Formal Methods for Wireless Mesh Networks

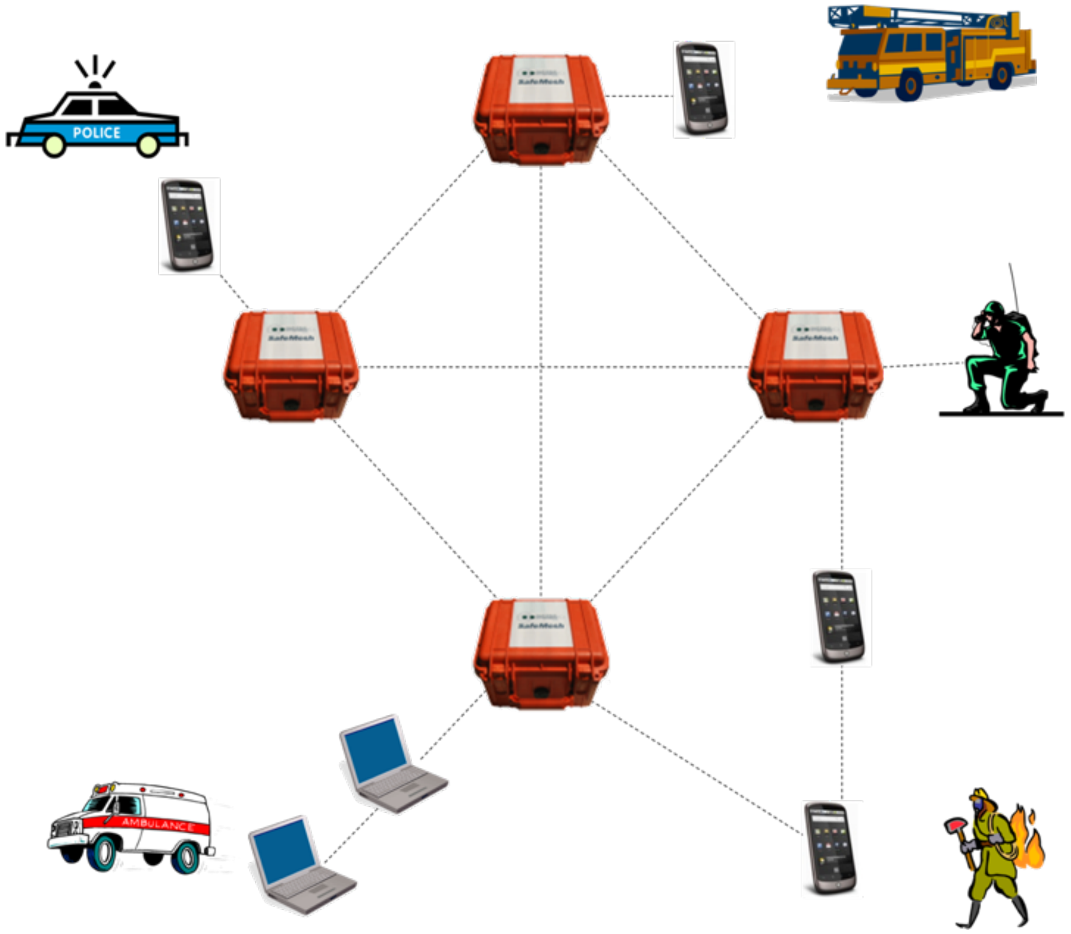
Peter Höfner

Pisa

February 3, 2012

What is the Problem?

- Wireless Mesh Networks (WMNs)
  - key features: mobility, dynamic topology, wireless multihop backhaul
  - quick and low cost deployment
- Applications
  - public safety
  - emergency response, disaster recovery
  - transportation
  - mining
  - smart grid
  - ...
- Limitations in reliability and performance

