A Transparent Cache-based Mechanism for Mobile Ad Hoc Networks

Ying-Hong Wang[1], Jenhui Chen[2], Chih-Feng Chao[1]*, and Chien-Min Lee[1]
[1]*Dept. of Computer Science and Information Engineering, Tamkang University, Taiwan*
[2]*Dept. of Computer Science and Information Engineering, Chang Gung University, Taiwan*
inhon@mail.tku.edu.tw;jhchen@mail.cgu.edu.tw;cfchao@giga.net.tw;693191545@s93.tku.edu.tw*

Abstract

Mobile ad hoc networks (MANETs) enable all mobile nodes to communicate each other without base stations or access points, and the transfer of data packets is completed through the relay among all mobile nodes. However, a MANET is a self-organizing and adaptive wireless network formed by the dynamic gathering of mobile nodes, and the topology of a MANET frequently changes. To cope with the intrinsic properties of MANETs, a transparent cache-based mechanism is proposed in this paper. With the aid of the mechanism, the repetition of data and data path occurring in a MANET could be cached in some special mobile nodes. Routes and time span to access data are therefore shortened, and the data reusable rate is enhanced to reduce the use of bandwidth and the power consumption of battery.

Index Terms—Ad hoc networks, cache, mobile computing, transparent caching, wireless networks.

1. Introduction

Unlike infrastructured wireless networks, mobile ad hoc networks(MANETs) provide less stable communication for MNs to send and to receive signals; however, MANETs possess the characteristic of dynamic topologies, in which a MANET is a self-organizing and adaptive wireless network formed by the dynamic gathering of MNs, and the topology of a MANET frequently changes due to the mobility of MNs. As a result, MANETs could be applied to more sites, especially to special environments, such as battlefields, remote districts, rescue scenes and temporary assemblies.

The dynamic properties of MANETs are therefore challenging to routing protocol design [1, 2, 3]. To cope with the intrinsic properties of MANETs, we proposed a backup node mechanism, *Dynamic Backup Routes Routing Protocol (DBR^{2}P)*, for quick reconnection during link failures in [4, 5]. DBR^{2}P is proposed to focus attention on the intrinsic properties of MANETs, and the results of the simulation experiment show that DBR^{2}P has good performance.

Researches on routing protocols in MANETs have always been an important topic for discussion. Most researches, however, focus on the discovery of the shortest path or the highest bandwidth; issues related to the data transmission in MANETs are rarely considered [6, 7, 8]. Therefore, a suitable mechanism to cooperate with routing protocols is indispensable in order to solve the problems of data transmission and to enhance the entire efficiency in MANETs [9, 10].

A transparent cache-based mechanism integrated with dynamic backup routes routing protocol for MANETs is presented in this paper. With the aid of cache-based mechanism and DBR^{2}P, the repetition of data and data path occurring in a MANET could be cached in some special MNs. Routes and time span to access data are therefore shortened, and the data reusable rate is enhanced to reduce the use of bandwidth and the power consumption of battery.

In the sections that follow the concept of DBR^{2}P is briefly introduced. Section 3 describes the characteristics of temporal locality and spatial locality in MANETs. Section 4 illustrates the operations of the transparent cache-based mechanism integrated with DBR^{2}P. Finally, Section 5 concludes with suggestions of current challenges and potential directions for future research.

2. Dynamic Backup Routes Routing Protocol

DBR^{2}P [4, 5] is an on-demand routing protocol [1] that requires no routing table. It replies a complete route from the source node to the destination node on demand and sets up some backup routes dynamically for quick reconnection when a link fails. DBR^{2}P

Proceedings of the Third International Conference on Information Technology and Applications (ICITA'05)
0-7695-2316-1/05 $20.00 © 2005 IEEE

allows intermediate nodes to receive and transmit the same request packets obtained from the source node to gather more information to establish backup nodes. The basic packets, their functions and main fields defined for DBR^{2}P are illustrated in Table I.

TABLE I. THE MAIN PROTOCOL PACKETS OF DBR^{2}P.

