Adaptive MANET Routing for Low Overhead

Yangcheng Huang
University College London
Dept of Computer Science
London WC1E 6BT, UK
yangchenghuang@ieee.org

Saleem Bhatti
University of St. Andrews
School of Computer Science
Fife KY16 9SX, UK
saleem@cs.st-andrews.ac.uk

Søren-Aksel Sørensen
University College London
Dept of Computer Science
London WC1E 6BT, UK
s.sorensen@cs.ucl.ac.uk

Abstract

In wireless mission-critical systems, systems may be resource-constrained including limited bandwidth, so minimising protocol overhead, whilst maintaining performance, is important. Proactive MANET routing protocols tend to provide smaller route discovery latency than on-demand protocols because they maintain route information to all the nodes in the network at all time. However, such protocols may impose excessive soft-state routing control overhead which is generated by disseminating periodic update messages. In order to mitigate the side effects of the soft-state control overheads, we propose two adaptive proactive routing algorithms, namely DT_MIAD *and* DT_ODPU*. By tuning the value of refresh intervals dynamically and automatically, refresh updates are triggered based on traffic conditions and node mobility. We show through simulations that the proposed adaptive routing algorithms can outperform a traditional proactive routing protocol (OLSR).*

1 Introduction

Mobile ad hoc networks (MANETs) are characterised with frequent topology changes and resource constraints (such as battery life and bandwidth). Typical MANET mission-critical applications, including emergency rescue operations, and battlefield communications, exhibit high degrees of connection dynamics due to mobility and complex natural effects (thunderstorms etc). Consequently, a fundamental challenge in ad hoc networks is the design of routing protocols that can respond quickly to network conditions whilst still maintaining low protocol overhead for operation in resource-constrained environments.

Proactive protocols like OLSR [7], TBRPF [3] and DSDV [11] tend to provide smaller route discovery latency than on-demand protocols like AODV [10] and DSR [9] because they maintain route information to all the nodes in the network at all time. Frequent soft-state routing updates maintain accurate topology information, by the disseminating of periodic HELLO messages and topology control (TC) messages in soft-state maintenance. However, such frequent updates may impose excessive routing control overhead, which could contribute to channel congestion and degrade network performance. So, proactive routing algorithms have a fundamental trade-off between performance and routing overhead.

There have been several adaptive routing approaches for MANETs [4] [12] [5]. Benzaid et al [4] presented an approach to adjust refresh frequency based on node mobility and the status of neighbouring nodes. Ramasubramanian et al [12] proposed a zone-based hybrid routing algorithm which combined proactive and reactive strategies. Boppana et al [5] proposed an adaptive Distance Vector routing algorithm by adopting flexible route update strategies according to conditions. These adaptive approaches have the following potential drawback.

Dependency on network measurement. The routing performance of the approaches in [4] and [12] largely depend on the accuracy of network measurement. It is an open question on how to get accurate estimation sof real-time network/traffic characteristics in practice.

Increased complexity. For example, in [12], the operations in zone maintenance and continuous network monitoring not only introduce extra processing overhead but also increase the complexity in configuration and implementation. The performance of ADV [5] is determined by *constant trigger thresholds*, which need to be manually configured.

Unknown performance bounds. For example, in ADV [5], the route update frequency increases quickly with node mobility, which brings larger overheads than periodic updates. Also, since only partial route information is maintained, ADV takes longer for a new connection to find a valid route.

This paper proposes two adaptive proactive routing algorithms, namely DT_MIAD and DT_ODPU. By tuning the value of soft-state refresh interval timers *dynamically* and *automatically*, the refresh updates are triggered based on network load and mobility conditions. We have shown with simulations that the proposed adaptive routing algorithm outperforms OLSR [7], and has the following benefits.

First, the operations of the proposed algorithms are independent of network measurement and node mobility detection. Based on analytical studies on link change rate, we propose a simple method in detecting node mobility.

Second, the proposed algorithms are simple in both configuration and implementation. The adaptability process is totally automated with only a few parameters. Using feedback control theory, the proposed algorithm can be implemented incrementally, with no need to make significant changes to the existing protocols.

