# A Comparison of Temporal and Topological Soft State Updates for a Proactive MANET Routing Protocol

Yang Cheng Huang†, Saleem Bhatti‡, Søren-Aksel Sørensen†

†University College London, ‡University of St. Andrews

**Abstract:** In an mobile ad hoc networks (MANET), each node maintains routing information about other nodes in the network at all times. The nodes propagate state refresh messages to maintain correct routing information. In this paper we present a quantitative analysis on the impact of *temporal updates* and *topological updates* for a proactive MANET routing protocol and assess its impact on performace. Simulation results show that the temporal state updates have a significant impact on the throughput, while the topological state updates do not. Furthermore, frequent topological state updates in relatively high-density networks lead to performance degradation because of the large overhead introduced.

## 1 Introduction

Each node in a MANET regularly transmits state updates in order to enable other nodes to maintain correct routing state. As state information may time-out unless it is refreshed, it is called *soft state*. In particular, nodes using proactive routing protocols use two types of soft state updates: *temporal state* updates for localised neighbour detection and *topological state* updates for consistent connectivity knowledge of each node.

In this paper, we present a quantitative analysis on the performance impact of these different soft-state updates, on a proactive MANET routing protocol called OLSR (Optimised Link State Routing Protocol)[5] by adjusting the relevant state refresh control parameters (i.e. HELLO intervals and TC intervals, explained later).

The rest of the paper is organised as follows. Section 2 gives background information on the soft state mechanisms in OLSR; section 3 describes the simulation configurations used in this study; section 4 presents our observations based on simulations; conclusions and future work are summarized in section 5.

## 2 Optimized Link State Routing Protocols

The Optimised Link State Routing Protocol(OLSR)[5] uses proactive routing to maintain the routing information about each node in the OLSR network. OLSR nodes detect neighbour changes by exchanging periodical HELLO messages, and propogate topologyical control (TC) messages among all the nodes of the network to discover available routes in the presence of mobility and resource constraints. On the other hand, OLSR inherits the use of the link state algorithm, using shortest path first forwarding. Unlike other LS protocols, OLSR optimises flooding via multipoint relays (MPRs).

In this section, we introduce briefly the state management mechanisms in the Optimised Link State Routing Protocol.

### 2.1 Link State Repository

In order to maintain the topology information of the whole network in the presence of mobility and failure, an OLSR daemon needs to maintain two types of state repositories in its internal tables, namely a *local state repository* and a *topological state repository*.

The *local state repository* records and keeps updated the localized information, including link tuples, neighbour tuples and MPR tuples.

*Link tuples* in a link set keep track of the link status between the node and its neighbours. There are two types of status: *SYM* link (e.g. bi-directional) and *ASYM* link (e.g. uni-directional). Each link tuple contains the interface addresses of the local node and the neighbour node (i.e. the end points of a link), and the *valid time* during which the link is considered as available and useable.

The *neighbour set* contains neighbour tuples to keep track of a nodes neighbour status, including willingness and valid time etc., while the *2-hop neighbour set* records a set of 2-hop tuples that describe symmetric links between its neighbours and the symmetric 2-hop neighbourhood.

The *MPR set* maintains a set of neighbours that are selected as MPRs, while the *MPR selector set* records a set of MPR-selector tuples and describes the neighbours that have been selected by this node as MPRs.

The *topological state repository* (or *topology set* as defined in OLSR[5]) maintains the topology information of the whole network. This information is acquired from Topology Control (TC) messages and is used for routing table calculations.

## 2.2 State Update Mechanism

### 2.2.1 Temporal State Update

OLSR nodes update the local state repositories by exchanging HELLO messages; such a process is called *temporal state update*. The rate of the temporal state updates is the sending rate of HELLO messages. OLSR nodes generate periodic HELLO messages based on link status information recorded in link set. When receiving a HELLO message, OLSR nodes update the link set first; the other state repositories are then synchronised or re-calculated by examining the link tuples in the link set.

### 2.2.2 Topological State Update

OLSR nodes update their topological state repository via TC (Topology Control) messages. The rate of the topological state updates is the sending rate of TC messages.

In OLSR, only MPR nodes generate topology control messages based on the MPR selector set. Also, only MPR nodes are responsible for forwarding control traffic, intended for diffusion into the entire network. MPRs provide an efficient mechanism for flooding control traffic by reducing the number of message transmissions.

It is clear that the maintenance of the internal state information held at nodes is directly related to the exchange of HELLO and TC messages and so anything that affects when these messages are generated, such as refresh timer intervals, is likely to impact on protocol performance.

# 3 Simulations

The simulation study is conducted with the OLSR implementation from [3] which runs in version 2.9 of NS2 [1] and uses the ad-hoc networking extensions provided by CMU [2], with a radio range of 250m radius and the use of MAC/802_11 as the media access control.

We use a network consisting of n nodes: $n = 20$ to simulate a low-density network, $n = 50$ to simulate a high-density network. The nodes are randomly placed in an area of 1000m by 1000m.

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" [4] 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.

The mean node speed, $v$, ranges between 1m/s to 30m/s. For example, when the mean node speed is 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 random 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 CBR packet size is fixed at 512 bytes. There are at least $n/2$ data flows that cover almost every node.

In each simulation, we measure each CBR flows throughput and then calculate the mean performance of each metric as the result of the simulation. Throughput 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).

We vary the rate of HELLO messages, $h$, and the rate of TC messages, $t$ (both $h$ and $t$ have units of 'messages per second'). In order to gain good confidence in the measurement results, we run the simulations *100* times for each data point with a different mobility pattern file, i.e. a different starting state for the node positions.

