RapidMesh

A Declarative Toolkit for Rapid Experimentation of Wireless Mesh Networks

http://netdb.cis.upenn.edu/rapidnet

Shivkumar C. Muthukumar*, Xiaozhou Li*, Changbin Liu*, Joseph B. Kopena†, Mihai Oprea*, Richardo Correa*, Boon Thau Loo*, Prithwish Basu††

*** University of Pennsylvania** **† Drexel University** **†† BBN**

Motivation

- Proliferation of MANET routing protocols
  - Reactive (DSR, AODV)
  - Proactive (LS, HSLS, OLSR)
  - Epidemic
- No “one size fits all” protocol.
  - Variations in network mobility and connectivity.
  - Wide range of traffic patterns.
- Lack of systematic tools for rapid prototyping.

Motivation

- Simulation studies are useful but may not be complete.
  - Real-world effects manifest themselves in actual deployments.
- Advent of open wireless testbeds (like ORBIT) allows evaluation under realistic settings.
- Deploying and experimenting on testbeds remains arguably time consuming.
- A case for unified tool support for
  - Simulation: Controlled large scale experiments under a variety of mobility models.
  - Testbed-based experimentation: Evaluation under real world effects.

Outline

- Motivation
- Overview of RapidMesh
- Background on Declarative Networking
- Rapid Prototyping Example: LS to HSLS
- Evaluation on ORBIT Testbed
- Ongoing Work

Overview: The Approach

- A development toolkit that unifies rapid prototyping, simulation and experimentation.
- Integrates a *declarative networking engine* with the ns-3 network simulator and emulator.
- Declarative Networking **[Loo et. al., SIGCOMM '05]**
  - Use of database query languages to specify protocols.
  - Compiled to distributed dataflows and executed by a distributed query engine.
- ns-3 network simulator and emulator (http://www.nsam.org)
  - Discrete event simulator targeted Internet systems.
  - Intended as an eventual replacement of ns-2.

Overview: Why Declarative Networking?

- Compact and high-level representation of protocols.
- Orders of magnitude reduction in code size.
  - MANET Protocols
    - Proactive: Link State – 8 rules, HSLS – 14 rules, OLSR – 27 rules
    - Reactive: DSR – 10 rules
    - DTN: Epidemic – 16 rules
  - Overlay Networks: Chord DHT – 48 rules
- Rapid prototyping and ease of customization.
- Implementation of verification and correctness checks.

Why ns-3?

- A feature-rich toolkit for networking experiments.
- Open source – collaboration and sharing.
- Easy to get started and work with.
- Emulation capabilities - ns-3 emulator based on raw sockets.
- Unifies simulation with emulation – Same specifications are used.
- One less tool to learn - compared to a standalone system.

Declarative Networking: Background

- A database-inspired approach to define network behavior.
- Nodes are modeled as databases.
- The protocol is specified in terms of database query rules.
- A declarative paradigm
  - Specifying “what” to do instead of “how”
  - Specification is confined to “what” and implementation of the “how” is automated.
- Declarative networks perform efficiently compared to imperative implementations.

Declarative Networking: Background

- RapidMesh uses the *Network Datalog* (NDlog) language.
  - A distributed recursive query language for querying networks.
- Based on *Datalog*
  - *A* rule-based language for querying graph structures.
- NDlog rules are compiled into distributed dataflows
  - Execution model is similar to the *Click* modular router.
- A good balance of flexibility, performance and safety.

