Volume 2, Issue 2, February 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Caching Strategies For On-Demand Routing Protocols In Ad-hoc Networks

[1]**Karthikeyan.U**
Research Scholar ,Dept of CSE
Bharath University

[2]**Dr.Rajani**
Research Director,
Bharath University

Abstract- On-demand routing protocols are widely used in mobile ad hoc networks due to their capability of adjusting to frequent network topology changes within acceptable routing overhead. In order to further reduce routing overhead, especially the overhead from the network-wide coding in the route discovery phase, two techniques named route caching and searching localization are usually performed. In this paper, we reinvestigate these two techniques, in particular their joint effect on the routing overhead. For quantitative analysis purposes, we define one essential parameter for each technique: route caching validation probability and local searching radius. Based on the analytic results, we propose a new routing strategy that adapts to the current caching availability and is self-tunable towards the optimal performance. We demonstrate through extensive simulations that this routing strategy can reduce the routing overhead greatly under general scenarios.

1. Introduction

In ad hoc networks, there is no pre-existing led net-work architecture. Mobile nodes, typically with similar transmission and computational capabilities, cooperate by forwarding packets for nodes that are not in each other's direct transmission range. The properties of ad hoc networks such as node mobility, limited available bandwidth and the broadcast nature of the wireless medium make the design of efficient routing protocols for ad hoc networks more challenging than for traditional networks.

Routing protocols proposed for ad hoc networks can be roughly divided into two categories: table-driven (proactive) and on-demand (reactive). On-demand protocols typically have lower routing overhead than table-driven proto-cols, and thus attract more interest. This is because they only initiate a route discovery process when a packet is ready to be transmitted, which removes the necessity of persistent maintenance of a routing table using costly shortest path algorithms. Typical examples of these reactive proto-cols are DSR [1], AODV [2]. However, despite the reduced routing overhead using this reactive approach, the performance is still not satisfactory. Two primary techniques are introduced to improve the performance: route caching and searching localization.

Route caching plays an important role in reducing both the overhead and the discovery latency. After storing a route cache from the route discovery phase, a node is able to send a new packet without delay and respond to route requests from other nodes without further broadcasting. However, the routing overhead cannot be effectively reduced by those intermediate nodes due to the coding nature of route re-quests. Also, a stale cache may bring about even more routing overhead and even longer packet delay.

Caching optimizations have been extensively researched to fully exploit been ts from caches. However, in this pa-per, we point out that in order to obtain full been ts, a local searching scheme must be performed in cooperation with the caching scheme. Also, a more accurate method for quantifying the quality of caches is required in order to avoid the adverse effects from stale caches. The major contribution of our paper is to provide a method to accurately measure the quality of caches and determine the optimal local search radius according to current caching conditions (see Section 3). Based on the analytical results, we present our local-searching and caching strategy that can adaptively adjust itself to the caching availability conditions and approach the optimal performance in Section 4. Finally, we implement this strategy in DSR and demonstrate its advantage through extensive simulations.

2. Overview and related work

When a node has a packet to send but has no route to the destination node, it initiates a route discovery procedure by broadcasting a Route REQuest (RREQ) packet. Upon receiving a RREQ, if an intermediate node has a cached route for the destination or itself is the destination, it uncast a Route Response (RRES) following the reversed route back to the source node. The discovered route will be stored in a route cache by the source node for future use. Intermediate nodes without route caches for the target attach their addresses in the RREQ packet and continue to flood the packet.

Caching strategies and caching designs for DSR are studied in [3]. The authors achieved the optimal choices for timeout and route cache capacity through exhaustive searching over specie scenarios. In our paper, we not only study the effects of caching, but also the joint effects of searching localization. Some optimizations, such as *salvaging*, *gratuitous route repair* and *promiscuous listening*, have been proposed and have been shown to be effective in reducing stale caches and improving the performance of route caching [1]. Some other proactive optimizations, such as *Negative caches*, *Wider error notification* and *Adaptive timeout selection*, are proposed in [4]. These schemes, al-though not taken into account in our analysis and simulations, can cooperate with our routing strategies seamlessly. Their existence only changes the caching availability conditions in the network, while our routing strategy is able to adjust itself based on the caching conditions and achieve the optimal performance.

