor network",Ad Hoc Networks, Volume 11, Issue 4, June 2013, Pages 1276-1286

[5] Donato Macone, Guido Oddi, Antonio Pietrabissa,"MQ-Routing: Mobility-, GPS- and energy-aware routing protocol in MANETs for disaster relief scenarios",Ad Hoc Networks, Volume 11, Issue 3, May 2013, Pages 861-878

[6] Oussama Souihli, Mounir Frikha, Mahmoud Ben Hamouda, "Load-balancing in MANET shortest-path routing protocols", Ad Hoc Networks, Volume 7, Issue 2, March 2009, Pages 431-442

[7] Ouzecki, D.; Jevtic, D., "Reinforcement learning as adaptive network routing of mobile agents," MIPRO, 2010 Proceedings of the 33rd International Convention , pp.479,484, 24-28 May 2010

[8] Ramzi A. Haraty and Badieh Traboulsi "MANET with the Q-Routing Protocol" ICN 2012 : The Eleventh International Conference on Networks

[9] S Kumar, Confidence based Dual Reinforcement Q Routing : An on line Adaptive Network Routing Algorithm. Technical Report, University of Texas, Austin 1998.

[10] Kumar, S., 1998, "Confidence based Dual Reinforcement Q-Routing: An On-line Adaptive Network Routing Algorithm," Master's thesis, Department of Computer Sciences, The University of Texas at Austin, Austin, TX-78712, USA Tech. Report AI98-267.

[11] Kumar, S., Miikkulainen, R., 1997, "Dual Reinforcement Q-Routing: An On-line Adaptive Routing Algorithm," Proc. Proceedings of the Artificial Neural Networks in Engineering Conference.

[12] Shalabh Bhatnagar, K. Mohan Babu "New Algorithms of the Q-learning type" Science Direct Automatica 44 (2008) 1111- 1119. Website: www.sciencedirect.com.

[13] Soon Teck Yap and Mohamed Othman, "An Adaptive Routing Algorithm: Enhanced Confidence Based Q Routing Algorithms in Network Traffic. Malaysian Journal of Computer, Vol. 17 No. 2, December 2004, pp. 21-29

[14] Desai, Rahul, and B P Patil. "Cooperative reinforcement learning approach for routing in ad hoc networks", 2015 International Conference on Pervasive Computing (ICPC), 2015.

[15] Desai, Rahul, and B P Patil. "Reinforcement learning for adaptive network routing", 2014 International Conference on Computing for Sustainable Global Development (INDIACom), 2014.