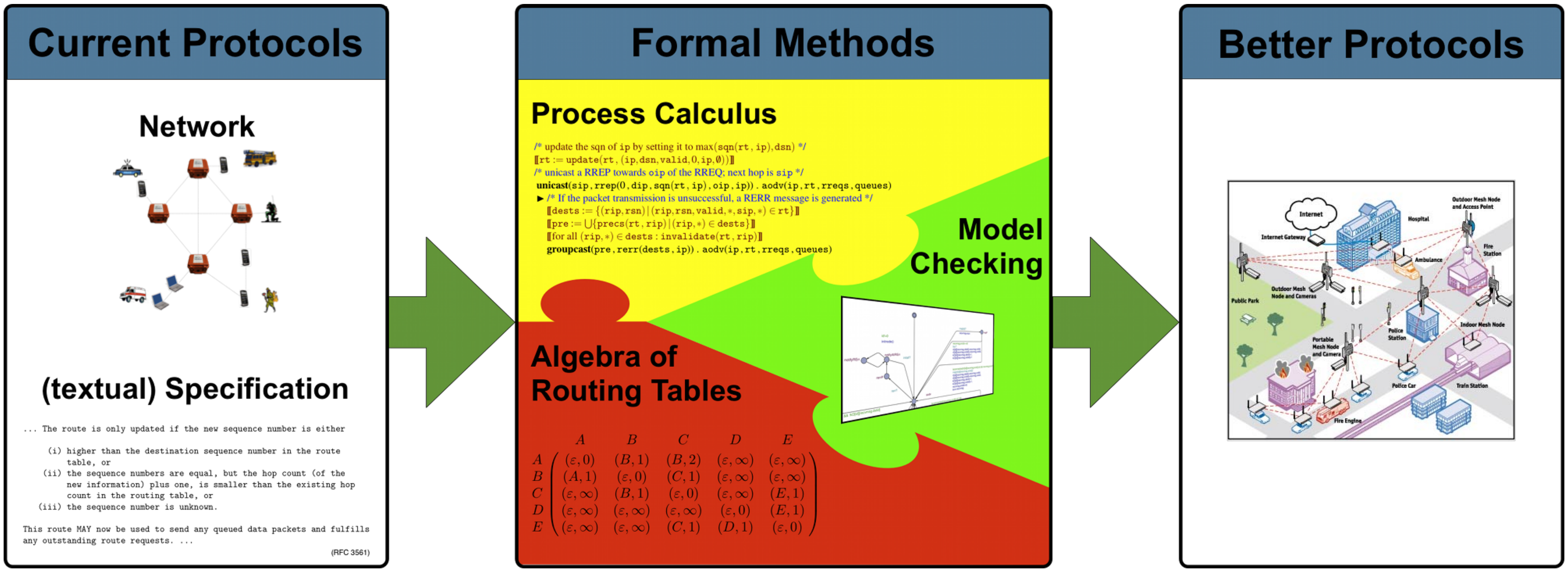


Formal Methods for Mesh Networks

- Goal
  - model, analyse, verify and increase the performance of wireless mesh protocols
  - develop suitable formal methods techniques
- Benefits
  - more reliable protocols
  - finding and fixing bugs
  - better performance
  - proving correctness
  - reduce "time-to-market"
- Team (Formal Methods)
  - Ansgar Fehnker, Rob van Glabbeek, Peter Höfner, Annabelle McIver, Marius Portmann, Wee Lum Tan

Formal Methods for Mesh Networks

- Main Methods used so far
  - process algebra
  - model checking
  - routing algebra



Ad Hoc On-Demand Distance Vector Protocol

- Routing protocol for WMNs

- Ad hoc (network is not static)
- On-Demand (routes are established when needed)
- Distance (metric is hop count)
- Vector (routing table has the form of a vector)

- Developed 1997-2001 by Perkins, Beldig-Royer and Das (University of Cincinnati)

Ad Hoc On-Demand Distance Vector Protocol

- AODV control messages
  - route request (RREQ)
  - route reply (RREP)
  - route error message (RERR)

- Main Mechanism
  - if route is needed
    BROADCAST RREQ
  - if node has information about a destination
    UNICAST RREP
  - if unicast fails or link break is detected
    SEND RERR

Ad Hoc On-Demand Distance Vector Protocol

- *Properties of AODV*
  - *route correctness*
  - *loop freedom*
  - *route found*
  - *packet delivery*

Ad Hoc On-Demand Distance Vector Protocol

- *Properties of AODV*
  - *route correctness* ✓
  - *loop freedom* ✓ (at least for some interpretations)
  - *route found* ✗
  - *packet delivery* ✗

Ad Hoc On-Demand Distance Vector Protocol

- Properties of AODV
  - route correctness ✓
  - loop freedom ✓
  - route found ✗
  - packet delivery ✗
- so far only simulation and test-bed evaluations
  - important, valid methods
  - limitations
    - resource intensive, time-consuming, no generality