NDlog Example: All-pairs Reachable

Rule Head

Rule Body

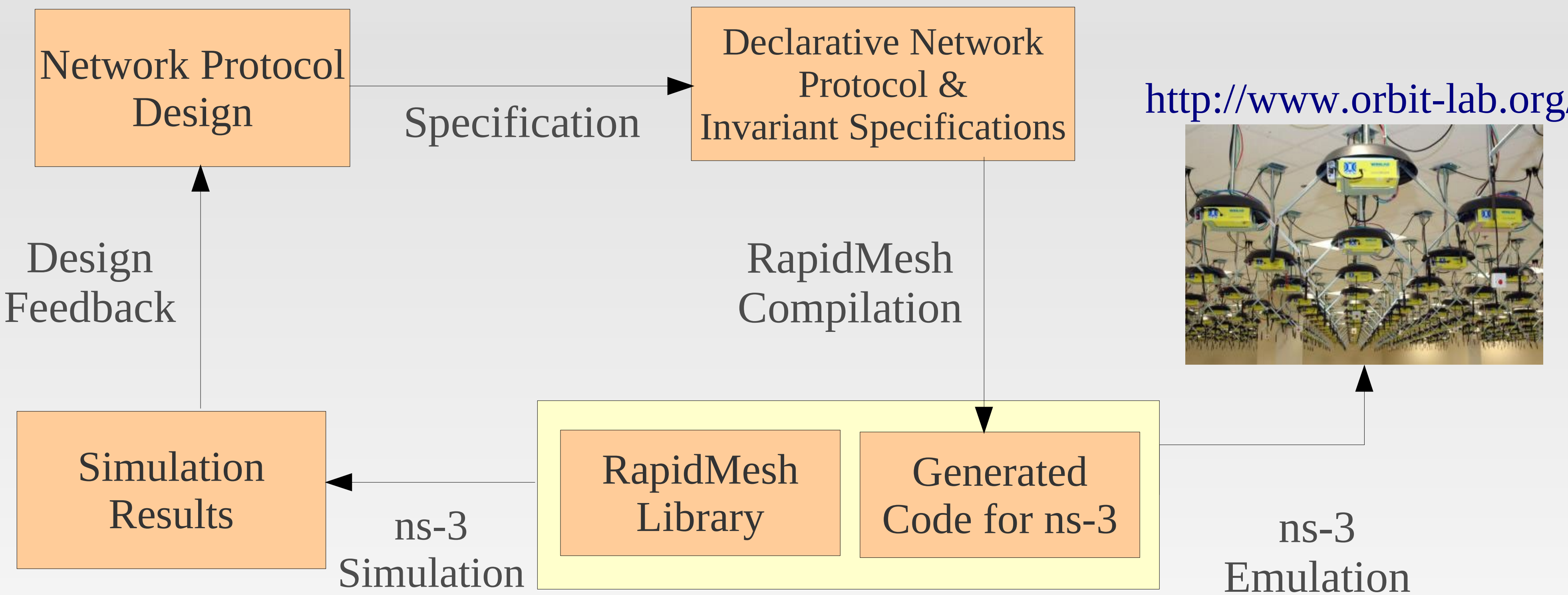
```
r1 reachable(@S,N) :- link(@S,N).
r2 reachable(@S,D) :- link(@S,N), reachable(@N,D).
```

@: Location Specifier

link@S and **reachable@N** distributed match on **N**

- Input: **link (source, neighbor)** table
- Output: **reachable (source, destination)** table
- **r1:** Computes all pairs of nodes reachable in one hop.
- **r2:** If there is a link from S to N and N is reachable from D, then S can reach D.
- Distributed transitive closure computation.

RapidMesh Development Cycle



Outline

- Motivation
- Overview of RapidMesh
- Background on Declarative Networking
- Rapid Prototyping Example: LS to HSLS
- Evaluation on ORBIT testbed
- Ongoing Work

Rapid Prototyping: Link State (LS)

Broadcast specifier

Built-in periodic trigger

```
ls1 lsu(@*,S,N,C,N) :- periodic(@S,T), link(@S,N,C).
ls2 lsu(@*,S,N,C,Z) :- lsu(@Z,S,N,C,W).
```

- Input: **link (src, next, cost)** table
- Output: **lsu (loc, src, dest, cost, from)** table
- **ls1:** Periodically, store links as link state updates (*lsu*)
- **ls2:** Broadcast forward the lsu data to neighbors.

What is Hazy-Sighted Link State (HSLS)?