Compared to the extensive studies on route caching, study in the searching localization area is relatively lacking. Although LAR [5] is able to localize its querying area, it requires geographical information, which we do not assume in our study. In DSR, the *non-propagating route re-quest technique* is performed by the source node to search one-hop neighbor's rest before resorting to a network-wide hood. In AODV [2], an expansion ring searching scheme is proposed. A source node starts a route discovery process with an initial searching radius and increases the searching radius linearly upon each failure. A network-wide search is performed when the searching radius exceeds a preened threshold. However, in [6], it is shown that when the existence of caching availability in the network is weak (e.g., in networks with infrequent traffic), using one-hop lo-cal searching has an only insignia cant savings in overhead, while the expansion ring scheme has more overhead than a simple network-wide coding. Another searching localization technique is also proposed in [7]. It utilizes prior routing histories but does not take route caches into account. Our scheme also utilizes prior route histories, but in a different manner, and our paper concentrates on the joint effects of the route caching and the local searching techniques rather than only one of these. Also, in contrast to the experimental study on cache validation and optimization schemes in [8], our study exposes the underlying relationship between routing overhead and caching optimization methods quantitatively.

The authors in [9] studied the effects of DSR with both caching and local searching, and they mentioned the possible ineffectiveness of the expansion ring technique under weak caching existence. They compared the performance of one specie expansion ring scheme with the one-hop local searching scheme and analyzed when a node should switch from one scheme to the other. In our paper, we do not consider the expansion ring scheme. Instead, we analytically determine the optimal local searching radius among all the possible choices in different scenarios and propose our protocol to realize it. To the best of our knowledge, this is the rest study on ending the optimal performance of on-demand routing protocols for general scenarios with both techniques applied.

3 Model and analysis

3.1 Nomenclature and assumptions

Without loss of generality, let us consider *N* homogenous nodes with unit transmission range that are randomly distributed in a disk of radius *R*. *N* is large enough to form a network with good connectivity [10, 11]. Nodes move with the maximum speed of S_m in a random waypoint method. Each node has a total event rate of $_T$. In this paper, *event* indicates a one-way traffic flow towards a destination that is randomly selected from all the other nodes. The arrival of the events is a random variable that follows a poison distribution, and each event lasts for a fixed lifetime T_l . During this lifetime, it is not necessary for the traffic to be continuous. For example, for an event with a lifetime of 10 seconds, there may be only 10 packets, or one packet per second. We denote Src as the source node and D as the destination node.

During our analysis, we assume that we are studying the DSR protocol with only the options of *gratuitous route re-pair* and *non-propagation route request* turned on. Without *gratuitous route repair*, after a RERR is received, the source node will receive invalid caches from intermediate nodes each time it resends the RREQ. The loop of RREQ-invalid cache-RERR will continue until the cache in the intermediate nodes expires. The performance of the protocol without this option is too poor to be studied. *Non-propagation route request* is the same as our local searching technique and will be fully studied. Furthermore, we assume that each node has at most one route cache entry for each destination. To avoid reply storms, we also assume that the destination only replies to the rest route query packet.

In the remainder of this section, we will rest introduce two crucial protocol parameters. Then we reveal the relationship between routing overhead and these two parameters. Finally, we give the optimal numerical results for these two parameters to maximize the overhead reduction.

3.2 Local searching radius

A Local searching scheme must work with a caching scheme to be meaningful. In [6], it is shown that when there is no caching, even the optimal local searching scheme can reduce the searching overhead only by an amount of at most 8% while bringing about excessive latency.

In general, a local search has two-sided effects on the searching overhead. If a local search nods the target itself or it nods cached routes to the target in intermediate nodes, the network-wide hood can be avoided if the route cache is correct, and the searching overhead is reduced effectively. However, if this search fails to return any result, a network-wide search is still required and the overall searching cost is even more than a simple network-wide hood. Thus, the lo-cal searching radius k should be carefully chosen to achieve the most benefit from the local search. If the radius k is chosen too large, the probability of discovering a route returned from the destination itself is large and little benefit can be obtained from the route caches. If the radius k is chosen too small, the chance of discovering the destination or a cached route to the destination in this local search is also small and little benefit can be gained from this local search because the first round local search will be part of the total searching overhead. As we will show later, the radius k is a crucial parameter in determining the RREQ overhead and is also closely related to the amount of caching in the network.

