Journal of Computer Science 4 (2): 80-86, 2008
ISSN 1549-3636
© 2008 Science Publications

Ant Colony Based Node Disjoint Hybrid Multi-path Routing for Mobile Ad Hoc Network

[1]B. Kalaavathi, [2]K. Duraiswamy
[1]Department of Information Technology, K.S. Rangasamy College of Technology, KSR Kalvi Nagar, Tiruchengode-637215, India
[2]Dean, K.S. Rangasamy College of Technology, KSR Kalvi Nagar, Tiruchengode-637215, India

Abstract: Mobile Ad hoc Networks are characterized by multi-hop wireless links, without any infrastructure and frequent node mobility. A class of ant colony based routing protocols has recently gained attention because of their adaptability to the network changes. *AntHocNet* is ant colony based hybrid algorithm, which combines reactive path setup with proactive path probing, maintenance and improvement. Multi-path routing represents a promising routing method for mobile ad hoc network. Multi-path routing achieves load balancing and is more resilient to route failures. This research introduces node disjoint multi-path property to *AntHocNet* routing algorithm. A virtual class room is one that can be established by using mobile devices and whose members can be dynamically added or removed. The implementation of virtual class room for lesson handling and query discussion using mobile ad hoc network is analyzed. The data are spread among the N (here N = 3) node disjoint routes from the beginning of the data transmission session. The performance metrics, average end-to-end delay, packet delivery ratio and load balancing have been analyzed for various pause times. The average end-to-end delay and packet delivery ratio have not varied significantly. Optimal load distribution is achieved by spreading the load among different node disjoint routes.

Keywords: Mobile ad hoc network, ant colony optimization, personal digital assistant, virtual class room, anthocnet

INTRODUCTION

A mobile ad hoc network is a collection of mobile nodes that can communicate with each other using multi-hop wireless links without utilizing any fixed base station infrastructure and centralized management. Each node in the network acts as both a host and a router. The design of an efficient and reliable routing protocol in such a network is a challenging issue. Existing IP protocols can be classified either as proactive or reactive[1]. Proactive protocols attempt to continuously evaluate all the routes within a network-so that when a packet needs to be forwarded, a route is already known and can be immediately used. It is well known that proactive protocols are not optimal for Mobile Ad Hoc Networks (MANET) which have rapidly changing topologies. In contrast, reactive routing protocols invoke a route determination procedure on-demand only. Thus, if route is needed some sort of global search procedure is used. On-demand routing protocols in particular, are widely developed because they consume much less bandwidth than proactive protocols. Ad hoc On-Demand Distance Vector (*AODV*)] and Dynamic Source Routing *(DSR*) are the two most widely studied on-demand ad hoc routing protocols. The limitation of these protocols is that both of them build and rely on a unipath route for each data session. Whenever there is a link break on the active route, each of the two routing protocols has to invoke a route discovery process. Each route discovery flood is associated with significant latency and overhead. In addition, reactive protocols introduce additional latency for real-time traffic. Hybrid routing protocols that use a mix of both proactive and reactive routing techniques are proposed to combine the merits of both and overcome their shortcomings. In practice, many algorithms are hybrid algorithms[1] (e.g., *ZRP*-Zone Routing Protocol. *HARP*-Hybrid Ad Hoc Routing Protocol, *AntHocNet*[2]) using both proactive and reactive components. In *ZRP* and *HARP*, each node maintains only routing information for those nodes that are within its zone and its neighbouring zones. They exhibit proactive behaviour within a zone and reactive between zones. The route to each destination within a

Corresponding Author: B. Kalaavathi, Department of Information Technology, K.S. Rangasamy College of Technology, Tiruchengode 637215, India

zone is easily established without delay, while a route discovery and route maintenance are required for every other destination. *AntHocNet* is a hybrid algorithm based on the framework of Ant Colony Optimization (ACO). It does not maintain paths to all destinations at all times, but sets up paths when they are needed at the start of a session. This is done in a reactive path setup phase, where ant agents called reactive forward ants are used. While a data session is going on, the paths are probed, maintained and improved proactively using different ant agents called proactive forward ants. *AntHocNet* reacts to link failures with either local repair or by warning preceding nodes on the paths.