## 4 Results

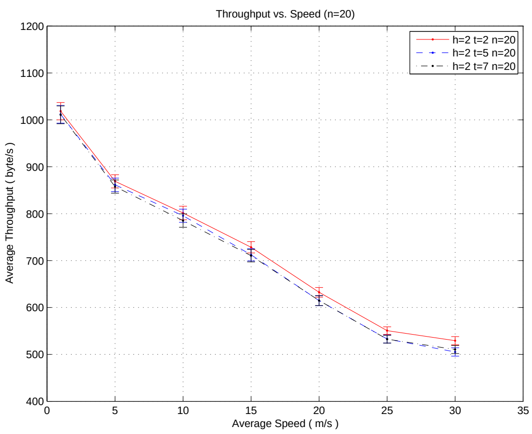


Figure 1: The Effects of Varying $t$ (n=20, h=2)

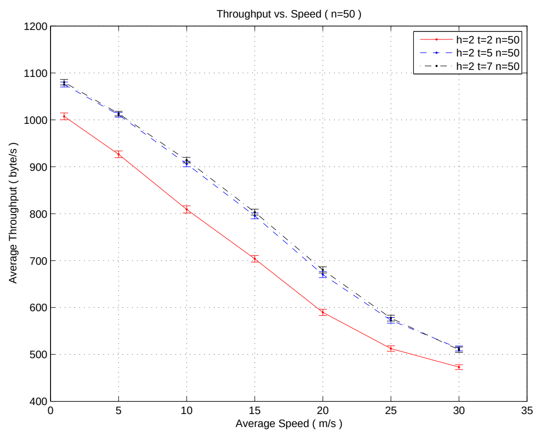


Figure 2: The Effects of Varying $t$ (n=50 h=2)

For Fig 1 and 2, we hold $h$ constant, and for Fig 3 and 4, we hold $t$ constant.

From Fig 1,2,3 and 4, we see that with the increase of node speed, the average throughput drops almost linearly. Further, by comparing Fig 1 and 3 with Fig 2 and 4 we can see that the linearity in high-density networks is better than in low-density networks.

From Fig 3 and 4, under various scenarios (i.e. node speed and node density), we see that increasing temporal state update rates could improve the average throughput. In addition, by comparing Fig 3 and 4 we can see that the improvements in high-density networks are more significant than in low-desity networks networks.

As seen in Fig 3, only when the average speed is relatively high ($v \geq 15$m/s) (no overlaps on the error bars) are the average throughputs under different temporal update rates are significantly different. On the other hand, from Fig 4, the significant difference of the average throughputs

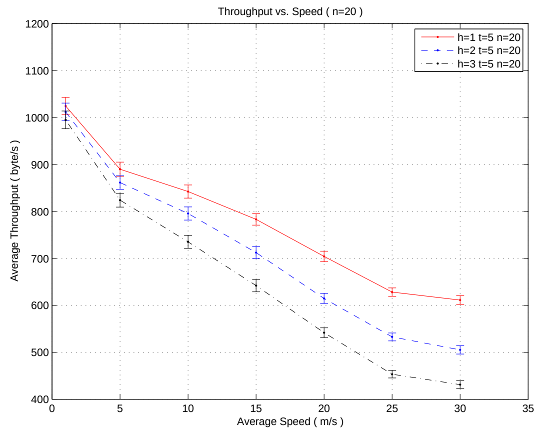


Figure 3: The Effects of Varying $h$ (n=20, t=5)

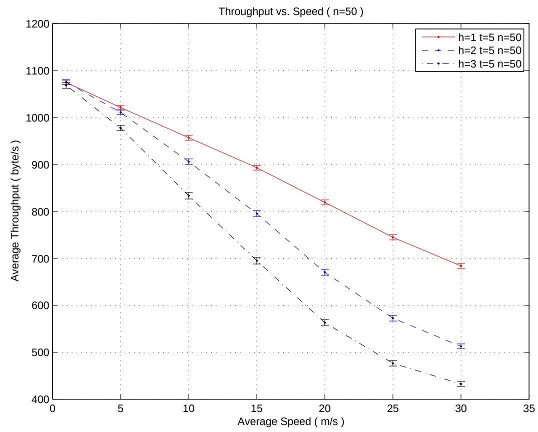


Figure 4: The Effects of Varying $h$ (n=50, t=5)

is observed when the average speed is relatively low (i.e. $v$ greater than approximately 7m/s).

Similar patterns are *not* observed in terms of topological update rates. From Fig 1, under various scenarios, increasing topological update rates brings no significant improvement to the throughput. The means of the average throughputs with the update interval $t = 2$ are only slightly better than those with longer update intervals. In addition, as seen in Fig 2, in high-density networks, increasing the topological update rates leads to significant performance degradation. This could be explained by the large amount of control traffic, which finally causes congestion and leads to packet drops.

## 5 Conclusions and Future Work

In this paper we present a quantitative analysis on the impacts of *temporal updates* and *topological updates* on the perfromance of a proactive MANET routing protocol. The simulation results show that the rate of temporal state updates (HELLO messages) have a significant impact on the throughput, whilst the rate of topological state updates (TC messages) seem to have a negligible impact. Furthermore, frequent topological state updates in relatively high-density networks leads to performance degradation because of the large overhead introduced.

We are currently working on a fast neighbour detection mechanism, which could enhance the temporal state update process and then improve the routing performance.

## 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] Um-olsr. http://masimum.dif.um.es/um-olsr/html/.

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

[5] T. Clausen and P. Jacquet. Optimized link state routing proto-col. RFC 3626, 2003.