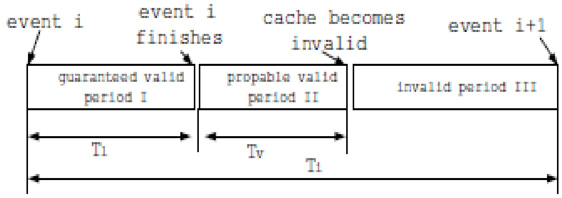

Figure 1. Time periods for a node's route caching between two events towards the same destination. In Period I, the cache is almost valid all the time. In the probable valid period II, the node has stopped sending traffic to the destination and therefore has only a probable valid route cache. In the invalid period III, the route cache is of very low quality

3. Life periods of a cache

To understand how a node responds to other nodes' route requests, we need to clarify the life periods of a route cache rest. The time interval between two traffic events from a certain Src to a certain D can be divided into three caching periods, as shown in Fig. 1, which are the guaranteed valid period I, the probable valid period II and the invalid period III. In different periods, a route cache is of different qualities and has different effects on the routing overhead and the routing performance.

Starting from the leftmost of, when a new event, say event i, just arrives, Src initiates a route discovery process and caches the discovered route for future use. During period I, all of the traffic packets of this event will follow this cached route. Meanwhile, if the route is broken due to node mobility, the route maintenance will detect it and initiate a new route discovery. Thus, during the event lifetime T_l , this node maintains a valid route for D. More strictly speaking, during period I, a node can respond to a RREQ from other nodes with a route cache whose validation probability P_v is very close to 1.

During life period II when there is no more traffic between Src and D, there are no more proactive methods such as route maintenance to refresh the cached route.

3.2 Protocol description

Only minor changes are needed for existing on-demand routing protocols to fully attain the beets of caches. Two primary parameters are needed, the caching validation probability threshold p_t and the local searching radius k. When the network is just set up, or a node just joins a network, these values should be set to p_t = 0:4 and $k = b \frac{M}{2} c$, assuming weak or moderate caching conditions. When more abundant caching conditions are detected based on history, k should be set to a smaller value accordingly

3.3 New data structures

A Newel for caching validation probability is required for both the RREQ and the RRES packets. For RREQ packets, the value of this led is calculated through the parameter adjustment rules described below and appended in the RREQ packets to serve as the caching validation threshold. For RRES packets, the value of this led is calculated by the node that initiates the RRES packet to indicate the cache's quality using equation 1.

Also, each node maintains a statistic such as the number of recent RREQ attempts, the values of k and p_t applied, the number of guaranteed valid caches and the number of probable valid caches received. This information is used to estimate the current caching condition to serve for the parameter adjustment rules. The counting of these numbers does not differentiate destinations and only needs a little extra storage. This non-differential counting is valid for uniform traffic scenarios. For biased traffic scenarios such as the client-server traffic model, maintenance of the history of different destinations may provide more accurate parameter adjustment. The tradeoff is much larger extra storage for each destination node. In our current work, we utilize the general statistical method without destination differentiation.

4. Protocol procedure

When a source node needs to send a RREQ, it calculates the parameters k and p_t according to the parameter adjustment rules and attaches the values in the RREQ packet. Intermediate nodes calculate P_v for their cached route to the destination from equation 1 and return a RRES packet with P_v attached if P_v satisfies $P_v > p_t$. The source node picks the cached route with the largest P_v . When two cached routes have close P_v values, the one with a shorter route length is preferred. Each node refreshes the statistics each time it sends out a RREQ packet and receives RRES packets from intermediate nodes.

4.1 Parameter adjustment rules

The parameter adjustment rules determine the value of p_t according to the current caching situation. A node first calculates the average number of guaranteed valid route caches N_g and the average number of probable valid route caches

Performance evaluation

4.2 Assumptions, metrics and methodology

We simulate our routing scheme as an add-on to the DSR protocol in a mobile ad hoc network scenario. The simulations are performed using ns-2 [12]. In order to focus our study in the routing level, we programmed an ideal lower

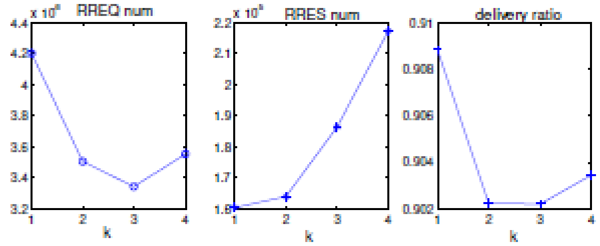

Figure 2. Performance of DSR-LC with pt _xed at 0.4. The X-axis indicates the local search- ing radius k, ranging from 1 to 4. The optimal point is at k = 3 for the number of RREQ with almost no effect on other metrics

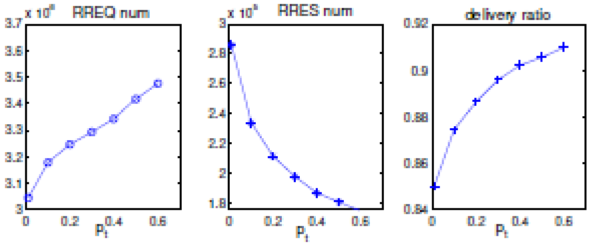

Fig. 3, the increase of TIMEOUT causes both metrics to decrease. A good balance point is at TIMEOUT=10 seconds.