- A scalable variant of LS
- Suitable for high rate of change of network topology
- Basic idea: Route updates from farther in the network are less significant.
- Use scoped flooding, i.e. -
  - Updates to farther nodes sent less frequently.
  - Use of a TTL field to limit the forwarding of updates
- Updates to 2^k hop neighbors sent with a period 2^k * Tp
- Updates are forwarded if TTL > 0.

Rapid Prototyping: HSLS

```
hs1 lsu(@*,S,N,C,N,TTL) :- periodic(@S,T), link(@S,N,C),
                           TTL:=f_pow(2,K), T:=TTL*Tp,
                           K:=range[1,5].
hs2 lsu(@*,S,N,C,Z,TTL) :- lsu(@Z,S,N,C,W), TTL > 0.
```

- Input: **link (src, dest, cost)** table
- Output: **lsu (loc, src, dest, cost, from)** table
- **hs1:** Updates to 2^k hop neighbors sent with a period 2^k * Tp.
- **hs2:** Broadcast forward the lsu data if TTL > 0.
- Ease of customization. (Think OLSR!)

Outline

- Motivation
- Overview of RapidMesh
- Background on Declarative Networking
- Rapid Prototyping Example: LS to HSLS
- Evaluation on ORBIT testbed
- Ongoing Work

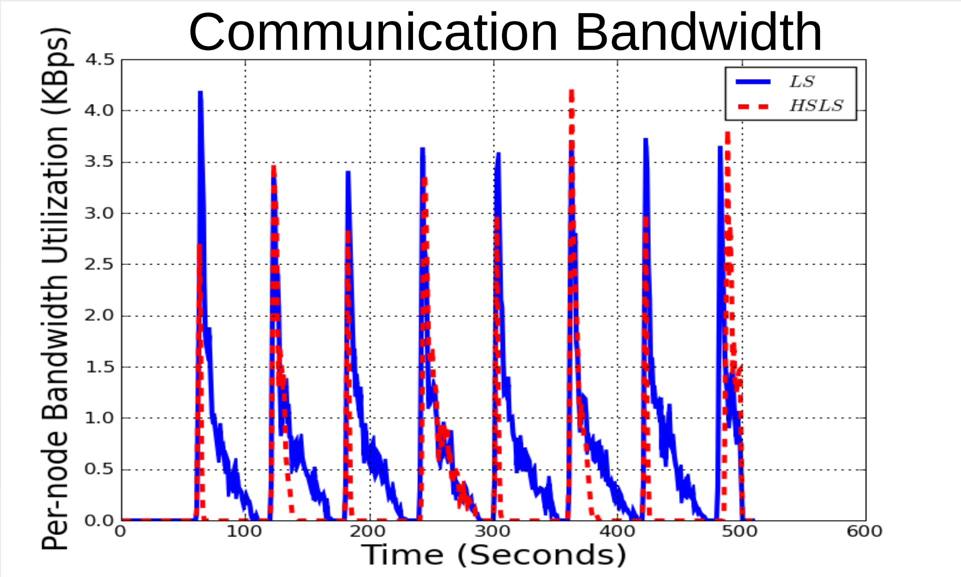
Evaluation: Strategy

- We evaluate declarative implementations of Link State (LS) and HSLS.
- Evaluation modes:
  - ns-3 simulation
  - ns-3 emulation over ORBIT wireless testbed
- Simulation results match emulation results.
- Performance Metrics:
  - Per-node communication bandwidth
  - Average Route Validity
  - Average Route Stretch
- Route Validity: If the links on the computed route exist (0/1).
- Route Stretch: The ratio of the hop counts in the computed route to that in the optimal route (>= 1).

Evaluation: Emulation Setup

- 35 wireless nodes in ORBIT that communicate over 802.11a.
- ns-3 mobility traces of random walk 2-dimensional model.
- Nodes move at 0.15 m/s in a 550m X 750m arena.
- Application level filtering to accept packets only from neighbors.
- Random jitter to reduce collision losses.

