Formal Methods for Wireless Mesh Networks

Peter Höfner

Singapore
February 22, 2012

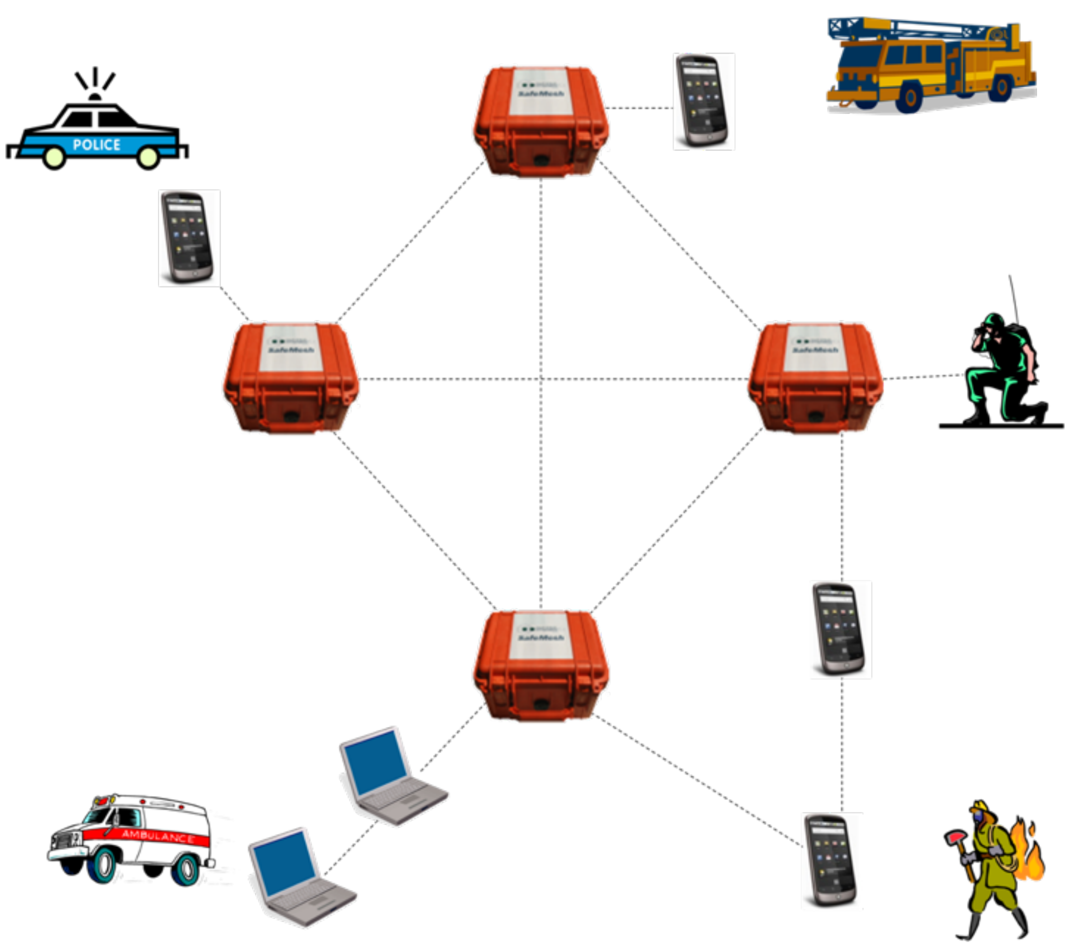
Australian Government
Department of Broadband, Communications and the Digital Economy
Australian Research Council