Packet Name	Function	Main Fields
Route Discovery Request Packet (***RD-request***)	To find and record the route content from the source node to the destination node.	Sequence Number, Source Node Address, Destination Node Address, Route Content.
Route Discovery Reply Packet (***RD-reply***)	The destination node replies the route content back to the source node.	Sequence Number, Source Node Address, Destination Node Address, Route Content.
Backup Route Setup Packet (***BR-setup***)	The destination node transmits the backup information to the backup nodes to set up backup routes.	Sequence Number, Backup node Address, Backup Route Content.
Link_Fail_ Message Packet ***(LF-message)***	When a link fails, this packet is used to announce the backup nodes along the route to replace a backup route.	Sequence Number, Route Content.

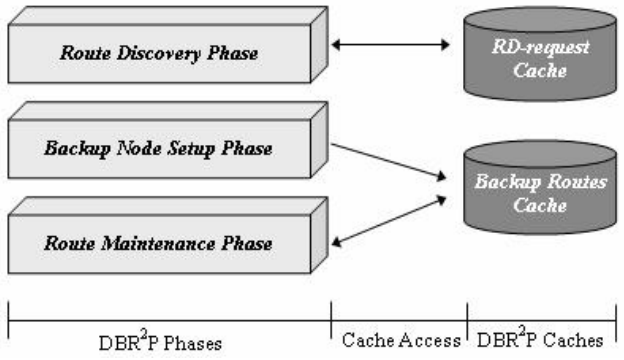

Figure 1. The main architecture of DBR^{2}P.

DBR^{2}P includes three phases requiring two kinds of cache as shown in Figure 1. A source node sets a unique request identification number for each RD-request packet from a locally-maintained sequence number. The RD-request_Cache of a node is used to store temporary counters, which record how many times this node receives the RD-request packets with the same identification number, in the route discovery phase. The entity, *<#RD, n>*, is used to present this counter parameter, where *#RD* is the sequence number of the RD-request, and n is the number of times that RD-request with the same *#RD* has been received. The Backup_Routes_Cache is used to store backup routes. After the route discovery phase is initiated by a source node, the destination node may receive some routes, and then enter the backup node setup phase in order to analyze some backup nodes and backup routes. In backup node setup phase, the backup routes are sent to each backup node by BR-setup packets and stored in the Backup_Routes_Cache of each backup node.

2.1 Route Discovery Phase

When source node S requires a route to destination node *D*, *S* enters the route discovery phase to find a route to the destination node. The phase of route discovery is illustrated in Figure 2. In this phase, source node S broadcasts RD-request to nearby nodes. The RD-request is used to discover some routes to the destination node. The RD-request includes a sequence number field to distinguish the route discovery process from the others, and a route content field to record all the addresses of nodes along the path from *S* to *D*. Initially, the address of the source node is inserted in the route content field of RD-request.

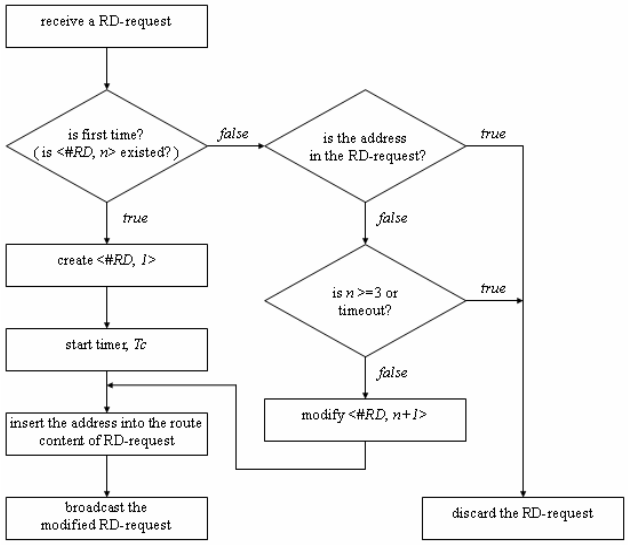

Figure 2. Route discovery phase of DBR^{2}P.