Multipath routing consists of finding multiple routes between a source and destination node. Multipath routing protocols can attempt to find node disjoint, link disjoint or non-disjoint routes. Multiple paths can provide load balancing, fault-tolerance and higher aggregate bandwidth[3].

This research *ACNDHMR* - Ant Colony based Node Disjoint Hybrid Multi-path Routing proposes to integrate node disjoint feature into the *AntHocNet* to provide load balancing, fault- tolerance and reduce end-to-end delay. The basic idea of the ACO is taken from food searching behaviour of real ants. When ants are on the way to search for food, they start from their nest and walk toward the food. While walking, ants deposit a pheromone, which ants are able to smell, which helps to mark the route taken. The concentration of pheromone on a certain path is an indication of its usage. With time the concentration of pheromone decreases due to diffusion effects. This is important because it integrates the dynamic property into the path searching process. In the next section the previous work related ant based routing algorithms are briefly reviewed. Then the *ACNDHMR* is described. In the results and discussions section the *ACNDHMR* performance is compared with the *AntHocNet* by means of implementation of Virtual Class Room[4]. Here the data packets are spread among N (N = 3) number of optimal routes. Finally the result of the work done is summarized

ANT BASED ROUTING PROTOCOLS

ACO approaches are inspired by the problem solving paradigms of ants rather than building exact replicas of biological ants. In addition, there have been several works that apply to the idea of ant routing. The first ACO routing algorithms were designed for wired networks (e.g., *AntNet*[1] for packet switched networks and *ABC*[1] for circuit-switched networks). These algorithms exhibit a number of interesting properties which are also desirable for MANET routing: they can work in a fully distributed way, are highly adaptive to network and traffic changes, provide multi-path routing and data spreading. However the fact that they crucially rely on repeated path sampling which can cause significant overhead if not dealt with carefully. There have already been some attempts to design ACO routing algorithms for MANETs. Examples are *ARA*[1] , *PERA*[1], *Termite*[1] and *AntHocNet*[2].

However, these algorithms *ARA*, *PERA* and *Termite* loose much of the proactive sampling and exploratory behaviour of the original ant-based algorithms.

AntHocNet is a hybrid routing algorithm consisting of both reactive and proactive components. When a communication session is started at a source node s, this node starts a reactive path setup phase, in which ant agents called reactive forward ants are spread over the network in order to find the destination d of the session. They follow existing routing information if available and are otherwise broadcast. The first ant to find d becomes a reactive backward ant which returns to s, setting up entries in the pheromone tables of intermediate nodes indicating a path between s and d. They indicate in a node the quality of going over a certain next hop to reach a certain destination. During the duration of a communication session, the algorithm tries to proactively improve the existing paths. To facilitate this process, nodes include pheromone information they have about active destinations in their hello messages. A node is considered an active destination if data have recently been forwarded to it. Hello messages are short messages broadcast to all neighbors at regular intervals. The pheromone information forwarded from node to node in hello messages spreads over the network in a process which is termed as pheromone diffusion, in analogy with the volatile and diffusive character of ant pheromone in nature. The pheromone diffusion creates a field indicating possible paths to the destinations. This field, which is built up of bootstrapped information, contains potentially unreliable information and it is important to verify it before it can be used safely by data. To this end, the source node of each communication session periodically sends out proactive forward ants during the course of the session, which follow the diffused pheromone in order to find new and better paths. In this way, the single path which is set up during the route setup phase is extended to a mesh of multiple paths. Data are spread concurrently across these paths, with a strong preference for the best path. Link failures are dealt with either by using local route repair or by warning preceding nodes on the paths.

