AHR: A Two-State Adaptive Mechanism for Link Connectivity Maintenance in AODV

Carles Gomez
Technical University of Catalonia
Avda. Canal Olímpic, s/n
08860 Castelldefels (Spain)
+34 93 413 72 06
carlesgo@entel.upc.edu

Alex Cuevas
Technical University of Catalonia
Avda. Canal Olímpic, s/n
08860 Castelldefels (Spain)
+34 93 413 72 06
acuevas@entel.upc.edu

Josep Paradells
Technical University of Catalonia
C/ Jordi Girona, 1-3, C3
08034 Barcelona (Spain)
+34 93 401 6024
teljpa@entel.upc.edu

ABSTRACT
Real world MANET routing protocol implementations generally use layer three mechanisms for carrying out connectivity maintenance tasks. In many cases, such mechanisms are based on periodical transmission of Hello messages. When topology changes occur frequently, usage of higher Hello rates than default may lead to higher end-to-end connectivity and available bandwidth. However, in rather static conditions, low Hello rates are adequate to avoid unnecessary bandwidth and power consumption waste. In this paper we present Adaptive Hello Rate (AHR), a two-state adaptive mechanism for adjusting HELLO_INTERVAL parameter in AODV. We evaluate AHR by conducting a set of experiments in real ad-hoc testbeds. Results quantify the potential benefits and tradeoffs of the proposed solution.

Categories and Subject Descriptors
C.2.2 [**Network Protocols**]: Routing protocols.

General Terms
Algorithms, Measurement, Performance, Design, Theory.

Keywords
MANET, AODV, adaptive mechanism, Hello Interval, implementation.

1. INTRODUCTION
One of the features of a MANET routing protocol with significant influence on network performance is connectivity maintenance [1, 2, 3]. This is an important task which aims at monitoring the availability of links used in an on-going communication. Should any link fail, the routing protocol can attempt to repair the route or find an alternative one.

Most MANET routing protocol specifications contemplate a range of layer two and layer three link failure detection strategies. A majority of routing protocol implementations use by default a layer three based link failure detection mechanism. In prior work [3] we conducted a set of experiments in real ad-hoc testbeds using the Ad-hoc On-demand Distance Vector (AODV) routing protocol [4]. Results suggested that the HELLO_INTERVAL parameter should be adaptive, so that an appropriate value could be used taking into account network dynamics. For instance, using a higher Hello rate than default may be considered for faster reaction in case of topology changes in high mobility conditions. Otherwise, if the network is rather static, low Hello rates are suitable for avoiding an unnecessary waste of power and bandwidth resources.

In this paper we present Adaptive Hello Rate (AHR), a two-state adaptive mechanism for tracking link connectivity in AODV. We have evaluated AHR theoretically and implemented it on AODV-UU v.0.8.1 [5] for analysis in real environments. The remainder of the article is organized as follows. Section 2 describes the AHR mechanism operation and the rationale behind its design. Section 3 presents and discusses tests and results. The modified AODV implementation including the AHR code tested, as well as the scripts used in our trials, can be found at [6].

2. AHR MECHANISM

2.1 Design goals
We next enunciate the requirements that AHR mechanism should fulfil:

i) Providing higher Hello rates than default in a dynamic network to increase network connectivity. Such benefits should compensate bandwidth consumption of Hello messages themselves.

ii) Providing low Hello rates in a rather static network, to avoid unnecessary bandwidth and energy waste.

iii) HELLO_INTERVAL autoconfiguration criteria should be based on Time to Link Failure (TLF) values (i.e., link lifetimes) and Time Without link Changes (TWC) of a node.

iv) AHR mechanism should be lightweight. Implementation overhead should be minimized in terms of

REALMAN'06, May 26, 2006, Florence, Italy.
ACM 1-59593-360-3/06/0005.

computing resources. Moreover, consumption of additional bytes in AODV messages should be avoided.

2.2 Two-state algorithm

We have considered a two-state algorithm as follows:

Low dynamics state. A low Hello rate is used. This is the default state. HELLO_INTERVAL used in this state is equal to *HELLO_INTERVAL_LD*. If estimated TLF is smaller than a given threshold, denoted by *Thr1*, the system switches to the High dynamics state defined next.

High dynamics state. A high Hello rate is used. In this state, HELLO_INTERVAL is equal to *HELLO_INTERVAL_HD*. When measured TWC becomes greater than a threshold denoted by *Thr2*, the mechanism switches back to the Low dynamics state.