A source node sets a unique request identification number for each RD-request packet from a locally-maintained sequence number. When a node receives a RD-request from its neighbor, it will check whether the RD-request is received for the first time according to the sequence number of the RD-request in the records of its RD-request_Cache. There is no *<#RD, n>* entity in its RD-request_Cache if this node receives the RD-request from a node for the first time. Then, the entity, *<#RD, 1>*, is stored in the RD-request_Cache of the node, where *#RD* is the sequence number of the RD-request, and the value, 1, means this node receives the RD-request for the first time. Also, the timer, *Tc*, is started. Then, this node inserts its address into the route content field of the RD-request, and broadcasts this modified RD-request to its neighboring nodes.

If a node receives the RD-request with duplicate sequence number from its neighboring nodes, then there is an entity, *<#RD, n>*, in its RD-request_Cache,

where the *#RD* is equal to the sequence number of this RD-request. This node continues to check whether the route content field of RD-request includes its address; if so, the node discards this RD-request to avoid the infinite loop. On the other hand, if the address of this node is not included in the route content field of the RD-request, the node will then check whether the value of n is not smaller than three, or whether the timer, *Tc*, is timeout; if so, the node will discard this RD-request. Otherwise, the node increases the value n of <*#RD, n*> by one, and then inserts its address into the route content field of RD-request and broadcasts this modified RD-request to its neighboring nodes.

The parameter n in the <*#RD, n*> entity, stored in the RD-request_Cache of a node, is used to prevent too many upstream or downstream paths crossing this node as this node is an intermediate node of the route or backup routes. If there are too many transmissible paths of the route and backup routes crossing a node, a lot of backup routes may be invalid when this node disconnects or moves out. In addition, *Tc* begins when a node first receives RD-request from an upstream node, and an over-long route can be avoided if a node only receives duplicate RD-request within the period *Tc*. Moreover, the control message overhead, such as RD-request packets, can be bounded.

2.2 Backup Node Setup Phase

After the route discovery phase, the destination node *D* may gather many routes within a period, *Tc*. The nodes (excluding S and D) among those routes from S to D are intermediate nodes. Backup nodes are nodes with at least two different paths to their neighboring nodes in those routes from S to D. Thus, S is itself a possible backup node. The nodes in those routes that *D* has received are compared pair wise (from beginning to end) to find whether any two paths have a section in common. A final node (excluding *D*) in such a section is a backup node. A subset of backup nodes can be gathered from any two routes. Then, all the subsets of backup nodes are joined and the BR-setup that includes each backup node and the partial paths from the backup node to *D* is generated. Then *D* uses the BR-setup to set up the Backup_Route_Cache of those backup nodes separately. The BR-setup contains the sequence number of this routing process, the address of a backup node, and the path from the backup node to the destination node. Backup nodes store partial paths from the backup node to the destination node in their Backup_Route_Cache after they receive the BR-setup.

2.3 Route Maintenance Phase

In DBR^{2}P, the mechanism of passive acknowledgement [7] is used to detect a link failure. When a link failure is detected, a node in the route from the source to the destination can no longer transmit the data packet. This node will pass a "Link_Fail_Message packet (LF-message)" to an upstream node until the message reaches a backup node. After the backup node receives the LF-message, the backup route of Backup_Route_Cache is fetched to replace the route behind the backup node, and the source node S is informed to change the route. Then, S sends packets along the new route. A backup route that has been fetched by the Backup_Route_Cache is labeled as a non-backup route. If Backup_Route_Cache includes no other backup route, then the node loses the qualification to be a backup node. The source node will re-enter the Route Discovery Phase to establish a new route to the destination node when no available backup node exists.

After the destination node replies a path back to the source node as the current route for sending data packet, some backup routes are established and stored in the backup nodes. Sometimes the backup routes may be incorrect if the destination node receives inconsistent routes due to the loss of RD-request in the route discovery phase or the movement of the nodes along the backup routes in the route maintenance phase. If the current route is still alive, the situation that backup routes are incorrect will not influence the communication of the current route. If the current route is broken and replaced by a backup route, DBR^{2}P can still operate even though the backup route is broken again. That is because the link failure will be detected and a LF-message will be sent to find another backup node.