The rest of the paper is organised as follows. Section 2 gives some background information on traditional proactive routing algorithms. Section 3 gives the detailed description of the routing algorithms. Section 4 introduces the simulation configurations used in this study. Section 5 presents our observations based on the NS2 simulations. Conclusions are summarised in section 6.

2 Traditional Proactive Routing Protocols for MANETs

In this section, we present an overview of the traditional proactive routing algorithms including Link State algorithm such as OLSR and Distance Vector algorithm such as DSDV.

In Link State (LS) protocols like OLSR [7], each node discovers and maintains a complete and consistent view of the network topology, by which each node computes a shortest path tree with itself as the root (i.e. *shortest path first (SPF)* algorithm), and applies the results to build its forwarding table. This assures that packets are forwarded along the shortest paths to their destinations. LS protocols rely on periodic refresh messages to reflect topology changes and maintain correct topology information. Each node sends HELLO messages periodically to discover new neighbors and detect link failures. LS protocols in MANETs advocate periodic topology update to avoid the large volumes of of topology update messages triggered by frequent topology change events.

Table 1. DT_MIAD **Notation**

h_0	Initial HELLO interval of node i
$link\_chg\_cnt$	Change rate over current refresh period
$prev\_chg\_cnt$	Change rate over previous refresh period
$prev2\_chg\_cnt$	Change rate over the period before previous
β	Additive decrease rate
α	Multiplicative increase rate
h_{max}	Upper limit of refresh interval
h_{min}	Lower limit of refresh interval

In Distance Vector(DV) protocols like DSDV [11], each node maintains a routing table containing the distance from itself to all other nodes in the network. Each node broadcasts periodically its routing table to each of its neighbours and uses similar routing tables from neighbouring nodes to update its table. The route selection is based on Distributed Bellman-Ford(DBF) algorithm. To keep up with network changes, DV protocols use both periodic and triggered updates.

The main problem of traditional proactive routing (especially LS) lies in the use of fixed timer intervals. The refresh intervals are configured by administrators, usually with the default values recommended by protocol designers. High mobility demands small intervals to speed-up topology change detection, while low mobility only needs relatively large intervals. Due to the non-uniform distribution of node mobility, both temporally and spatially, fixed timer intervals fail to be effective when node mobility is high, but may be inefficient when node mobility is low. Thus, the refresh intervals need to be adapted to network conditions.

3 Adaptive Proactive Routing Protocol

In this study, we improve periodic update strategies of existing proactive routing protocols by adapting dynamically refresh rates. In the following paragraphs, we present the details of our proposed algorithms, namely DT_MIAD (Dynamic Timer Based on Multi-Increase Additive Decrease) and DT_ODPU (Dynamic Timer Based on On-Demand Proactive Update).

3.1 Dynamic Timer Based on Multi-Increase Additive Decrease

The dynamic timer algorithm based on Multi-Increase Additive Decrease (MIAD) is inspired by control-theoretic adaptive mechanisms similar to those widely adopted in the Internet, i.e. Additive Increase Multiplicative Decrease (AIMD) of TCP, which is used to adjust sending rates in response to network congestion. Our approach in this algorithm uses a Multiplicative-Increase Additive-Decrease (MIAD) controller to adapt the soft-state refresh rate r to the conditions of node mobility and data traffic.

Briefly, the refresh rate r is multiplied by factor α ($\alpha > 1$) if node mobility or data packet drop rate increases, otherwise it is decremented by factor β. By aggressively increasing r in presence of increased of packet loss rate and network link change rate, the routing algorithm improves link failure detection, which reduces packet loss and increases link availability. Whenever the link change rate decreases, the routing algorithm lowers the refresh rate conservatively until it reaches a steady state.

The key question is, what is the quantitative relationship between node mobility and the link change rate? If it is linear, the node mobility can be simply detected by monitoring the link change rate. We clarify this issue in the following paragraphs and present the details of the proposed algorithm.