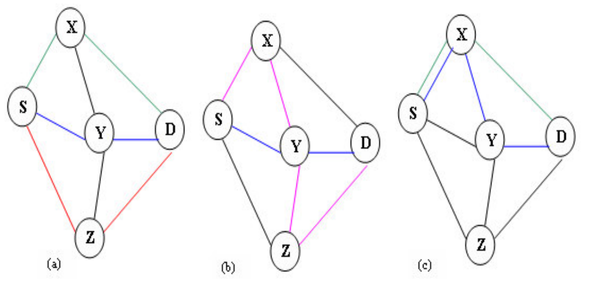

Fig. 1: (a): Node disjoint; (b): Link disjoint and (c): Non-disjoint

However, the multiple routes generated by the *AntHocNet* are non-disjoint routes. Non-disjoint routes can have nodes and links in common. Refer to Fig.1 for examples of the different kinds of multi-path routes. Disjoint routes offer certain advantages over non-disjoint routes. For instance, non-disjoint routes may have lower aggregate resources than disjoint routes, because non-disjoint routes share links or nodes. In principle, node disjoint routes offer the most aggregate resources, because neither links nor nodes are shared between the paths. Disjoint routes also provide higher fault-tolerance. When using non-disjoint routes, a single link or node failure can cause multiple routes to fail. In node or link disjoint routes, a link failure will only cause a single route to fail. However, with link disjoint routes, a node failure can cause multiple routes that share that node to fail. Thus, node disjoint routes offer the highest degree of fault-tolerance. The main advantage of non-disjoint routes is that they can be more easily discovered. Because there are no restrictions that require the routes to be node or link disjoint, more non-disjoint routes exist in a given network than node or link disjoint. Because node disjointedness is a stricter requirement than link disjointedness, node-disjoint routes are the least abundant and are the hardest to find. It has been found[10] that in moderately dense networks, there may only exist a small number of node disjoint routes between any two arbitrary nodes, especially as the distance between the nodes increases. This is because there may be sparse areas between the two nodes that act as bottlenecks. Given the trade-offs between using node disjoint versus non-disjoint routes, node disjoint routes offer a good compromise between the two. In the following subsection, some of the proposed multi-path protocols for finding node disjoint, link disjoint and non-disjoint paths are reviewed. After a source begins sending data along multiple routes, some or all of the routes may break due to node mobility and/or link and node failures. In unipath routing, route maintenance must be performed in the presence of route failures. In the case of multi-path routing, route discovery can be triggered each time one of the routes fails or only after all the routes fail. Waiting for all the routes to fail before performing a route discovery would result in a delay before new routes are available. This may degrade the QoS of the application. However, initiating route discovery every time one of the routes fails may incur high overheads. Performing route discovery when *N* routes fail, where N is less than the number of paths available, may be a compromise between the two options.

Split Multipath Routing [*SMR*] proposed in[12] is an on-demand multi-path source routing protocol. SMR is similar to DSR and is used to construct maximally disjoint paths. Unlike DSR, intermediate nodes do not keep a route cache and therefore, do not reply to route requests (RREQ). This is to allow the destination to receive all the routes so that it can select the maximally disjoint paths. Maximally disjoint paths have as few links or nodes in common as possible. Duplicate RREQs are not necessarily discarded. Instead, intermediate nodes forward RREQs that are received through a different incoming link and whose hop count is not larger than the previously received RREQs. SMR algorithm only selects two routes. However, the algorithm can be extended to select more than two routes. In the algorithm, the destination sends a route reply (RREP) for the first RREQ it receives, which represents the shortest delay path. The destination then waits to receive more RREQs. From the received RREQs, the path that is maximally disjoint from the shortest delay path is selected. If more than one maximally disjoint path exists, the shortest hop path is selected. If more than one shortest hop path exists, the path whose RREQ was received first is selected. The destination then sends an RREP for the selected RREQ.