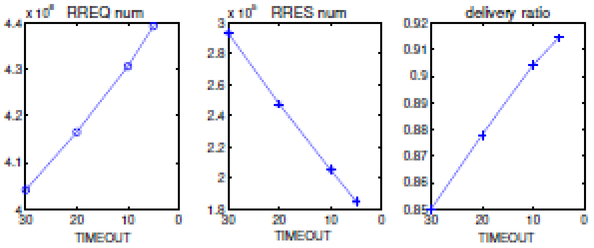

Figure 4. Performance of DSR-C with one-hop neighbor searching. The X-axis indicates the timeout value, ranging from 30 seconds to 5 seconds. The tradeoff is also between the RREQ number and the delivery ratio. Just like

5. Conclusions and future work

The main contributions of this paper are to determine the optimal value for parameters of local searching radius and route caches valid probability threshold and to propose an adaptive routing strategy that can be easily added on to existing on-demand protocols. The new scheme adjusts the local searching radius and the required caching quality according to the caching conditions. It reduces routing overhead consistently and significantly, for both RREQ and RRES packets, with almost no effects on other performance aspects.

Further research on the parameter adjustment and the performance of our routing scheme working with specific MAC protocols is needed. One avenue of future work is to take into account more history information to aid in determining the local searching radius. Another avenue is to investigate how our scheme performs in a more realistic model. Currently, we ignore the existence of a MAC layer and the physical layer. In our future work, we will examine the performance improvement of our scheme over the popular ad hoc MAC protocol 802.11 and realistic wireless channels. However, as explained earlier in section 5, we believe that the impact of realistic MAC layers and wireless channels should be negligible.

6. References

[1] D.B.Johnson and D.A.Maltz. *Mobile Computing*, Chapter Dynamic source routing in ad hoc wireless networks, pages 153-181. Kluwer Academic Publishers,Imielinski and Korth edition, 1996.

[2] C.Perkins and E.M.Royer. .Ad hoc on-demand distance vector routing. *Proceedings of IEEE WMCSA' 99*, pp. 90.100, Feb. 1999.

[3] Y.-C. Hu and D. B. Johnson, .Caching strategies in on-demand routing protocols for wireless networks,. *Proc. ACM/IEEE MobiCom*, August 2000.

[4] M. K. Marina and S. R. Das, .Performance of Route Caching Strategies in Dynamic Source Routing, . *Proc. of WNMC in conjunction with ICDCS*,pp. 425.432, 2001.

[5] Y.-B. Ko and N. H. Vaidya, .Location-aided routing (LAR) in mobile ad hoc networks,.*In ACM/IEEE Int. Conf. MobiCom'98*, Oct. 1998

[6] Z. Cheng, W. Heinzelman,.Flooding strategy for target discovery in wireless networks,. *Proc. of the 8th MSWIM 2003*, Sep 2003.

[7] R. Castaneda and S. Das, .Query localization techniques for on-demand routing protocols in ad hoc networks, . *in Proc. ACM/IEEE MOBICOM '99*, 1999, pp. 186.194.

[8] N. Panchal and N. B. Abu-Ghazaleh, .Active Route Cache Optimization for On-Demand Ad Hoc Routing Protocols., *ICN 2004*

[9] J. Sucec and I. Marsic, .An Application of Parameter Estimation to Route Discovery by On-Demand Routing Protocols., *Proc. ICDCS 2001*, pp. 207.218.

[10] Feng Xue and P. R. Kumar. .The number of neighbors needed for connectivity of wireless networks. Manuscript, 2002. Available from http://black1.csl.uiuc.edu/prkumar/postscript _les.html

[11] B. Krishnamachari, S. Wicker, R. Bejar, and M. Pearlman, .Critical Density Thresholds in Distributed Wireless Networks,. *in Communications, Information and Network Security, Kluwer Publishers*, 2002.

[12] http://www.isi.edu/nsnam/ns/ns-documentation