Any change in the set of links of a node may be either due to the set-up of a new link or to the loss of an active link. Thus, the expected link change rate for a node, ψ, is equal to the sum of the expected new link arrival rate, η, and the expected link breakage rate, ξ.

Samar and Wicker studied the theoretical quantitative relationship between link change rate ψ and factors including node velocity in [13]. They found that, in a practical ad hoc or sensor network where "the number of neighbors of a node is bounded", the expected rate of link breakages ξ is equal to the expected rate of new link arrivals η. Therefore, the expected link change rate for a node, ψ, is twice the expected new link arrival rate, η.

Equation (1) describes the expected new link arrival rate [13]:

$$\begin{aligned}\dot{\eta}(v) \quad = \quad & \frac{2R\delta}{\pi b}[\frac{v^2}{4}\int_0^{\pi} p(\phi)\log(\frac{b+\sqrt{b^2-v^2\sin^2\phi}}{v+v\cos\phi})d\phi \\ & +b^2\varepsilon(\frac{v}{b})] \qquad (1)\end{aligned}$$

Here, ε is the standard Complete Elliptic Integral of the Second Kind; ϕ is the direction of motion (i.e. the degree of the angle with x axis); $p(\phi)$ equals $1 + 3cos(2\phi)$; R is the transmission range; σ is the average density of nodes within a transmission zone; b is the maximum velocity.

Consider the impacts of node velocity, v, on link change rate ψ, i.e. $\frac{d\psi}{dv}$ the derivative of ψ with respect to v. We obtain:

$$\dot{\psi}'_t > 0 \qquad (2)$$

$$\dot{\psi}''_t > 0 \qquad (3)$$

From Equations (2) and (3), with the increase of node velocity, the expected link change rate increases. Moreover, the increasing speed of the expected link change rate increases with the node velocity. Therefore, we can examine the dynamics of link change rate in order to detect any changes of node mobility.

The pseudocode of the proposed algorithm is as shown in Algorithm 1. We use the notation as shown in Table 1.

Algorithm 1 *DT_MIAD*

```
Input: $h_0 < \frac{1}{\beta}$
  $h \leftarrow h_0$
  $link\_chg\_cnt \leftarrow 0$
  $prev\_chg\_cnt \leftarrow 0$
  $prev2\_chg\_cnt \leftarrow 0$
  rest_of_init()
  loop
    Proporgate_Refresh_Msg()
    if $link\_chg\_cnt > prev\_chg\_cnt$ then
      if $link\_chg\_cnt - prev\_chg\_cnt > prev\_chg\_cnt - prev2\_chg\_cnt$ then
        $h \leftarrow \frac{h}{\alpha}$
        if $h < h_{min}$ then
          $h \leftarrow h_{min}$
        end if
      end if
    end if
    $h \leftarrow \frac{h}{1-h*\beta}$
    if $h > h_{max}$ then
      $h \leftarrow h_{max}$
    end if
    SynchroniseTimerInterval()
    $prev2\_chg\_cnt \leftarrow prev\_chg\_cnt$
    $prev\_chg\_cnt \leftarrow link\_chg\_cnt$
    $link\_chg\_cnt \leftarrow 0$
    DELAY($h$)
    /* ... do something else ... */
  end loop
```

3.2 Dynamic Timer Based on On-Demand Proactive Update

Dynamic Timer Based on On-Demand Proactive Update (*DT_ODPU*) is based on the concept of the status of a node which is in one of two states: *dynamic* and *static*. When internal link changes are detected

Algorithm 2 *DT_ODPU*

```
Input: $0 < h_{min} < h_{max}$
  $h \leftarrow h_{min}$
  $prev\_refresh\_time \leftarrow now$
  $link\_chg\_cnt \leftarrow 0$
  rest_of_init()
  loop
    if $link\_chg\_cnt > 0$ then
      Proporgate_Refresh_Msg()
    else if $now \geq (prev\_refresh\_time + h_{max})$ then
      Proporgate_Refresh_Msg()
      $prev\_refresh\_time \leftarrow now$
    end if
    $link\_chg\_cnt \leftarrow 0$
    DELAY($h$)
    /* ... do something else ...*/
  end loop
```