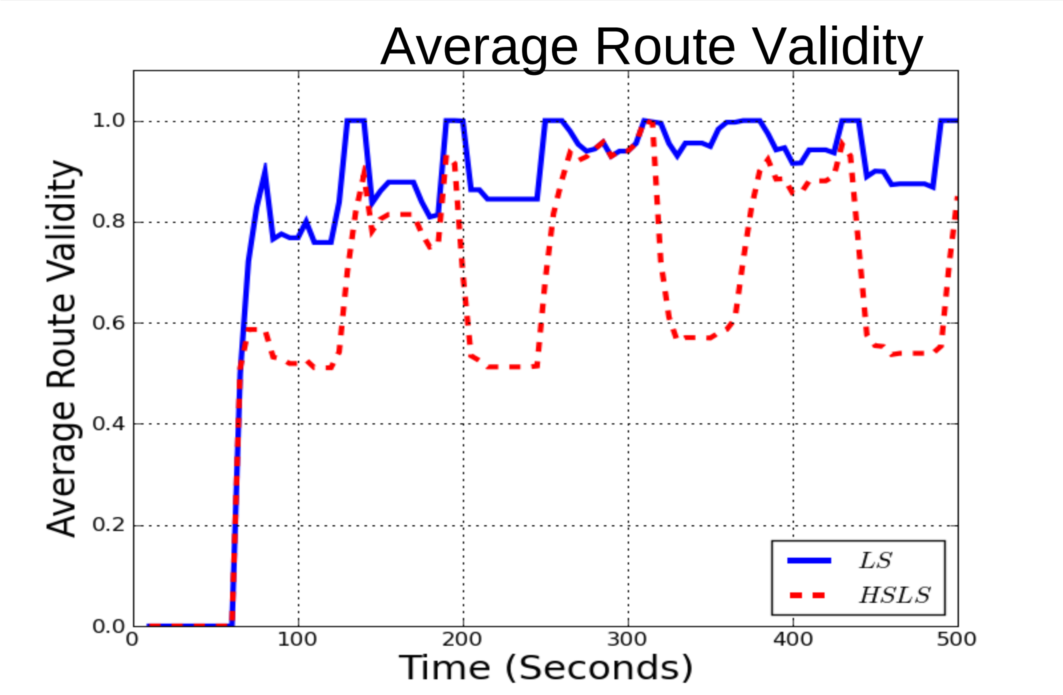
Evaluation: Emulation Results



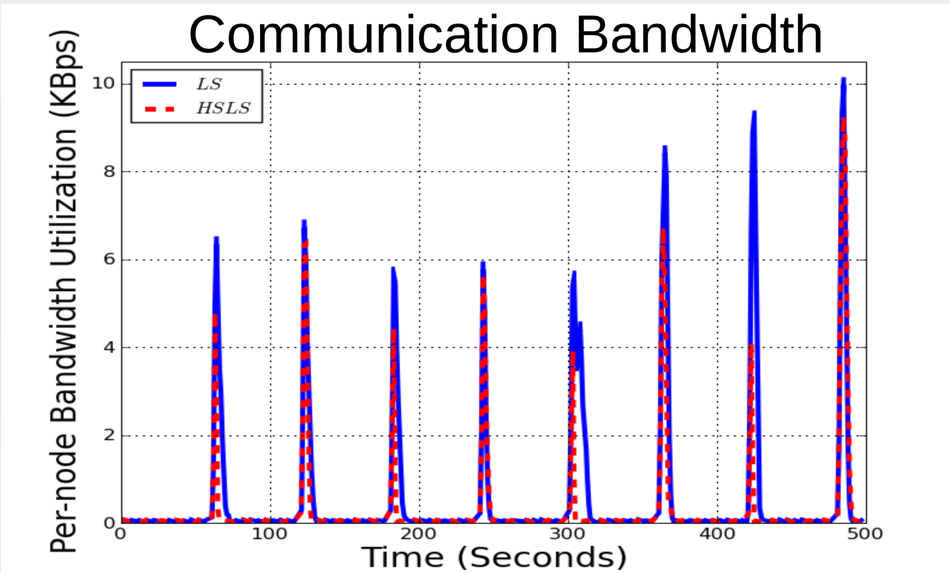
- Lower average bandwidth consumption HSLS (0.29 kB/s) compared to LS (0.61 kB/s) due to scoped flooding.