Ad Hoc On-Demand Distance Vector Protocol

NICTA

- Properties of AODV
  - route correctness ✓
  - loop freedom ✓ (at least for some interpretations)
  - route found ✗
  - packet delivery ✗
- so far only simulation and test-bed evaluations
  - important, valid methods
  - limitations
    - resource intensive, time-consuming, no generality

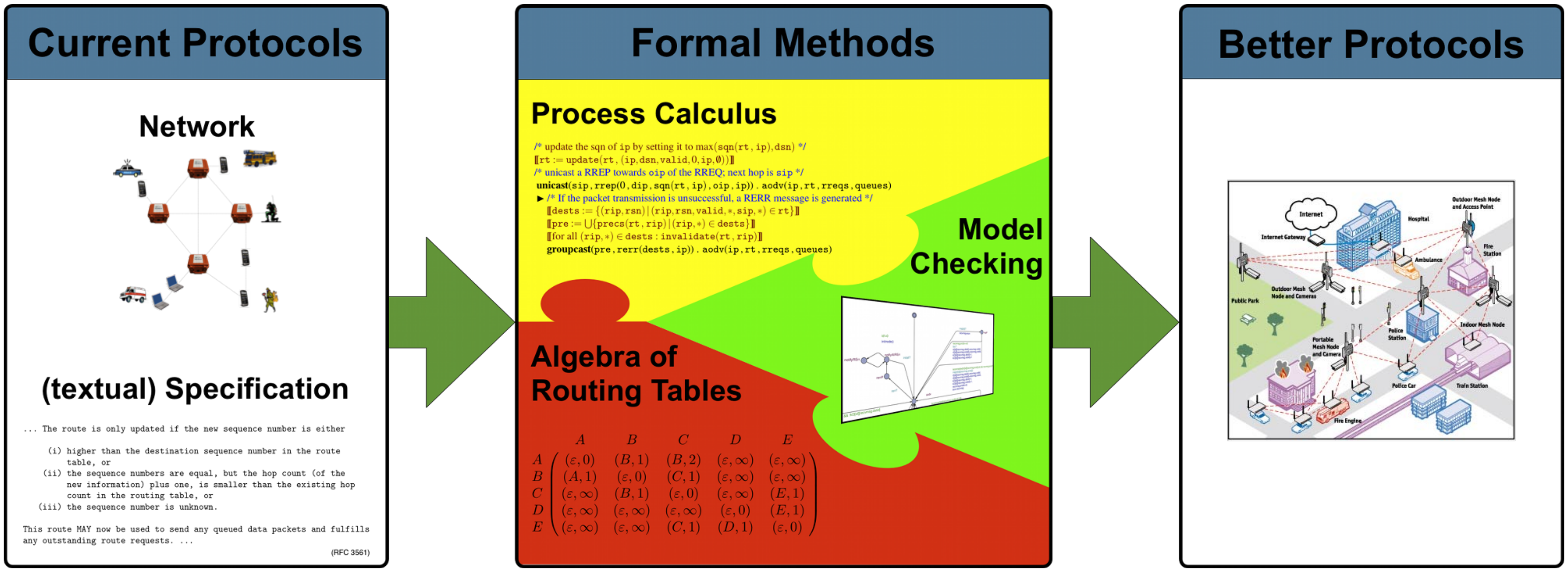
RFC 3561

- Request for Comments (de facto standard)

```
sequence number field is set to false.  The route is only updated if
the new sequence number is either

(i)        higher than the destination sequence number in the route
           table, or

(ii)       the sequence numbers are equal, but the hop count (of the
           new information) plus one, is smaller than the existing hop
           count in the routing table, or

(iii)      the sequence number is unknown.
```

Formal Methods for Mesh Networks

- Main Methods used so far
  - process algebra
  - model checking
  - routing algebra