Table 2. MAC/PHY Layer Configurations

MAC Protocol	IEEE 802.11
Radio Propagation Type	TwoRayGround
Interface Queue Type	DropTailPriQueu
Antenna Model	OmniAntenna
Radio Radius	250m
Channel Capacity	2Mbits
Interface Queue Length	50

(link_chg_cnt > 0), the node is in *dynamic* state; correspondingly, it uses a smaller refresh interval h_{min}. Otherwise, the node is still and uses a larger refresh interval h_{max}. In this algorithm, the soft-state update is still *proactive* since refresh messages are still exchanged periodically. However, the refresh *frequency* (or refresh *interval*) dynamically is adjusted in a *reactive* manner. The pseudocode of the proposed algorithm is as shown in Algorithm 2.

4 Performance Analysis

4.1 Simulation Set-up

We integrate our proposed algorithms with the OLSR implementation which runs in version 2.9 of NS2 [1] and uses the ad-hoc networking extensions provided by CMU [2]. The detailed configuration is shown in Table 2.

We use a network consisting of n nodes: $n = 20$ to simulate a low-density network, $n = 50$ to simulate a high-density network. Nodes are placed in a $1000 \times 1000\ m^2$ field. All simulations run for 100s.

We use the Random Trip Mobility Model, "a generic mobility model that generalizes random waypoint and random walk to realistic scenarios" [6] and performs perfect initialisation. Unlike other random mobility models, Random Trip reaches a steady-state distribution without a long transient phase and there is no need to discard initial sets of observations. Manhattan Mobility Model is also used under different scenarios.

The mean node speed, v, ranges between 1m/s to 30m/s. For example, when the mean node speed is 20m/s the individual node speeds are uniformly distributed between 0m/s and 40m/s. The average node pause time is set to 5s.

A randomly distributed CBR (Constant Bit Rate) traffic model is used which allows every node in the network to be a potential traffic source and destination. The rate of each CBR traffic is 10kb/s. The CBR packet size is fixed at 512 bytes. There are at least $n/2$ data flows that cover almost every node.

For each sample point presented, 100 random mobility scenarios are generated. The simulation results are thereafter statistically presented with the mean of the metrics and the errors. This reduces the chance that the observations are dominated by a certain scenario which favours one protocol over another.

4.2 Performance Metrics

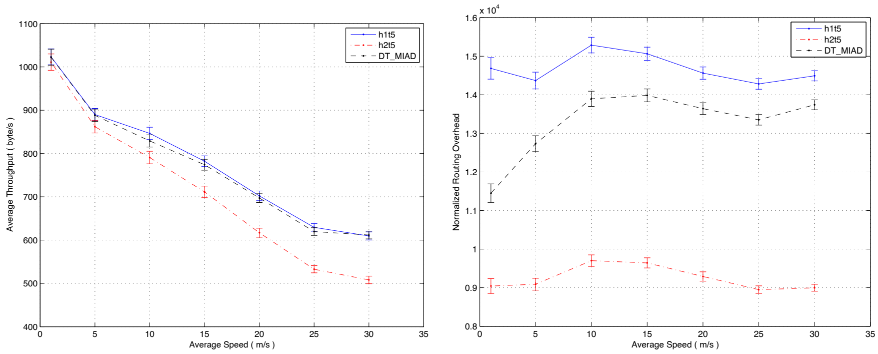
In each simulation, we measure each CBR flow's throughput and control traffic overhead and then calculate the mean performance of each metric as the result of the simulation.

Throughput is considered as the most straightforward metric for the MANET routing protocols [8]. It is computed as the amount of data transferred (in bytes) divided by the simulated data transfer time (the time interval from sending the first CBR packet to receiving the last CBR packet).

The control overhead consists of HELLO messages and TC messages. Considering the broadcast nature of the control message delivery, the packets are counted by summing up the size of all the control packets *received* by each node during the whole simulation period.