LS HSLS

- Validity is almost 1 when a periodic flood occurs and drops in between.
- Higher average validity for LS (81%) compared to HSLS (63%).
- HSLS: Lower communication overhead at the price of reduced route quality.



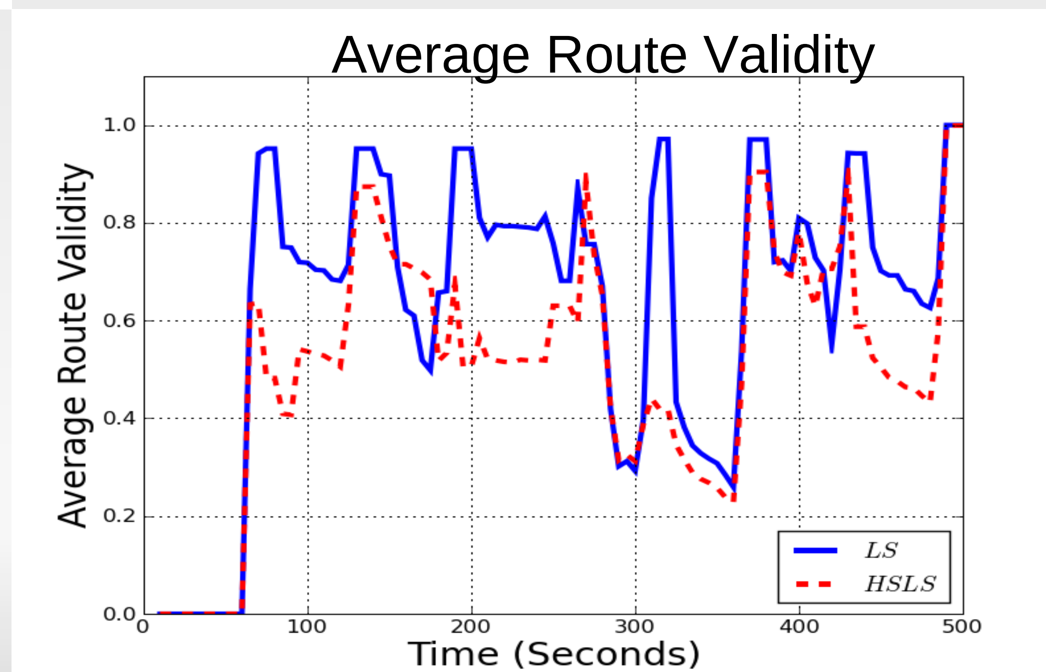
Evaluation: Simulation Results



- Lower average bandwidth consumption HSLS (0.30 kB/s) compared to LS (0.53 kB/s) due to scoped flooding.

LS HSLS

- Validity is almost 1 when a periodic flood occurs and drops in between.
- Higher average validity for LS (64%) compared to HSLS (51%).
- HSLS: Lower communication overhead at the price of reduced route quality.