Process Algebra


```
+ [ (oip, rreqid) ∉ rreqs ]      /* the RREQ is new to this node */
  /* update the route to oip in rt */
  [[rt := update(rt, (oip, osn, valid, hops + 1, sip, ∅))]]
  /* update rreqs by adding (oip, rreqid) */
  [[rreqs := rreqs ∪ {(oip, rreqid)}]]
  (
    [ dip = ip ]      /* this node is the destination node */
      /* update the sqn of ip by setting it to max(sqn(rt, ip), dsn) */
      [[rt := update(rt, (ip, dsn, valid, 0, ip, ∅))]]
      /* unicast a RREP towards oip of the RREQ; next hop is sip */
      unicast(sip, rrep(0, dip, sqn(rt, ip), oip, ip)) . AODV(ip, rt, rreqs, queues)
      ▶ /* If the packet transmission is unsuccessful, a RERR message is generated */
        [[dests := {(rip, rsn) | (rip, rsn, valid, ∗, sip, ∗) ∈ rt}]]
        [[pre := ⋃{precs(rt, rip) | (rip, ∗) ∈ dests}]]
        [[for all (rip, ∗) ∈ dests : invalidate(rt, rip)]]
        groupcast(pre, rerr(dests, ip)) . AODV(ip, rt, rreqs, queues)
    + [ dip ≠ ip ]      /* this node is not the destination node */
    (
      [ dip ∈ aD(rt) ∧ dsn ≤ sqn(rt, dip) ∧ sqn(rt, dip) ≠ 0 ]      /* valid route to dip that is
      fresh enough */
        /* update rt by adding sip to precs(rt, dip) */
        [[r := addpre(σ_route(rt, dip), {sip}); rt := update(rt, r)]]
```

Process Algebra

- Desired Properties
  - guaranteed broadcast
  - prioritised unicast
  - data structure

- Inspired by
  - π - Calculus
  - ω - Calculus
  - (LOTOS)

Structure of WMNs

NICTA

- User
  - Network as a “cloud”
- Collection of nodes
  - connect / disconnect / send / receive
  - “parallel execution” of nodes
- Nodes
  - data management
    - data packets, messages, IP addresses ...
  - message management (avoid blocking)
  - core management
    - broadcast / unicast / groupcast ...
  - “parallel execution” of sequential processes

Nodes (Sequential Process Expressions)

- Syntax of sequential process expressions

$$
\begin{array}{rcl}
SP & ::= & X(exp_1, \ldots, exp_n) \;\mid\; [\varphi]SP \;\mid\; [\![\mathtt{var} := exp]\!]SP \;\mid\; SP + SP \;\mid \\
 & & \alpha.SP \;\mid\; \mathbf{unicast}(dest, ms).SP \blacktriangleright SP \\
\alpha & ::= & \mathbf{broadcast}(ms) \;\mid\; \mathbf{groupcast}(dests, ms) \;\mid\; \mathbf{send}(ms) \;\mid \\
 & & \mathbf{deliver}(data) \;\mid\; \mathbf{receive}(\mathtt{msg})
\end{array}
$$

Structual Operational Semantics I

NICTA

- internal state determined by expression and valuation

$$
\begin{array}{rcll}
\xi, \mathbf{broadcast}(ms).p & \xrightarrow{\mathbf{broadcast}(\xi(ms))} & \xi, p & \\
\xi, \mathbf{groupcast}(dests, ms).p & \xrightarrow{\mathbf{groupcast}(\xi(dests),\xi(ms))} & \xi, p & \\
\xi, \mathbf{unicast}(dest, ms).p \blacktriangleright q & \xrightarrow{\mathbf{unicast}(\xi(dest),\xi(ms))} & \xi, p & \\
\xi, \mathbf{unicast}(dest, ms).p \blacktriangleright q & \xrightarrow{\neg\mathbf{unicast}(\xi(dest))} & \xi, q & \\
\xi, \mathbf{send}(ms).p & \xrightarrow{\mathbf{send}(\xi(ms))} & \xi, p & \\
\xi, \mathbf{deliver}(data).p & \xrightarrow{\mathbf{deliver}(\xi(data))} & \xi, p & \\
\xi, \mathbf{receive}(\mathtt{msg}).p & \xrightarrow{\mathbf{receive}(m)} & \xi[\mathtt{msg} := m], p & (\forall m \in \mathtt{MSG})
\end{array}
$$