AOMDV[13] is an extension to the AODV protocol for computing multiple loop-free and link-disjoint paths. To keep track of multiple routes, the routing entries for each destination contain a list of the next-hops along with the corresponding hop counts. All the next hops have the same sequence number. For each destination, a node maintains the advertised hop count, which is defined as the maximum hop count for all the paths. This is the hop count used for sending route advertisements of the destination. Each duplicate route advertisement received by a node defines an alternate path to the destination. To ensure loop freedom, a node only accepts an alternate path to the destination if it has a lower hop count than the advertised hop count for that destination. Because the maximum hop count is used, the advertised hop count therefore does not change for the same sequence number. When a route advertisement is received for a destination with a greater sequence number, the next-hop list and advertised hop count are reinitialized. *AOMDV* can be used to find node-disjoint or link-disjoint routes. To find node-disjoint routes, each node does not immediately reject duplicate RREQs. Each RREQ arriving via a different neighbor of the source defines a node-disjoint path. This is because nodes cannot broadcast duplicate RREQs, so any two RREQs arriving at an intermediate node via a different neighbor of the source could not have traversed the same node. In an attempt to get multiple link-disjoint routes, the destination replies to duplicate RREQs regardless of their first hop. To ensure link-disjointness in the first hop of the RREP, the destination only replies to RREQs arriving via unique neighbors. After the first hop, the RREPs follow the reverse paths, which are node-disjoint and thus link-disjoint. The trajectories of each RREP may intersect at an intermediate node, but each takes a different reverse path to the source to ensure link-disjointness.

AODVM[14] is an extension to *AODV* for finding multiple node disjoint paths. Intermediate nodes are not allowed to send a route reply directly to the source. Also, duplicate RREQ packets are not discarded by intermediate nodes. Instead, all received RREQ packets are recorded in an RREQ table at the intermediate nodes. The destination sends an RREP for all the received RREQ packets. An intermediate node forwards a received RREP packet to the neighbor in the RREQ table that is along the shortest path to the source. To ensure that nodes do not participate in more than one route, whenever a node overhears one of its neighbors broadcasting an RREP packet, it deletes that neighbor from its RREQ table. Because a node cannot participate in more than one route, the discovered routes must be node-disjoint. *ACNDHMR* is designed with three components[3] route discovery, route maintenance and traffic allocation and they are discussed in the following subsections.

Route discovery: When a source node s starts a communication session with a destination node d, it broadcasts a reactive forward ant. At each node, the forward ant is broadcasted, provided it is not a duplicate. The duplicate forward ants are ignored by the intermediate nodes, which avoids loop and results in certain number of node disjoint routes at the destination. The destination d receives all the incoming forward ants and maintains their route in a table. Each forward ant keeps a list of the nodes it has visited. Upon arrival at the destination d, it is converted into a backward ant, which travels back to the source retracing the path. At each intermediate node i, coming from neighbor n, the ant updates the entry T^i_{nd} which is the pheromone value indicating the estimated goodness of going from i over neighbour n to reach the destination d, in the i's pheromone table. The way the entry is updated depends on the path quality metrics used to define pheromone variables. For instance, if the pheromone is expressed using the number of hops as a measure of goodness, at each hop the backward ant increments an internal hop counter and uses the inverse of this value to locally assign the value τ^i_d which is used to update the pheromone variable T^i_{nd} as follows,

$$T^i_{nd} = \gamma T^i_{nd} + (1-\gamma)\tau^i_d \quad \gamma \in [0,1]$$

γ was set to 0.7 in the experiments.

Route maintenance: During the course of a communication session, source node s periodically sends out proactive forward ants to update the information about currently used paths and try to find new and better paths. However, the received proactive forward ant's paths are compared with the already stored paths by the destination d for achieving node disjointness. The destination d sends proactive backward ants only for the ants with the disjoint paths and others are ignored. The proactive ants follow pheromone and update pheromone tables in the same way as reactive forward ants. Such continuous proactive sampling of paths is the typical mode of operation in ACO routing algorithms.