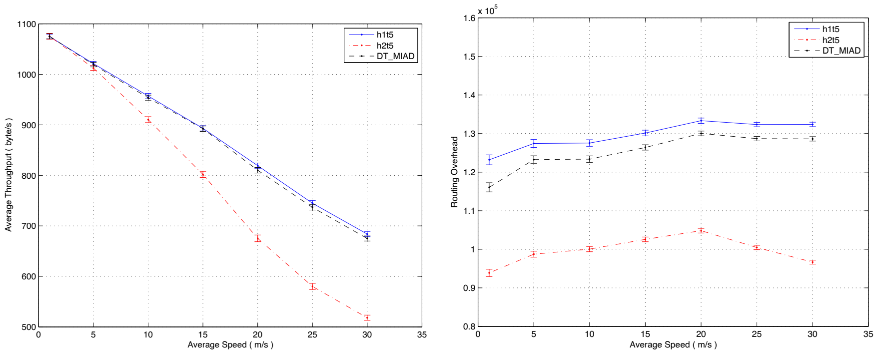
5 Observations

In this section, we compare the routing performance of the proposed adaptive routing algorithms with that of a standard proactive routing protocol, and present the observations under the variation of various parameters, such as node velocity and node density.



(a) Throughput (n=20)

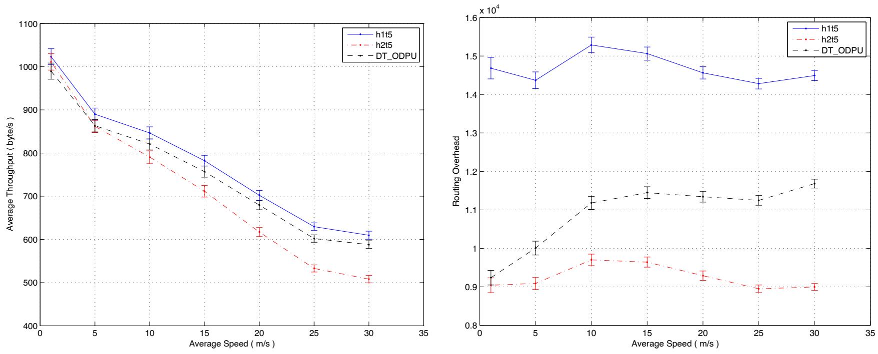
(b) Overhead (n=20)



(c) Throughput (n=50)

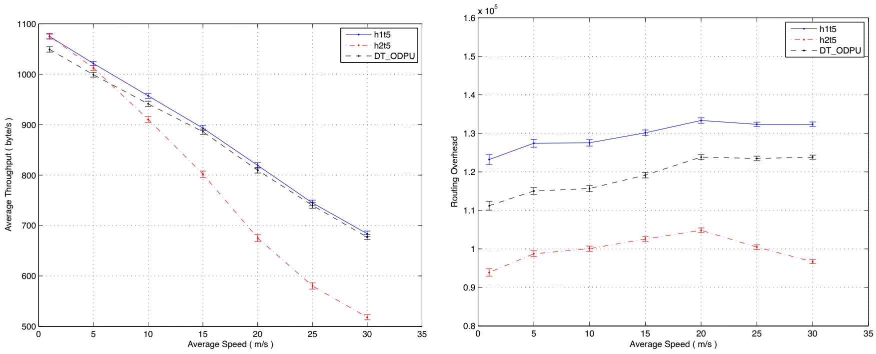
(d) Overhead (n=50)

Figure 1. Performance of DT_MIAD



(a) Throughput (n=20)

(b) Overhead (n=20)



(c) Throughput (n=50)

(d) Overhead (n=50)

Figure 2. Performance of DT_ODPU

5.1 DT_MIAD Performance

As shown in Fig 1, OLSR with *DT_MIAD* performs as well as standard OLSR configured with smaller refresh interval ($h = 1s$) but with much less overhead.