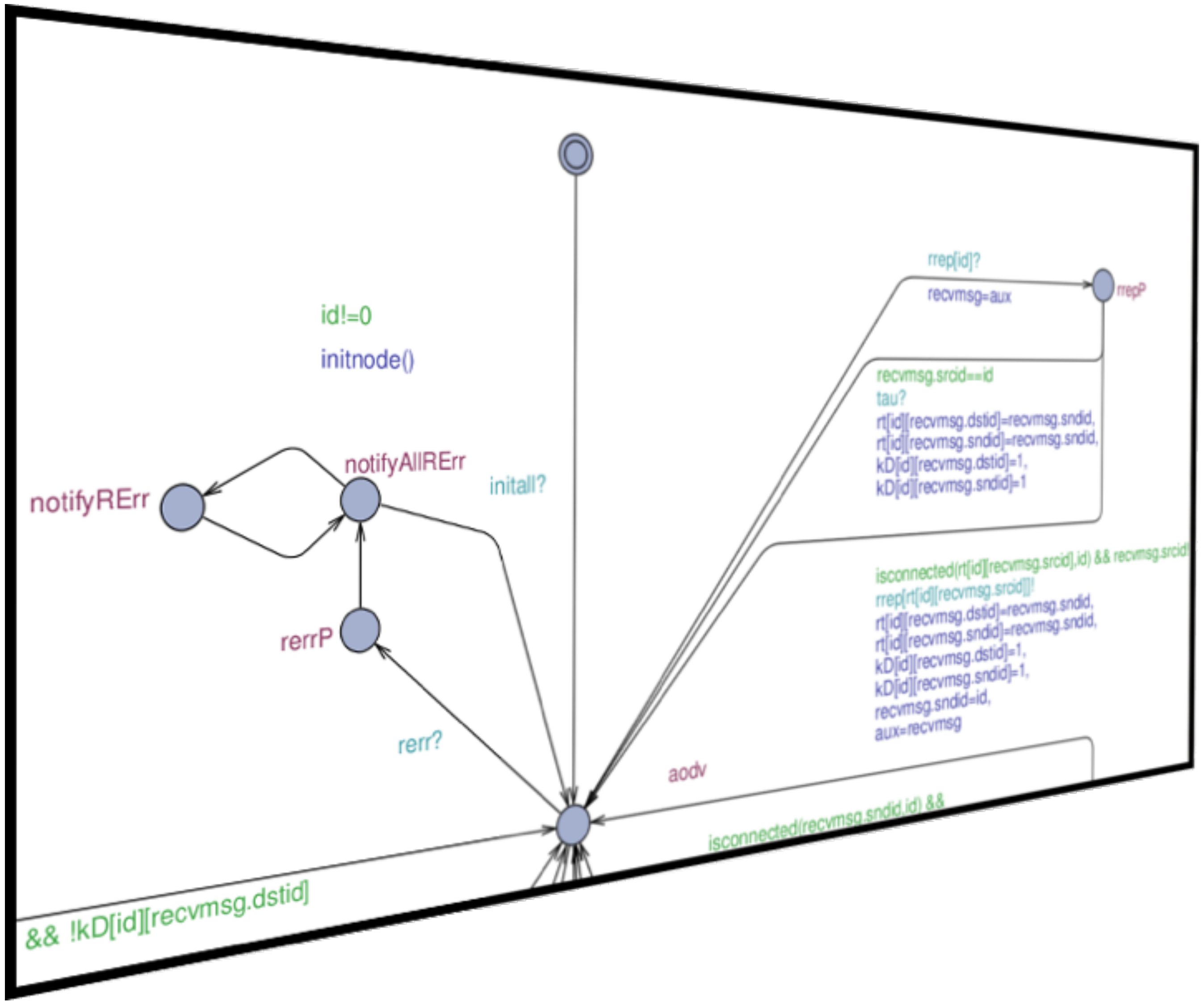
A Bit of Theoretical Results

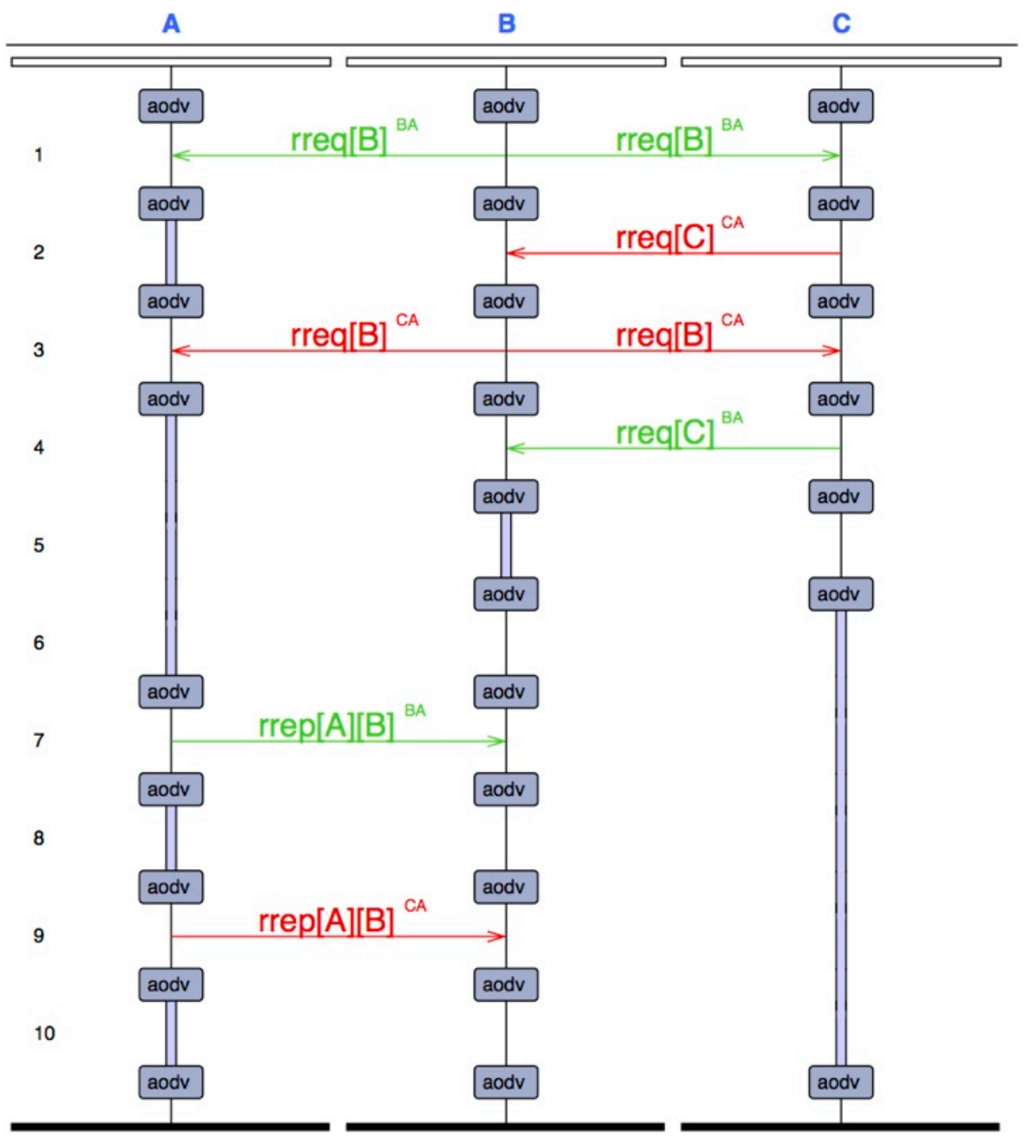
- process algebra is blocking (our model is non-blocking)
- process algebra is isomorphic to one without data structure --- a process for every substitution instance
- resulting algebra is in *de Simone* format (by this strong bisimulation and other semantic equivalences are congruences)
- both parallel operators are associative (follows by a meta result of Cranen, Mousavi, Reniers)

Process Algebra

- New process algebra developed
- Language for formalising specs of network protocols
- Key features:
  - guarantee broadcast
  - prioritised unicast
  - data handling
- Achievements
  - full concise specification of AODV (RFC 3561) (no time)
  - formally verified loop-freedom (without timeouts)
    - invariant proof
  - found several ambiguities, mistakes, shortcomings
  - found solutions for some limitations