Traffic allocation: Once the source node has selected a set of paths to the destination, it can begin sending data to the destination along the paths. The traffic allocation strategy used deals with how the data is distributed amongst the paths. The choice of allocation granularity is important in traffic allocation. The allocation[7] granularity specifies the smallest unit of information allocated to each path. For instance, a per-connection granularity would allocate all traffic for one connection to a single path. A per-packet granularity would distribute the packets from multiple connections amongst the paths. A per-packet granularity results in the best performance. This is because it allows for finer control over the network resources. It is difficult to evenly distribute traffic amongst the paths in the per-connection case, because all the connections experience different traffic rates. If a round-robin traffic allocation approach is used, however, a per-packet granularity may result in packets arriving out-of-order at the destination. Packet reordering is an issue that needs to be dealt with in multi-path routing, possibly at the transport layer. Nodes in ACNDHMR forward data stochastically according to the pheromone values. In order to exploit a maximum number of the available routes, route optimality should be considered in such a way to transmit more packets on the most optimal routes, thereby dispersing data load over a maximum number of nodes. The number of node disjoint routes N (here N = 3) is assigned. Data are spread based upon the probability of that route.

Link Failures: Nodes can detect link failures (e.g., a neighbor has moved away) when unicast transmissions (of data packets or ants) fail, or when expected hello messages were not received. When a neighbor is assumed to have disappeared, it removes the neighbor from its neighbor list and all the associated entries from its routing table. If the event was a failed transmission of a control packet, the node broadcasts a link failure notification message. All its neighbors receive the notification and update their pheromone table. If the number of node disjoint route N becomes 1 or 0, the route finding process is reinitiated.

RESULTS AND DISCUSSION

The above mentioned algorithm is implemented to form a Virtual Class Room (VCR). A virtual classroom[4] is one that can be immediately established and whose members can be dynamically added or removed; the group structure of the members can be reorganized dynamically. Figure 2 illustrates such an idea. The ad hoc classroom can support urgent and timely learning activities, thus improving learning effectiveness. For example[9], a teacher may establish a virtual classroom from his residence, students located around can take the opportunity to form an ad hoc group to improve the teaching learning process at any time using IEEE802.11g. VCR based on ad hoc network has been constructed[8] as shown in Fig. 2. The network has been formed with 30 PDA nodes. Each node in the network is assigned with static IP address. The software components used for development are Microsoft Visual Studio C#.Net 2005, Windows Mobile 5.0 Pocket PC SDK, Microsoft ActiveSync Version 4.2 and Microsoft.Net Compact Framework 2005 and XML technology. The XML technology was used for providing description and representation of data and control packets.

The application allows the user to initiate a query session with the peer or to lesson handling. The lesson

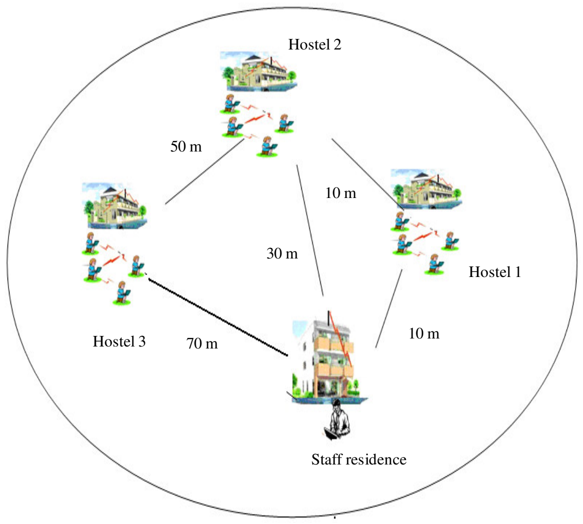

Fig.2: A scenario of VCR using MANET

file can include multimedia data like image, audio and video. Whenever a student (who is source of the communication session) wants to discuss any topic with another student or with a staff (who is the destination of the communication session), he can initiate a query session by selecting the destination from the member list displayed. To support transfer any type of file, UUEncode (Unix to Unix Encode) is used to convert the file contents into ASCII characters, which can be transmitted over the network. At the destination side UUDecode is used to get back the original contents of the file.

Figure 3 depicts the average end-to-end delay of *AntHocNet* and *ACNDHMR* for different pause times. It is larger for low pause time in which the nodes are highly mobile and it is smaller for high pause times. The drop in delay for high pause time is due to proactive path improvements, which is based on path delay.