2.3 Time to link failure estimation

AHR mechanism operates using a TLF estimate as an input. Note that network dynamics may vary with time. In consequence, TLF estimation should ideally provide adapted link lifetime values in a fast way. The algorithm we use for TLF estimation is similar to that used by TCP [7] for RoundTrip Time (RTT) averaging. Hereafter we will refer to the estimated TLF at instant *k* as *sTLF(k)*. The measured TLF at instant *k* will be denoted by *TLF(k)*. Accordingly, *sTLF(k)* computation algorithm is:

$$sTLF(k) = a \cdot sTLF(k-1) + (1-a) \cdot TLF(k-1) \qquad (2)$$

where *a* is a tuning parameter. A new iteration of the algorithm is performed every time a link fails, updating the TLF estimate with the measured TLF for the last broken link. Note that this algorithm implicitly considers the average degree of a node (i.e., average number of neighbors), since a large average degree leads to high probability of at least one link break during a certain observation period.

2.4 Time without link changes measurement

TWC is obtained as the difference of times between the last link change in the routing table of a node and current time. This measurement is updated every time a Hello message is sent by the node. TWC depends inversely on the average degree of a node.

2.5 Dissemination of High dynamics state

AHR nodes operate in Low dynamics state by default. When fast topology changes arise, a node adapts to the new situation, triggered by small TLF averaged values, by switching to the High dynamics state. To fully exploit AHR potential benefits, nodes within a given radius in number of hops should be informed as well of the new situation. We have designed a local dissemination mechanism of High dynamics state. When a node switches to High dynamics state, it activates a State Switch (SS) flag in AODV RREP headers, taking advantage of one of the reserved bits in such headers. We use the RREP message since Hello messages are built on top of RREP messages in the AODV implementation we have considered (see next subsection). Note that no overhead is added to AODV messages. If a node in the Low dynamics state receives a Hello message with an activated SS flag, the node switches to High dynamics state in any case. We have designed this High dynamics state dissemination mechanism following a local approach. When low TLFs trigger a node to switch to High dynamics state, only one-hop neighbors of this node perform the same state change. One-hop neighbors do not activate the SS flag in their own Hello messages. In specially high dynamic environments it could be interesting to increase the SS dissemination radius. Studying this possibility remains as future work.

2.6 Parameter choices

We have set *HELLO_INTERVAL_HD* to 0.2 seconds and *HELLO_INTERVAL_LD* to 1 second (i.e., default HELLO_INTERVAL value). We believe such values represent a reasonable tradeoff among reactivity to topology changes, bandwidth consumption and power consumption in High and Low dynamics state, respectively. We have set *Thr1* = *Thr2* = 30 seconds, which is the average TLF with Random Waypoint and Manhattan mobility models at a node mean speed equal to 5 m/s [8]. We have set *sTLF(0)* to 30 seconds as well, as a neutral choice for initial conditions.

3. TEST EXPERIMENTS AND RESULTS

3.1 Available bandwidth measurements

We have conducted a number of trials for measuring the available bandwidth improvement provided by AHR mechanism in a scenario with four laptops, forming a two-hop, two-path topology. The radio interface is IEEE 802.11b. Mobility is emulated with a script using the `iptables` tool and the `sleep` command. Emulated link lifetimes are pseudorandomly obtained assuming an exponential distribution and mean values for a range of speeds as shown in [8]. A UDP flow is sent during 1000 seconds. The sending rate is equal to 6 Mbps, which is higher than the actual capacity of the two-hop path. We measure the bitrate at the receiver in each trial. We have experimented with usage of *a*=0 and *a*=0.5 for the TLF estimation algorithm.

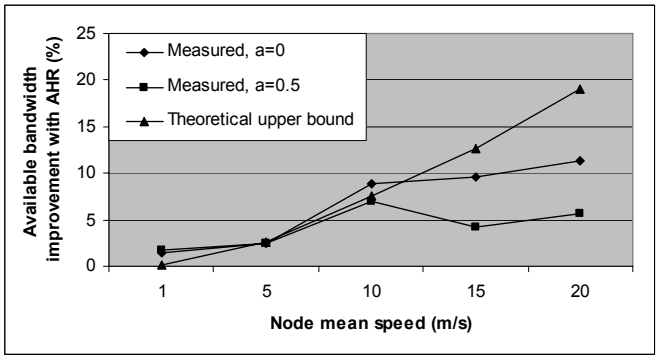

Figure 1. Available bandwidth improvement provided by AHR in comparison with default AODV