Model Checking





Model Checking

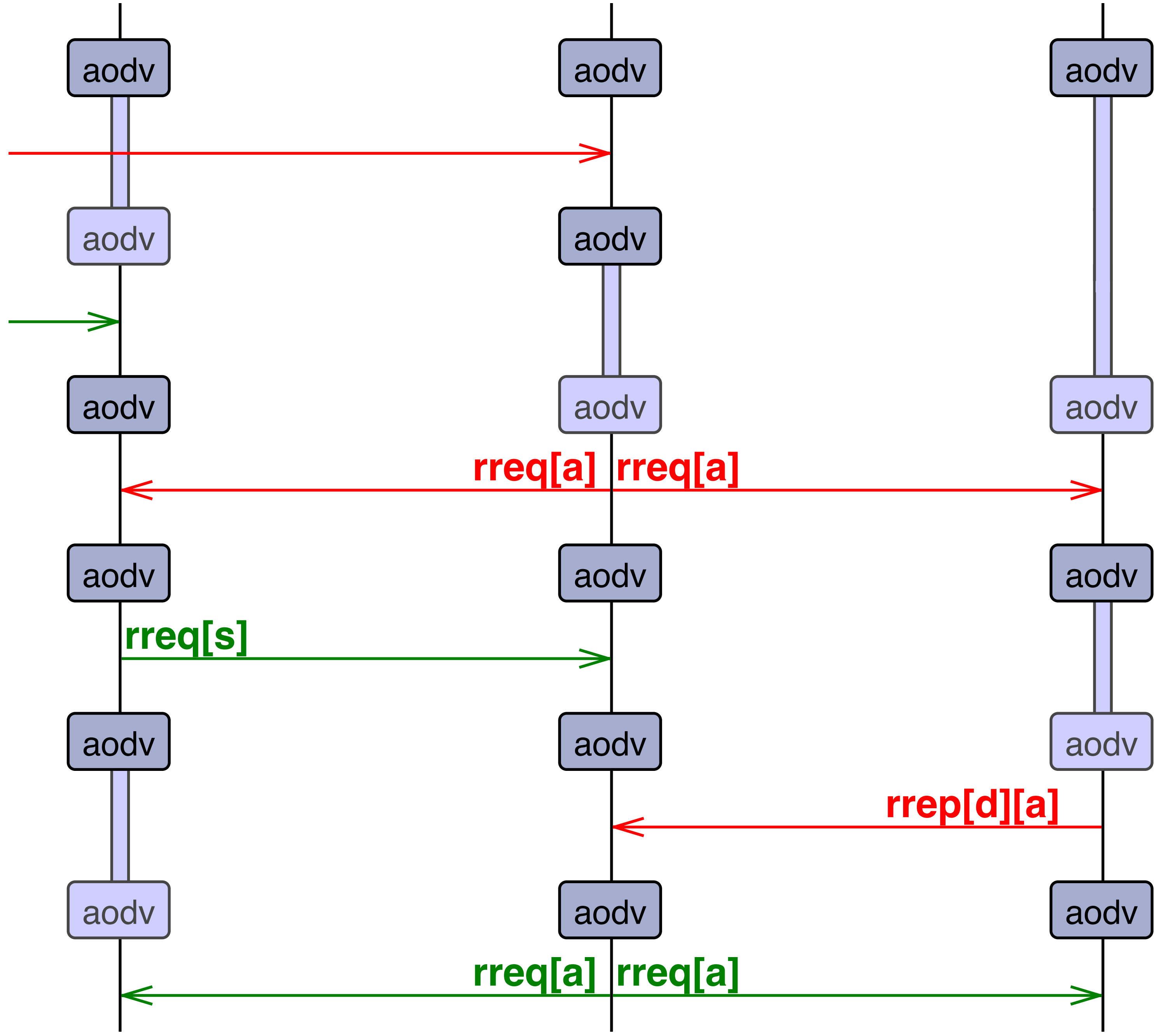
- Model checking routing algorithms
  - executable models
- Complementary to process algebra
  - find bugs and typos in model of process algebra
  - check properties of specification applied to particular topology
  - easy adaption in case of change
  - automatic verification
- Achievements
  - implemented process algebra specification of AODV
  - found/replayed shortcomings

Experiments Set-Up

- Exhaustive search
  - different properties
  - all topologies up to 5 nodes (one topology change)
  - 2 route discovery processes
  - 17400 scenarios
  - variants of AODV (4 models)

Results: Route Discovery (2004)

- Route discovery fails in a linear 3-node topology



Results: Route Discovery

- exhaustive search
  (potential failure in route discovery)
  - static topology: 47.3%
  - dynamic topology (add link): 42.5%
  - dynamic topology (remove link): 73.7%
- AODV repeats route request
- Other solution: forward route reply

