An OLSR implementation, experience, and future design issues

Implementation

- Nrlolsr evolved from OLSR draft version 3
  - NRL had previous work which modified INRIAv3 code for research purposes
  - This effort was a complete restart and independent code implementation
- Built on top of NRLs protolib library for system portability
- Works with: linux, windows, wince, openzaurus, ns-2, opnet
- Nrlolsr is a research oriented OLSR implementation
- Has non-standard command line options for research purposes
- Nrlolsr is not fully rfc3626 compliant
  - Does not support MID messages or dual interfaces
  - Does not have standard TC_REDUNDANCY
  - Has different message jitter functionality
  - Uses a link-local multicast address instead of broadcast by default

History

- **March 01:** Release for ns-2 OLSR draft v3
- **June 01:** Updated to draft v4
- **April 03:** Release for linux, draft v8

  Ns was temporarily not supported
- **May 03:** IPv6 support added
- **October 03:** Release for windows
- **Febuary 04:** Release for wince, ns2, and opnet

Current Design Structure

- Nrlolsr is built upon protolib and does not make any direct system calls.
- Protolib provides a system independent interface.
- Timers, socket calls, route table management, address handling are all taken care of though protolib calls.
- Core OLSR code is used for all supported systems.
- Porting Nrlolsr to a new system only requires re-defining existing protolib function calls.

Research

- Early development of nrlolsr helped test out early specifications.
- Nrlolsr is used to test out new methods.
  - Load balancing (not currently supported)
  - Fuzzy-sighted flooding
  - Tos routing
  - Simplified multicasting
- Nrlolsr was used to collect data for two NRL Milcom papers.
  - J. Dean, J. Macker, “A Study of Link State Flooding Optimizations for Scalable Wireless Networks,” MILCOM, Oct, 2003
  - J. Macker, J. Dean, W. Chao, “An Implementation and Study of Simplified Multicast Forwarding in Mobile Ad hoc Networks,” MILCOM, Nov, 2004 (pending)

Research Options

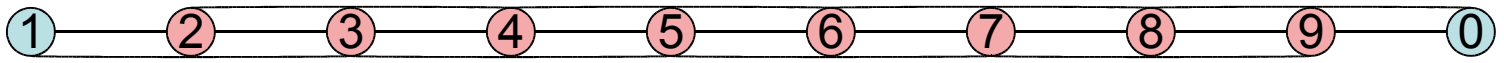
- **-al** option will set TC_REDUNDANCY higher than 2. All nodes source tc messages of all of their neighbors.
- **-fuzzy** option will set ttl and validity time of outgoing tc messages based upon a preset equation.
- **-slowdown** option attempts to slow down the rate of sending tc messages if local neighborhood is stable
- **-spf** will perform shortest path first route calculations and send around spf information set manually (ns) or though packets from outside sources (custom mac layer). ~Qos delay
- **-minmax** will perform maximum smallest value calculations and send around minmax values set in same manor as spf. ~Qos throughput.

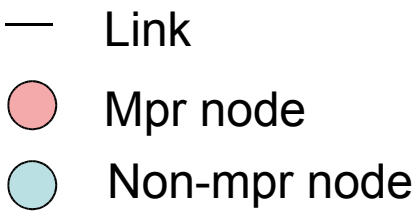
OLSR observations

- Number of MPRs in a network is not necessarily on the same scale as the total number of forwarders.
- Smaller hello interval or mac layer sensing generally improves route connectively.
- Willingness values in the range of 1-6 in most random networks do not drastically change overall performance. 0 and 7 can.
- -al option generally helps (-al=TC_REDUNDANCY2+)
- Its important to use active timers for hysteresis and not rely on sequence numbers.
- Jitter as specified changes interval.