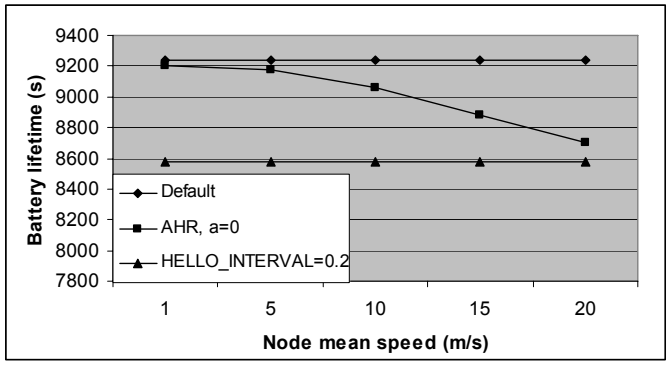

Figure 2. Power consumption measurements. Comparison among default AODV, usage of HELLO_INTERVAL=0.2 and AHR with *a*=0

Figure 1 shows the percentage improvement provided by AHR in comparison with default AODV in terms of available bandwidth. Next-hop layer three theoretical connectivity improvement is also plotted, labelled as "Theoretical upper bound". This value is an upper bound on the measured available bandwidth due to the following reasons: i) Hello messages themselves are responsible of consuming a fraction of bandwidth resources and ii) even if Hello messages are sent at the right instants, we have noticed that, in some cases, more time than expected is required to detect a link failure. We attribute this last observation to implementation non-idealities. Available bandwidth improvement grows with node mean speed as expected. The maximum measured improvement is equal to 11%, obtained at 20 m/s, with *a*=0. Note that usage of *a*=0.5 leads to smaller bandwidth increase. The latter constitutes a low-pass filter for TLF values, where the last four or five samples are taken into account [9]. A large link lifetime may compensate subsequent shorter ones, avoiding (or delaying) AHR switch to the High dynamics state.

3.2 Power consumption measurements

We have measured power consumption of nodes in three different situations: i) using default AODV; ii) using AODV with HELLO_INTERVAL = 0.2 seconds, and iii) AODV with AHR mechanism. We have tested AHR with parameter *a* equal to 0, which provides higher connectivity than *a*=0.5. However, AHR is expected to consume more energy with this setting. We have used the same mobility emulation scenario in which bandwidth measurements have been carried out. We obtain the battery lifetime of the sender as follows. First, the batteries of all nodes in the scenario are fully charged. Immediately after, traffic is sent in the aforementioned scenario, while the mobility emulation is started in parallel. Each trial ends when all the battery energy of the sender has been consumed. As shown in Figure 2, AHR leads to a power consumption close to that of default AODV at low node speeds. At high speeds, power consumption is similar to that obtained when a static HELLO_INTERVAL equal to 0.2 seconds is used.

3.3 Memory storage

We have measured the memory storage required by AHR. `gcc` has been used for compiling code in all cases. AHR represents a 1.8% increase in terms of memory storage. We believe this figure is assumable, though further code optimization should be performed.

4. ACKNOWLEDGMENTS

This work was supported in part by FEDER and the Spanish Government through project TIC2003-01748.

5. REFERENCES

[1] Lundgren H., Nordstrom E., Tschudin C., "Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks", IEEE WoWMoM'02, September, 2002

[2] Chakeres I.D., Belding-Royer E., "The Utility of Hello Messages for Determining Link Connectivity", 5th WPMC 2002

[3] Gomez C., Catalan M., Mantecon X., Paradells J., Calveras A., "Evaluating Performance of Real Ad-hoc Networks Using AODV with Hello Message Mechanism for Maintaining Local Connectivity", IEEE PIMRC 2005, September 2005

[4] Perkins C., Belding-Royer E., Das S., "Ad hoc On Demand Distance Vector Routing (AODV)", RFC 3561, July 2003

[5] AODV-UU implementation: http://core.it.uu.se/AdHoc/ImplementationPortal

[6] AODV implementation with AHR mechanism: http://agatha.upc.es/AHR/AHR.html

[7] Postel J., "Transmission Control Protocol", RFC 793, September 1981.

[8] Gomez C., Marchador X., Gonzalez V., Paradells J., "Multilayer analysis of the influence of mobility models on TCP flows in AODV ad-hoc networks", in Proc. of the 14th IEEE LANMAN 2005, Chania, Greece, September 2005

[9] Stallings, W. "Data and computer communications", 7th Edition, Prentice Hall, 2004