Further performance comparisons with standard OLSR configured with a larger interval ($h = 2s$), OLSR with *DT_MIAD*, shows good adaptability to node mobility. That is, with the increase of node mobility, the performance drop of OLSR with *DT_MIAD* is less significant. For example, as shown in Fig 1(c), when the node velocity increases from 10m/s to 20m/s, OLSR with *DT_MIAD* has 14.6% performance drop, while standard OLSR ($h = 2s$) has up to 32.6%. On the other hand, as shown in Fig 1(b), the overhead of OLSR with *DT_MIAD* is up to 22.5% less than that of standard OLSR configured with a small refresh interval. The overhead of OLSR with *DT_MIAD* increases as the nodes move faster.

To summarise, the simulation results show that *DT_MIAD* outperforms the standard OLSR proactive routing algorithm in terms of the balance of throughput and overhead.

5.2 DT_ODPU Performance

From Fig 2(a) and Fig 1(a), OLSR with *DT_ODPU* performs slightly worse than OLSR with *DT_MIAD*, since in some cases the throughput of OLSR with *DT_ODPU* is significantly lower than standard OLSR with smaller refresh intervals. However, in terms of control overhead, as shown in Fig 2(b) and Fig 2(d), OLSR with *DT_ODPU* shows better adaptability to node mobility. For example, when node velocity is relatively low, the control overhead introduced by OLSR with *DT_ODPU* is as low as that by standard OLSR configured with larger refresh intervals ($h = 2s$). The overhead increases with node mobility, which indicates that the refresh intervals are being tuned in response to the changing network conditions.

To summarise, compared with the standard proactive OLSR routing algorithm, *DT_ODPU* significantly improves the routing performance, while introducing lower control overhead than trying to improve throughput simply by reducing the configured refresh interval.

6 Conclusions

In this study, we present an adaptive scheme for proactive routing protocols and propose two adaptive routing algorithm enhnacements, namely *DT_MIAD* and *DT_ODPU*. We evaluate the performance of these two enhancements through extensive ns2 simulations, modifying OLSR, over a wide range of network scenarios. The results show that the proposed dynamic timer algorithms have better adaptability and routing performance than standard proactive OLSR routing algorithm.

The original data, the source code and the scripts used in this study are all available from the authors' websites (`http://www.cs.ucl.ac.uk/staff/y.huang/dt/`).

References

[1] Ns2 website. http://www.isi.edu/nsnam/ns/.

[2] The rice monarch project: Wireless and mobility extensions to ns-2. http://www.monarch.cs.rice.edu/cmu-ns.html.

[3] B. Bellur and R. G. Ogier. A reliable, efficient topology broadcast protocol for dynamic networks. In *IEEE INFOCOM*, pages 178–186, mar 1999.

[4] M. Benzaid, P. Minet, and K. Agha. Integrating fast mobility in the olsr routing protocol, 2002.

[5] R. V. Boppana and S. Konduru. An adaptive distance vector routing algorithm for mobile, ad hoc networks. In *IEEE INFOCOM*, pages 1753–1762, 2001.

[6] J. Y. L. Boudec and M. Vojnovic. Perfect simulation and station-arity of a class of mobility models. In *IEEE INFOCOM*, Miami, USA, 2005.

[7] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, a. Qayyum, and L. Viennot. Optimized link state routing protocol. In *IEEE INMIC Pakistan*, 2001. Best paper award.

[8] S. Corson and J. Macker. Mobile ad hoc networking (manet): Routing protocol performance issues and evauluation considerations. Request for Comments 2501, IETF, January 1999.

[9] D. B. Johnson and D. A. Maltz. The dynamic source routing protocol for mobile ad hoc networks. Internet draft (draft-ietf-manet-dsr-06.txt), November 2001.

[10] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc on-demand distance vector (aodv) routing. Request for Comments 3561, IETF, July 2003.

[11] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *SIGCOMM*, pages 234–244, New York, NY, USA, 1994. ACM Press.

[12] V. Ramasubramanian, Z. J. Haas, and E. G. Sirer. Sharp: A hybrid adaptive routing protocol for mobile ad hoc networks. In *MobiHoc*, Anapolis, MD, 2003.

[13] P. Samar and S. B. Wicker. On the behavior of communication links of a node in a multi-hop mobile environment. In *MobiHoc*, pages 145–156, New York, NY, USA, 2004. ACM Press.