Figure 4 shows the packet delivery ratio of *AntHocNet* and *ACNDHMR* for different pause times. The drop in delivery ratio for the highest pause times is due to connectivity issues. The nodes can become disconnected from the network for a certain time. For the highest pause times, these periods of lost connectivity, in which no packets can be delivered, can be long.

Figure 5 depicts the standard deviation (SD) of the routing load. The SD of the routing load in *ACNDHMR* is observed to be low compared to *AntHocNet* especially when the mobility is more (when the pause time is less). This is because with high mobility, link breaks are common which leads to local route repair or

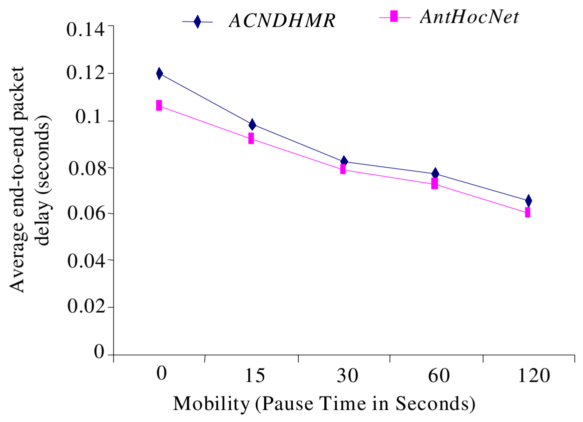

Fig. 3: Average End-to-End packet delay for various pause times

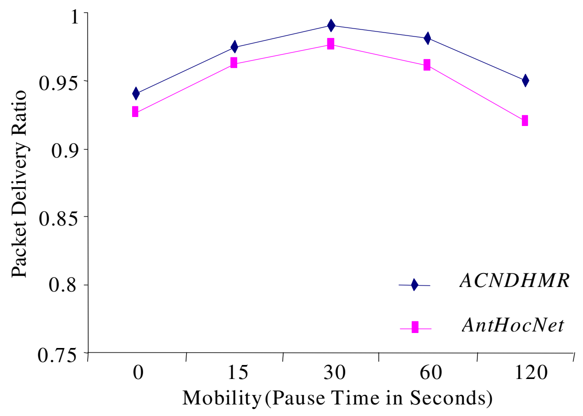

Fig. 4: Packet delivery ratio for various pause times

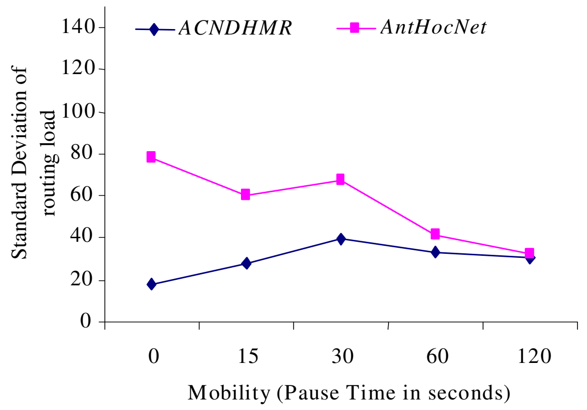

Fig. 5: Standard deviation of routing load for various pause times

link failure notification to the source nodes along the path. Upon link break, *AntHocNet* needs to initiate new route request and this adds significantly to the routing overhead. The low SD of *ACNDHMR* shows that every node in the network has almost equal routing overhead. This increases the mean time to failure of the nodes and the stability of the network.

CONCLUSION

In this research, the Ant Colony based Node Disjoint Hybrid Multi-path Routing Protocol has been presented. *ACNDHMR* is a hybrid routing protocol that establishes multiple node disjoint routes between a source destination pair and switches between the routes based on a heuristic. This heuristic takes the current network conditions (like hop count, propagation delay, queuing delay and node disjointness) into consideration. Providing multiple paths is useful in ad hoc networks because when one of the routes is disconnected, the source can simply use other available routes without performing the route discovery process again. The experimental results indicate that *ACNDHMR* outperforms *AntHocNet* because multiple routes provide robustness to mobility. The performance difference becomes evident since the load is distributed evenly to all the nodes, which help in network stability.