Evaluation Experiences

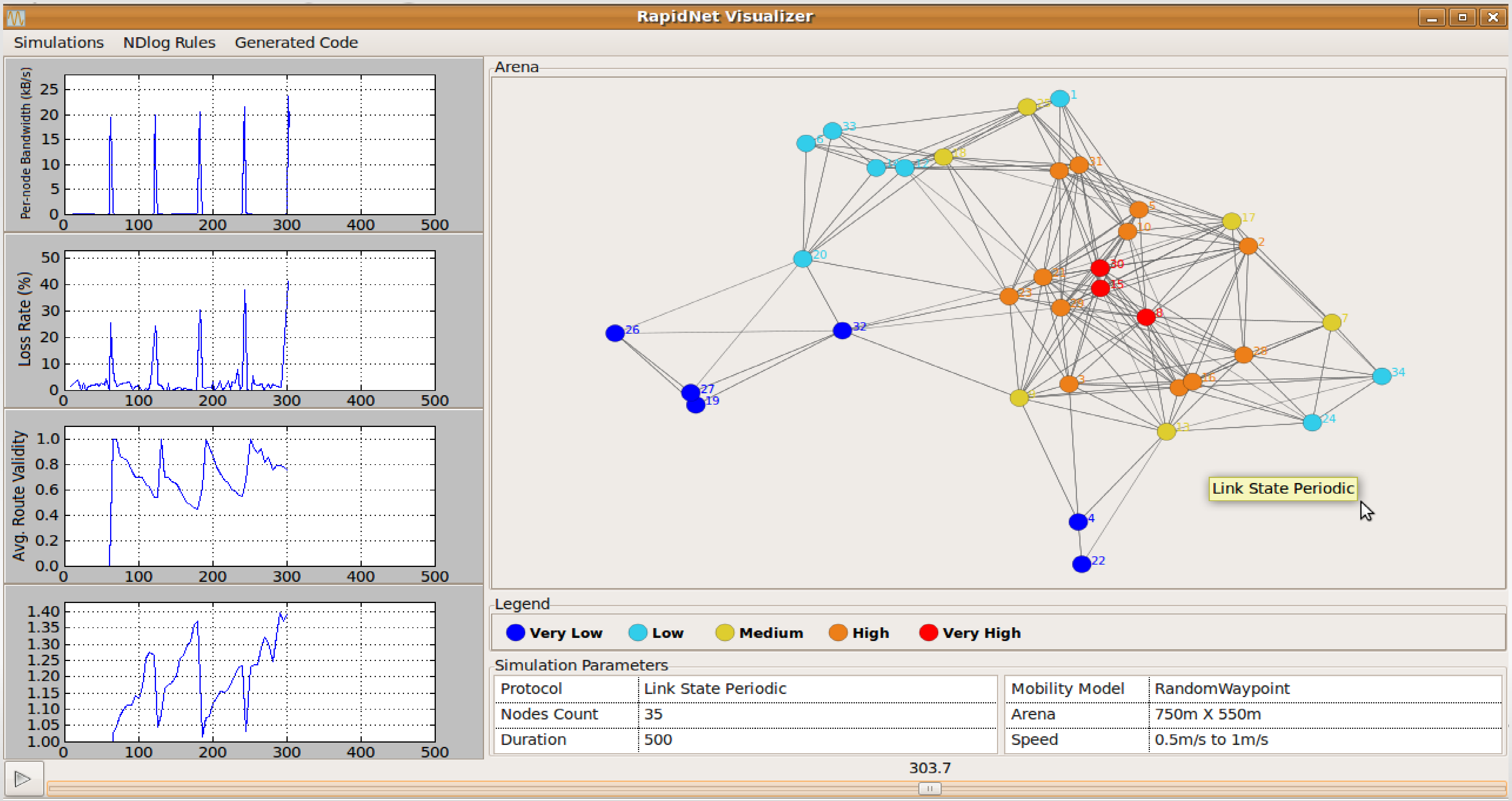
- Few rounds of simulation – correct specifications.
- Run on the ORBIT sandbox, switch WiFiNetDevice (wireless simulation) -> EmuNetDevice (emulation).
- Mobility traces: iptable filtering -> application-level filtering.
- Move to the testbed with 35 nodes.
- High collision losses -> Spaced out floods + random jitter.
- Subsequently, simulation to emulation switch was simply using a different runner script.

Ongoing Work

- Policy-based adaptive MANETs **[ICNP '09]**
  - Policy rules for dynamic switching based on the prevailing conditions.
- Verifiable networking **[HotNets '09]**
  - Translating rules to theorems for proving using mechanized theorem provers.
- Dynamic network composition **[ACM CoNEXT '09]**.

More Information

- Website: http://netdb.cis.upenn.edu/rapidnet/
- Open source code release version 0.1
- RapidMesh demonstration this afternoon.



RapidMesh Summary

- Uses declarative networking for compact specification and rapid prototyping.
- Bridges simulation with test-based experimentation.
- Development:
  - Specify protocols in the NDlog language.
  - Compile to ns-3 code
  - Simulate, emulate, repeat.

Thank You

Questions?