3. Temporal Locality and Spatial Locality in MANETs

- Spatial Locality characteristic: An accessed block exhibits spatial locality if blocks near it are likely to be accessed in the near future [11, 12].
- Temporal Locality characteristic: An accessed block exhibits temporal locality if it is likely to be accessed again in the near future [11, 12].

Generally speaking, the applications of MANETs are mostly in temporary occasions, such as battlefields, rescue scenes, assemblies, etc. That is to say the MNs in a certain MANET often gather temporarily for certain purpose. Therefore, comparing to other wired or wireless networks, the MNs in a MANET have more distinct temporal locality and spatial locality characteristics in data access.

Taken Figure 3 as an example, MNs in Group-1 (node A,B,C,D,E,F) and Group-2 (node G,H,I,J,K,L,M)

belong to the same MANET in an assembly. Assume the MNs in Group-1 and Group-2 belong to different communities, the MNs in Group-1 and Group-2 may have respective temporal locality and spatial locality characteristics in data access.

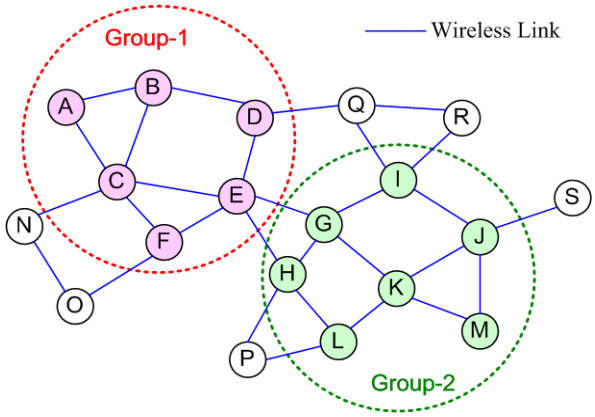

Figure 3. Temporal locality and spatial locality in a MANET.

3.1 Data Caching Scheme

Due to the aforementioned distinct temporal locality and spatial locality characteristics of data access in MANETs, data caching scheme could be adopted to help reduce the use of bandwidth and the power consumption of battery.

As shown in Figure 4, after Node D requests a data item (marked as d_i) from Node S, Node S will send d_i to Node D through the route i (marked as r_i : $S \rightarrow B \rightarrow F \rightarrow J \rightarrow N \rightarrow D$) set up by DBR^{2}P. During the process of data transfer in data caching scheme, all nodes on the data transfer routes cache d_i. Thus, when other nodes, those not on the r_i, such as E, K, etc., want to request d_i, they do not need to rebuild a route connecting Node S to transfer d_i. They (Node E or K) could connect to the nearest neighbor node on r_i to request d_i. For example, node E could connect to node F or node K could connect to node J to request d_i.

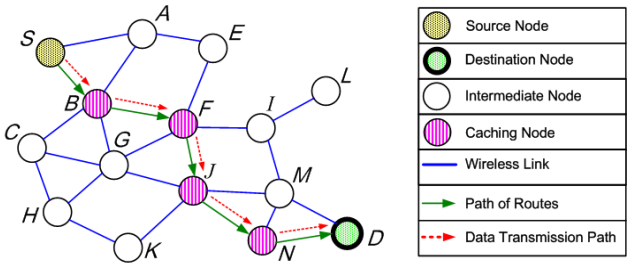

Figure 4. An example of data transmission path in a MANET.

Although using data caching scheme could shorten routes and time span to access data and raise data reusable rate to reduce the use of bandwidth, it will also cause the mass production of d_i clones in the MANET and result in additional waste of resources.

3.2 Data-Path Caching Scheme

Data-path caching scheme is also a kind of effective scheme to reduce data request delays without causing a large number of data clones.