MPR vs Forwards example

8 MPRs and at most 4 forwards

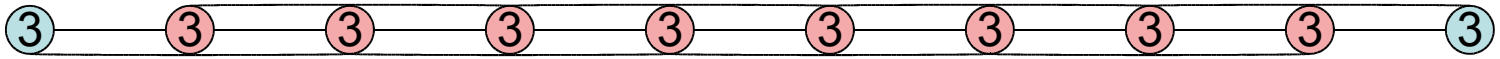




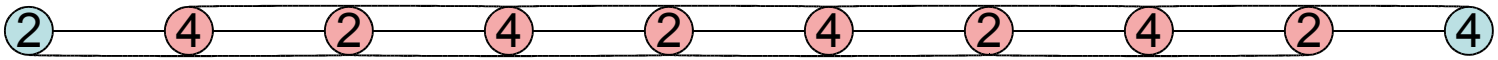
Willingness example

Willingness values in node

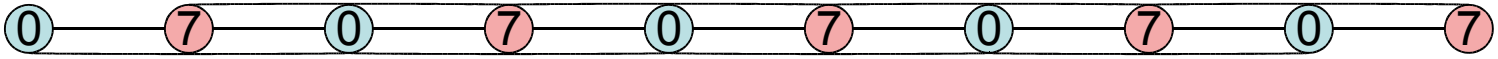
Default



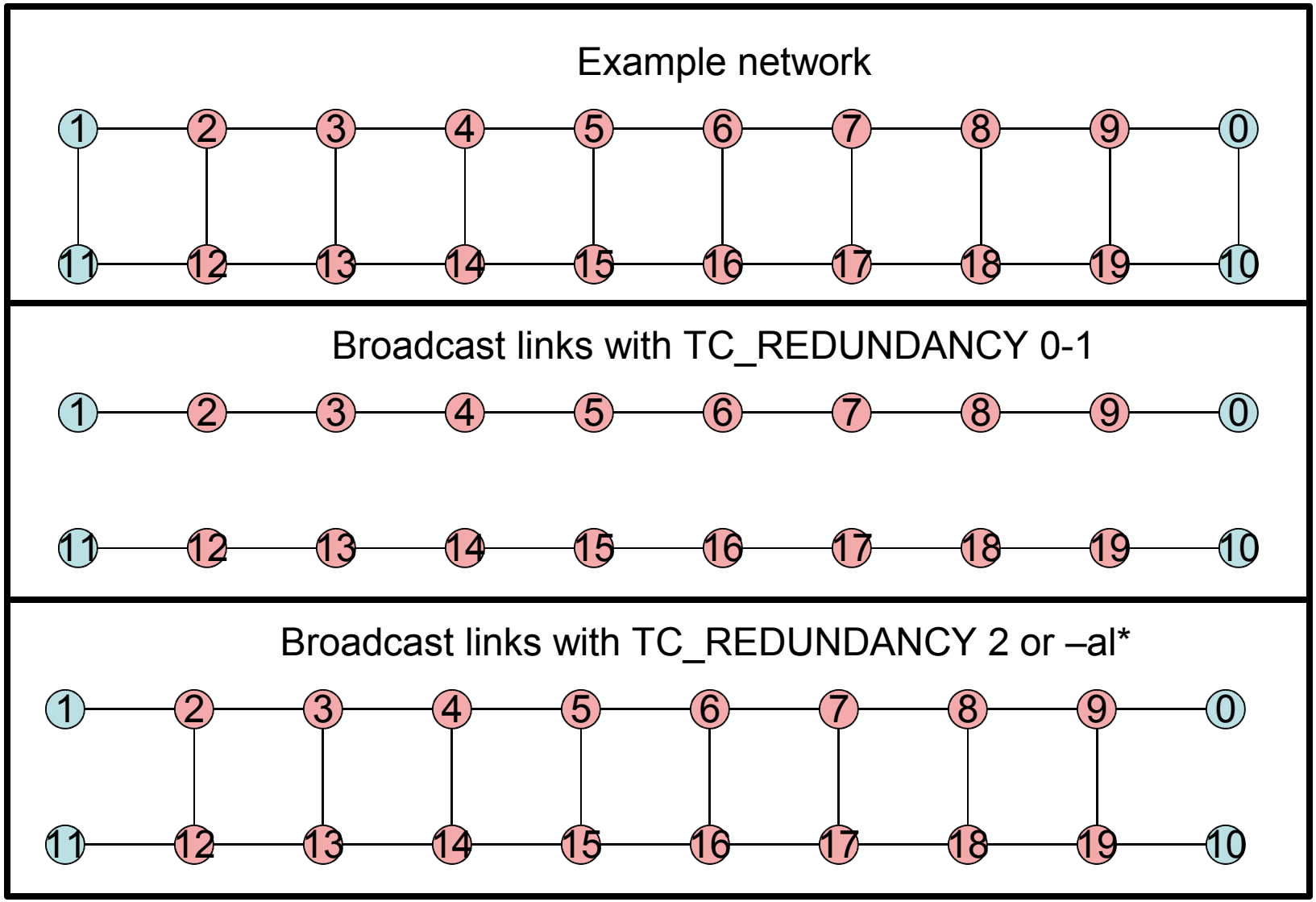
Willingness using 1-6 range



Willingness using 0 and 7



TC_REDUNDANCY example



Debugging Examples

- Simulation work solved many debugging issues
  - Many complex scenarios were examined
  - Packet forwarding code was located in wrong place causing extra forwards.
  - Incorrect route calculations in which next *connecting* link was added no matter hop count.
  - Willingness method broken while using –al option.

- User reported bug issues
  - Nrlolsr was forwarding ttl of 1
  - Many system specific issues
  - To many other to list

- Interop discovered bugs
  - Broadcast address configuration does not work correctly in linux
  - Parsing dual message packets does not process any but first message
  - Previous IPv6 support appears broken in NRLOLSR 7.4 release (will fix in next release)

Possible Future Improvements

- Hybrid proactive/reactive protocol
- Load balancing/type of service routing

Hybrid OLSR
Current limitations and possible solutions

- No standard way of limiting flooding in terms of area
  - Develop 3 or more standard flooding patterns
  - Send interval time and flooding pattern instead of validity time and extrapolate validity time using hop count.
- No event driven link state messages
  - Allow event driven link state messages but put a cap on the amount
  - Only send event driven tcs when global routing will be affected
- No reactive route building and suggested approach
  - Define when route requests might be sent
  - Include path state in route requests
  - Have route replies include full link state path to destination
  - Allow virtual links of greater than one hop
  - Add links to link state database and redo routing table

Load balancing and TOS routing

- OLSR does not presently have a standard method for extra link cost values to be sent
  - Allow for TC+ messages which include fields for link values and type of value field
  - TC+ messages should also include node values
  - Route calculation methods should be developed for different types of service
- Many routing table implementations do not allow for load balancing or ToS routing

Conclusion

- Interop a success – several lessons learned
- OLSR is mature but room still exists for improvement
- Nrlolsr provides both simulation and real world operations
- Nrlolsr is a research oriented implementation
- Nrlolsr is freely available at pf.itd.nrl.navy.mil