Routing Algebra

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{pmatrix} (\epsilon, 0) & (B, 1) & (B, 2) & (\epsilon, \infty) & (\epsilon, \infty) \\ (A, 1) & (\epsilon, 0) & (C, 1) & (\epsilon, \infty) & (\epsilon, \infty) \\ (\epsilon, \infty) & (B, 1) & (\epsilon, 0) & (\epsilon, \infty) & (E, 1) \\ (\epsilon, \infty) & (\epsilon, \infty) & (\epsilon, \infty) & (\epsilon, 0) & (E, 1) \\ (\epsilon, \infty) & (\epsilon, \infty) & (C, 1) & (D, 1) & (\epsilon, 0) \end{pmatrix} \end{array}$$

Routing Algebra – Elements, Operators

- Matrices over routing table entries

$$\begin{array}{cc} & \begin{array}{ccccc} A & B & C & D & \cdots \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \\ \vdots \end{array} & \begin{pmatrix} (\_,0) & (B,1) & (B,2) & (\_,\infty) & \\ (A,1) & (\_,0) & (C,1) & (\_,\infty) & \cdots \\ (\_,\infty) & (B,1) & (\_,0) & (\_,\infty) & \\ (\_,\infty) & (\_,\infty) & (\_,\infty) & (\_,0) & \\ & \vdots & & & \ddots \end{pmatrix} \end{array}$$

routing table of A

“routes” to B

- standard matrix operations
- further abstraction possible (semirings, test, domain, modules ...)

Routing Algebra – Elements, Operators

- Routing table entries (no sequence number so far)

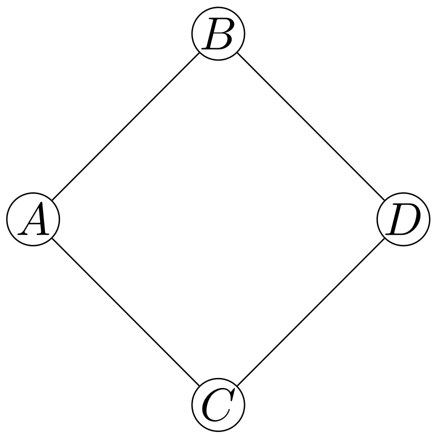
$$(\mathtt{nhip}\,,\,\mathtt{hops})$$

- Choice: $(A,5)+(B,2)=(B,2)$
- Multiplication: $(A,5)\cdot(B,2)=(A,7)$
  - destination and source must coincide

- idea: back to Backhouse, Carré, Griffin, Sobrinho

Example

- A route request is broadcast



$$\underbrace{\begin{pmatrix} (\_,0) & (B,1) & (C,1) & (\_,\infty) \\ (A,1) & (\_,0) & (\_,\infty) & (D,1) \\ (A,1) & (\_,\infty) & (\_,0) & (D,1) \\ (\_,\infty) & (B,1) & (C,1) & (\_,0) \end{pmatrix}}_{\text{topology}} \bullet \underbrace{\begin{pmatrix} (\_,0) & (\_,\infty) & (\_,\infty) & (\_,\infty) \\ (\_,\infty) & (\_,\infty) & (\_,\infty) & (\_,\infty) \\ (\_,\infty) & (\_,\infty) & (\_,\infty) & (\_,\infty) \\ (\_,\infty) & (\_,\infty) & (\_,\infty) & (\_,\infty) \end{pmatrix}}_{\text{sender}} \bullet \underbrace{\begin{pmatrix} (\_,0) & (B,1) & (\_,\infty) & (\_,\infty) \\ (\mathbf{D},\mathbf{3}) & (\_,0) & (\_,\infty) & (\_,\infty) \\ (A,1) & (\_,\infty) & (\_,0) & (D,1) \\ (C,2) & (\_,\infty) & (C,1) & (\_,0) \end{pmatrix}}_{\text{routing table}}$$

$$= \underbrace{\begin{pmatrix} (\_,0) & (B,1) & (\_,\infty) & (\_,\infty) \\ (\mathbf{A},\mathbf{1}) & (\_,0) & (\_,\infty) & (\_,\infty) \\ (A,1) & (\_,\infty) & (\_,0) & (D,1) \\ (C,2) & (\_,\infty) & (C,1) & (\_,0) \end{pmatrix}}_{\text{updated routing table}}$$

Conclusion/Future Work

- So far concentrated on AODV
  - well known
  - IETF standard
- Extend formal methods to other protocols
  - OSLR, DYMO, ...
- Add further necessary concepts
  - time
  - probability (links, measurements)
  - define quality of protocols

NICTA

From **imagination** to **impact**