NICTA Members

NICTA Partners

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

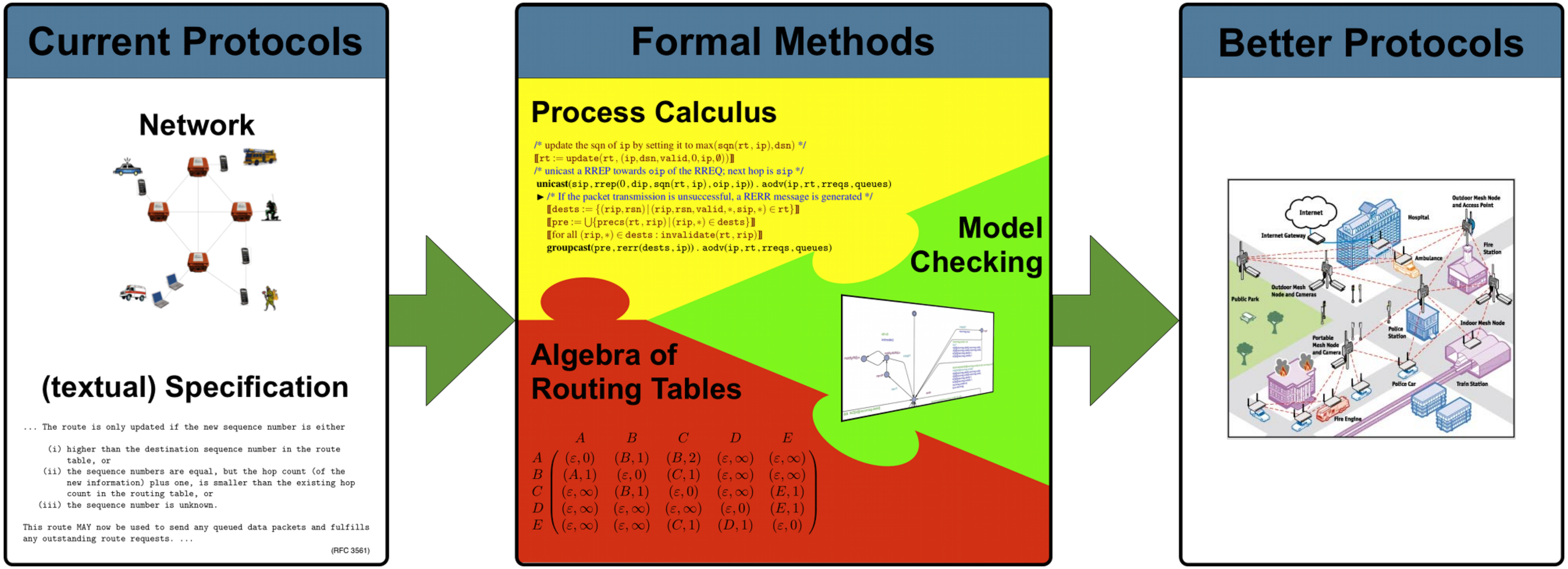


Formal Methods for Mesh Networks

- 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

NICTA

- Properties of AODV
  - route correctness
  - loop freedom
  - route discovery
  - packet delivery