ACKNOWLEDGEMENTS

We would like to acknowledge All India Council for Technical Education, India for providing the grant to support this project under Research Promotion Scheme. Grant number is 8023/RID/BOR/RPS-79/2005-2006

REFERENCES

1. Changlin Liu and Jorg Kaiser, 2005. Technical Report. A Survey of Mobile Ad Hoc Network Protocols, University of Magdebur, Germany.
2. Gianni Di Caro, Frederick Ducatelle and Luca Maria Gambardella, 2005. AntHocNet: An adaptive nature-inspired algorithm for routing in mobile ad hoc networks. Eur. Trans. Telecommun., 16: 443-455.
3. Stephen Muller, Rose P. Tsang and Dipak Ghosal, 2004. Invited research in Lecture Notes in Computer Science, 2004. Multipath Routing in Mobile Ad Hoc Networks: Issues and Challenges.
4. B.Kalaavathi and Dr.K.Duraiswamy, Implementation of Ant Colony based Hybrid Multi-path Routing Algorithm for Virtual Class Room using Mobile Ad Hoc Network, GESTS International Transactions on Computer Science and Engineering, Volume 43, Number 1,2007, pp:174-191.

5. Ducatelle F., G. Di Caro and L.M. Gambardella, 2006. An analysis of the different components of the *Anthocnet* routing algorithm. Proceedings of the 50th International Workshop on Ant Colony Optimization and Swarm Intelligence (ANTS 2006), August 25, 2006 Brussels, Belgium, pp: 37-48. doi: 10.1007/11839088
6. Anand Prabhu Subramanian, Janani Vasudevan P. Narayanasamy and A.J. Anto 2004. Multipath Power Sensitive Routing Protocol for Mobile Ad Hoc Networks. Lecture Notes in Computer Science, Springer, Wireless On-Demand Network Systems (WONS 2004), pp: 84-89.
7. Paul Barom Jeon and George Kesidis, 2005. Pheromone-aided robust multipath and multipriority routing in wireless MANETs. Proceedings of the 2nd ACM International Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor and Ubiquitous Networks, 2005. pp: 106-113. doi: http://doi.acm.org/10.1145/1089803.1089974
8. Jin-Hee Cho, 2004. M.S. Report. Design, Implementation and Analysis of Wireless Ad Hoc Messenger, Virginia Polytechnic Institute and State University.
9. Anna Trifonova, 2003. Mobile Learning-Review of the Literature, Technical Report, University of Tento, Italy.
10. Georgios Parissidis, Vincent Lenders, Martin May, and Bernhard Plattener,. Multi-path Routing Protocols in Wireless Mobile Ad Hoc Networks: A Quantitative Comparison. Proceedings of 6th International Conference on Next Generation Teletraffic and Wired/Wireless Advanced Networking NEW2AN 2006, St.Petersburg, Russia, pp: 313-326.
11. Aristotelis Tsirigos and Zygmunt J. Haas, 2004. Analysis of multipath routing-part I: The effect on the packet delivery ratio. IEEE Trans. Wireless Commun., 3: 138-146.
12. Lee, S.-J. Gerla, M. Split Multipath Routing with Maximally Disjoint paths in Ad Hoc Networks. Proceedings of the International Conference on Communications 2001, volume 3, pp: 867-871.
13. Marina, M.K., Das, S. R. On-Demand Multipath Distance Vector Routing in Ad Hoc Networks. Proceedings of IEEE International Conference on Network Protocols (ICNP), 2001. pp: 14-23
14. Ye, Z., S.V. Krishnamurthy, S.K. Tripathi, 2003. A framework for reliable routing in mobile ad hoc networks. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE, 2003. 30 March-3 April 2003 IEEE Computer Society, Washington, DC., USA. pp: 270-280.