As shown in Figure 4, after Node D requests a data item (d_i) from Node S, Node S will send d_i to Node D through the route i (r_i : $S \rightarrow B \rightarrow F \rightarrow J \rightarrow N \rightarrow D$) set up by DBR^{2}P. The data-path caching scheme enables all nodes to cache data-path of d_i (meanwhile data-path cache buffer will record that node S and D have d_i) during the process of data transfer. When other nodes (such as B, E, or K), except for node S and node D, also want to request d_i, they could detect which node has d_i by inquiring the cache data-path buffer of the nodes on the path. Then a route connecting node S or node D is set up to request d_i.

Through the use of data-path caching scheme, bandwidth and power are reduced while caching the data-path for each data because nodes can obtain data by using few hops. However, it will increase routing overhead to map data and cache nodes.

4. The Transparent Cache-Based Mechanism Integrated with DBR^{2}P

A transparent cache-based mechanism integrated with dynamic backup routes routing protocol for MANETs is presented in this section. With the aid of the mechanism, the repetition of data and data path occurring in a MANET could be cached in some special MNs. Routes and time span to access data are therefore shortened and the data reusable rate is enhanced to reduce the use of bandwidth.

4.1 Cache Sharing Interface

Each MN, as illustrated in Figure 5, is installed with a Cache Sharing Interface (CSI), which includes four major components — Cache Manager (CM), Handoff Manager (HM), Data Cache Buffer (DCB), and Data-Path Cache Buffer (DPCB).

CM is mainly responsible for putting the data that pass through the MN into DCB, or putting the data path into DPCB. Data path indicates which MNs contain the data. If the space of DCB or DPCB overflows, CM will adopt the Least Recently Used algorithm (LRU algorithm) to replace data or data path. The major job of HM is to transfer data seamlessly to a new backup node. When a link failure is detected,

DBR^{2}P will initiate a backup route. Meanwhile, the MN, which is also an on-the-route backup node, has to take the action of handoff, and the data cached in the MN's DCB will be transferred seamlessly to the new backup node.

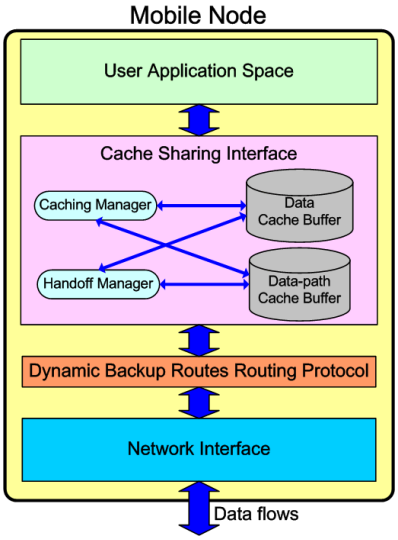

Figure 5. A mobile node is installed with a caching sharing interface.

4.2 Compound Caching Scheme

As discussed in Section 3, simply using data caching scheme or data-path caching scheme is definitely insufficient. Therefore, in our proposed transparent cache-based mechanism, a compound caching scheme is introduced to solve the defects of aforementioned data caching and data-path caching schemes.

Assume a route path set up by DBR^{2}P as shown in Figure 6. The major procedure of compound caching scheme is as follows:

1) Set the backup node which is nearest to destination node (node D) as the cache node of node D (assume the cache node is node C).
2) The cache node (node C) will cache the data item (d_i) transmitted from node S to node D.
3) Set the $BOUNDRY_{SC}$ at 1/2 hops on the routing path between source node (node S) and cache node (node C). Nodes closer to node S cache data-path of d_i (meanwhile the DPCB of nodes H, I will record that node S has d_i) while nodes closer to node C cache data-path of d_i (meanwhile the DPCB of nodes J, K will record that node C has d_i).
4) Set the $BOUNDRY_{CD}$ at 1/2 hops on the routing path between cache node (C) and destination node (node D). Nodes closer to C cache data-path of d_i (meanwhile the DPCB of node L will record that node C has d_i) while nodes closer to node D cache data-path d_i (meanwhile the DPCB of nodes M, N will record node D has d_i).
5) If 1/2 hops(S to C) or 1/2 hops(C to D) is odd, the procedure will set boundaries closer to cache node (node C), as illustrated in Figure 6. $BOUNDRY_{CD}$ will be set between node L and node M; hence, the effective cache range of three nodes — node S, node C, and node D — will be more equal.

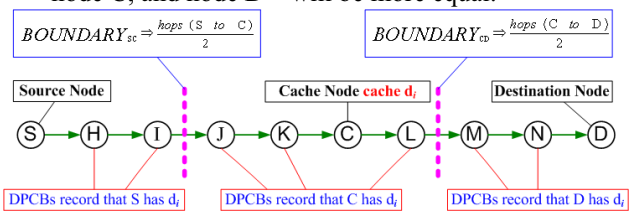

Figure 6. The cache boundaries when using compound caching sheme.

4.3 Handoff Processes

Refer to Figure 7(a), after node D requests a data item (d_i) from node S, node S will transmit d_i to node D following the route ($S \rightarrow B \rightarrow F \rightarrow J \rightarrow M \rightarrow D$) set up by DBR^{2}P. During the operation of compound caching scheme, the cache node of node D will be node J. Refer to Figure 7(b), if the link between $J \rightarrow M$ fails, d_i will be transmitted through the route ($S \rightarrow B \rightarrow F \rightarrow J \rightarrow N \rightarrow D$) under the workings of DBR^{2}P. If the link between $J \rightarrow M$ also fails, assume that d_i will be transmitted through another route ($S \rightarrow B \rightarrow F \rightarrow I \rightarrow M \rightarrow D$). In the meantime, the incomplete d_i cached in DCB of node J will be transferred seamlessly to a new take-over cache node (backup node F).

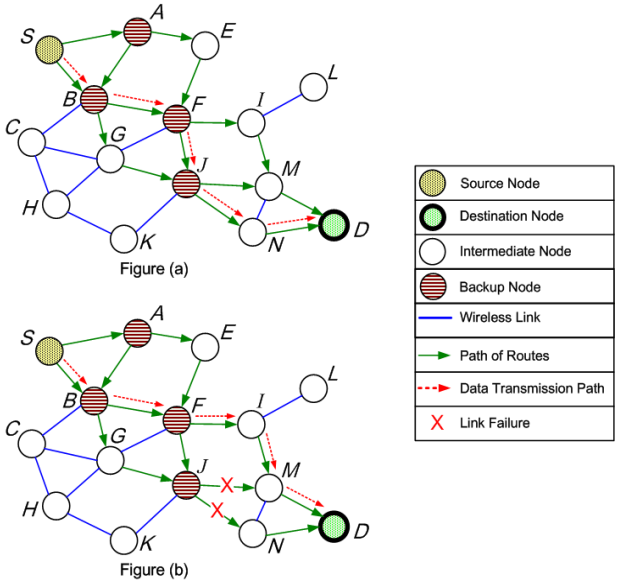

Figure 7. The handoff processes after link failures occur.

In the above-mentioned situations, in order to avoid the waste of bandwidth, HM will only transmit the incomplete d_i to the new cache node (backup node F) instead of transferring the entire data cached in DCB. Besides, to maintain cache consistency, while cache node starts the process of handoff, HM needs to inform the nodes, in which DPCB records node J has d_i, to modify their records. The original record that node J has d_i will therefore be updated to that the new take over cache node (backup node F) has d_i, and the original records related to node J in DPCB will be deleted entirely.

4.4 Transparent Cache-Based Mechanism Interchangeability

Although the transparent cache-based mechanism and its relevant schemes are proposed to operate in coordination with dynamic backup routes routing protocol, it is also possible to apply the mechanism to different routing protocols. If part of the procedure of the transparent cache-based mechanism is modified in accordance with the diverse characteristics of MANET routing protocols, it is believed that the mechanism could also have good performance in environments using different routing protocols and therefore reach the goal of the transparent cache-based mechanism interchangeability.

5. Conclusion and Future Work

Dynamic backup routes routing protocol (DBR^2P) could provide more reliable and more stable data transmission in the wireless network applications, such as the mobile learning, mobile commerce and mobile entertainment [13, 14, 15]. To cope with the intrinsic properties of mobile ad hoc networks (MANETs), a transparent cache-based mechanism integrated with DBR^2P for MANETs is proposed in this paper. This mechanism could help to shorten the routes to access data and enhance the data reusable rate to cut down the use of bandwidth and the battery power consumption.

The project on the detailed operation procedure of the proposed mechanism is proceeding and a simulation environment will be set up for performance test. Issues such as the quality of service and multicast could be addressed to improve the capability of the proposed mechanism. Moreover, supporting hierarchical and heterogeneous interfaces in MANETs could be considered in future research. Hopefully in the near future the proposed mechanism will be applied to real MANETs environments with the assistance of relevant simulation results.

6. References

[1] S. Corson and J. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," *IETF RFC2501*, Jan. 1999.

[2] E. M. Royer and C. K. Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks," *IEEE Personal Communications*, Vol. 6, No. 2, Apr. 1999, pp. 46-55.

[3] S.J. Lee, M. Gerla and C. K. Toh, "A simulation study of table-driven and on-demand routing protocols for mobile ad hoc networks," *IEEE Network*, Vol.13 , Issue.4 , July-Aug. 1999, pp.48-54.

[4] Y. H. Wang, C. M. Chung, C. P. Hsu, and C. C. Chuang, "Ad Hoc On-Demand Routing Protocol Setup with Backup Routes," *in Proceedings of International Conference on Information Technology: Research and Education (ITRE)*, Aug. 10-13, 2003, pp.137-141.

[5] Y.H. Wang, and C.F. Chao, "Dynamic Backup Routes Routing Protocol for Mobile Ad Hoc Networks," accepted to be published in the *Information Sciences: An International Journal*, Elsevier Science.

[6] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *ACM SIGCOMM '94 Computer Communications Review* 24(4), Aug. 1994, pp. 234-44.

[7] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad-Hoc Wireless Networks," in *Mobile Computing*, edited by T. Imielinski and H. Korth, Eds., Kluwer, 1996, pp. 153-81.

[8] C. E. Perkins and E. M. Royer, "Ad-hoc On-Demand Distance Vector Routing," in *Proceedings of 2nd IEEE Workshop Mobile Comp. Sys. and Apps.,* Feb. 1999, pp. 90-100.

[9] G. Cao,"Proactive Power-Aware Cache Management for Mobile Computing Systems," *IEEE Trans. Computers*, Vol.5, no.6, 2002, pp. 608-621.

[10] S. Lim, W.C. Lee, G. Cao and C.R. Das,"A Novel Caching Scheme for Internet Based Mobile Ad Hoc Networks," *Proc. IEEE Int'l Conf. Cmputer Comm. and Networks ((ICCCN)*, 2003, pp.38-43.

[11] P. J. Denning and S. C. Schwartz,"Properties of the Working-set Model," *Communications of the ACM*, 15(3), Mar. 1972.

[12] A. J. Smith, "Cache Memories," *ACM Computing Surveys* 14(3), Sep. 1982, pp. 473-530.

[13] T. Plagemann, V. Goebel, C. Griwodz, and P. Halvorsen, "Towards Middleware Services for Mobile Ad-hoc Network Applications," in *Proceedings of the Ninth IEEE Workshop on Future Trends of Distributed Computing Systems*, 2003, pp. 249-55.

[14] M. Milenkovic, S. H. Robinson, R. C. Knauerhase, D. Barkai, S. Garg, V. Tewari, T. A. Anderson, and M. Bowman, "Toward Internet Distributed Computing," *Computer*, Vol. 36, No. 5, May 2003, pp. 38-46.

[15] A. Fuller,P. McFarlane,D. Saffioti, "Distributed, collaborative learning environments using ad hoc networks," *Proc. IEEE Int'l Conf. Advanced Learning Technologies*, Aug.-Sept. 2004, pp.705-707.