Ad Hoc On-Demand Distance Vector Protocol

NICTA

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

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.
```

Process Algebra

NICTA

$+ [\, (\texttt{oip}, \texttt{rreqid}) \notin \texttt{rreqs} \,]$ /* the RREQ is new to this node */

/* update the route to $\texttt{oip}$ in $\texttt{rt}$ */

$[\![\texttt{rt} := \texttt{update}(\texttt{rt}, (\texttt{oip}, \texttt{osn}, \texttt{valid}, \texttt{hops}+1, \texttt{sip}, \emptyset))]\!]$

/* update $\texttt{rreqs}$ by adding $(\texttt{oip}, \texttt{rreqid})$ */

$[\![\texttt{rreqs} := \texttt{rreqs} \cup \{(\texttt{oip}, \texttt{rreqid})\}]\!]$

(

$[\, \texttt{dip} = \texttt{ip} \,]$ /* this node is the destination node */

/* update the sqn of $\texttt{ip}$ by setting it to $\max(\texttt{sqn}(\texttt{rt}, \texttt{ip}), \texttt{dsn})$ */

$[\![\texttt{rt} := \texttt{update}(\texttt{rt}, (\texttt{ip}, \texttt{dsn}, \texttt{valid}, 0, \texttt{ip}, \emptyset))]\!]$

/* unicast a RREP towards $\texttt{oip}$ of the RREQ; next hop is $\texttt{sip}$ */

$\textbf{unicast}(\texttt{sip}, \texttt{rrep}(0, \texttt{dip}, \texttt{sqn}(\texttt{rt}, \texttt{ip}), \texttt{oip}, \texttt{ip}))\,.\,\texttt{AODV}(\texttt{ip}, \texttt{rt}, \texttt{rreqs}, \texttt{queues})$

$\blacktriangleright$ /* If the packet transmission is unsuccessful, a RERR message is generated */

$[\![\texttt{dests} := \{(\texttt{rip}, \texttt{rsn}) \mid (\texttt{rip}, \texttt{rsn}, \texttt{valid}, *, \texttt{sip}, *) \in \texttt{rt}\}]\!]$

$[\![\texttt{pre} := \bigcup\{\texttt{precs}(\texttt{rt}, \texttt{rip}) \mid (\texttt{rip}, *) \in \texttt{dests}\}]\!]$

$[\![\text{for all } (\texttt{rip}, *) \in \texttt{dests} : \texttt{invalidate}(\texttt{rt}, \texttt{rip})]\!]$

$\textbf{groupcast}(\texttt{pre}, \texttt{rerr}(\texttt{dests}, \texttt{ip}))\,.\,\texttt{AODV}(\texttt{ip}, \texttt{rt}, \texttt{rreqs}, \texttt{queues})$

$+ [\, \texttt{dip} \neq \texttt{ip} \,]$ /* this node is not the destination node */

(

$[\, \texttt{dip} \in \texttt{aD}(\texttt{rt}) \wedge \texttt{dsn} \leq \texttt{sqn}(\texttt{rt}, \texttt{dip}) \wedge \texttt{sqn}(\texttt{rt}, \texttt{dip}) \neq 0 \,]$ /* valid route to $\texttt{dip}$ that is fresh enough */

/* update $\texttt{rt}$ by adding $\texttt{sip}$ to $\texttt{precs}(\texttt{rt}, \texttt{dip})$ */

$[\![\texttt{r} := \texttt{addpre}(\sigma_{route}(\texttt{rt}, \texttt{dip}), \{\texttt{sip}\});\ \texttt{rt} := \texttt{update}(\texttt{rt}, \texttt{r})]\!]$

Process Algebra

- Desired Properties
  - guaranteed broadcast
  - conditional unicast
  - data structure
- Inspired by
  - π- Calculus
  - ω- Calculus
  - (LOTOS)

Structure of WMNs

- 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)

NICTA

- Syntax of sequential process expressions

$$
\begin{array}{rcl}
SP & ::= & X(\mathit{exp}_1, \ldots, \mathit{exp}_n) \mid [\varphi]SP \mid [\![\mathtt{var} := \mathit{exp}]\!]SP \mid SP + SP \mid \\
 & & \alpha.SP \mid \mathbf{unicast}(\mathit{dest}, \mathit{ms}).SP \blacktriangleright SP \\
\alpha & ::= & \mathbf{broadcast}(\mathit{ms}) \mid \mathbf{groupcast}(\mathit{dests}, \mathit{ms}) \mid \mathbf{send}(\mathit{ms}) \mid \\
 & & \mathbf{deliver}(\mathit{data}) \mid \mathbf{receive}(\mathtt{msg})
\end{array}
$$

Structual Operational Semantics I

- internal state determined by expression and valuation

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

Network

- Node expressions: $M ::= \quad ip : P : R \quad | \quad M \| M$
- Operational Semantics (snippet)

$$\frac{P \xrightarrow{\textbf{broadcast}(m)} P'}{ip : P : R \xrightarrow{R:\textbf{*cast}(m)} ip : P' : R}$$

$$\frac{P \xrightarrow{\textbf{unicast}(dip,m)} P' \qquad dip \in R}{ip : P : R \xrightarrow{\{dip\}:\textbf{*cast}(m)} ip : P' : R} \qquad \frac{P \xrightarrow{\neg\textbf{unicast}(dip)} P' \qquad dip \notin R}{ip : P : R \xrightarrow{\tau} ip : P' : R}$$

$$ip : P : R \xrightarrow{\textbf{connect}(ip,ip')} ip : P : R \cup \{ip'\}$$

$$ip : P : R \xrightarrow{\textbf{disconnect}(ip,ip')} ip : P : R - \{ip'\}$$

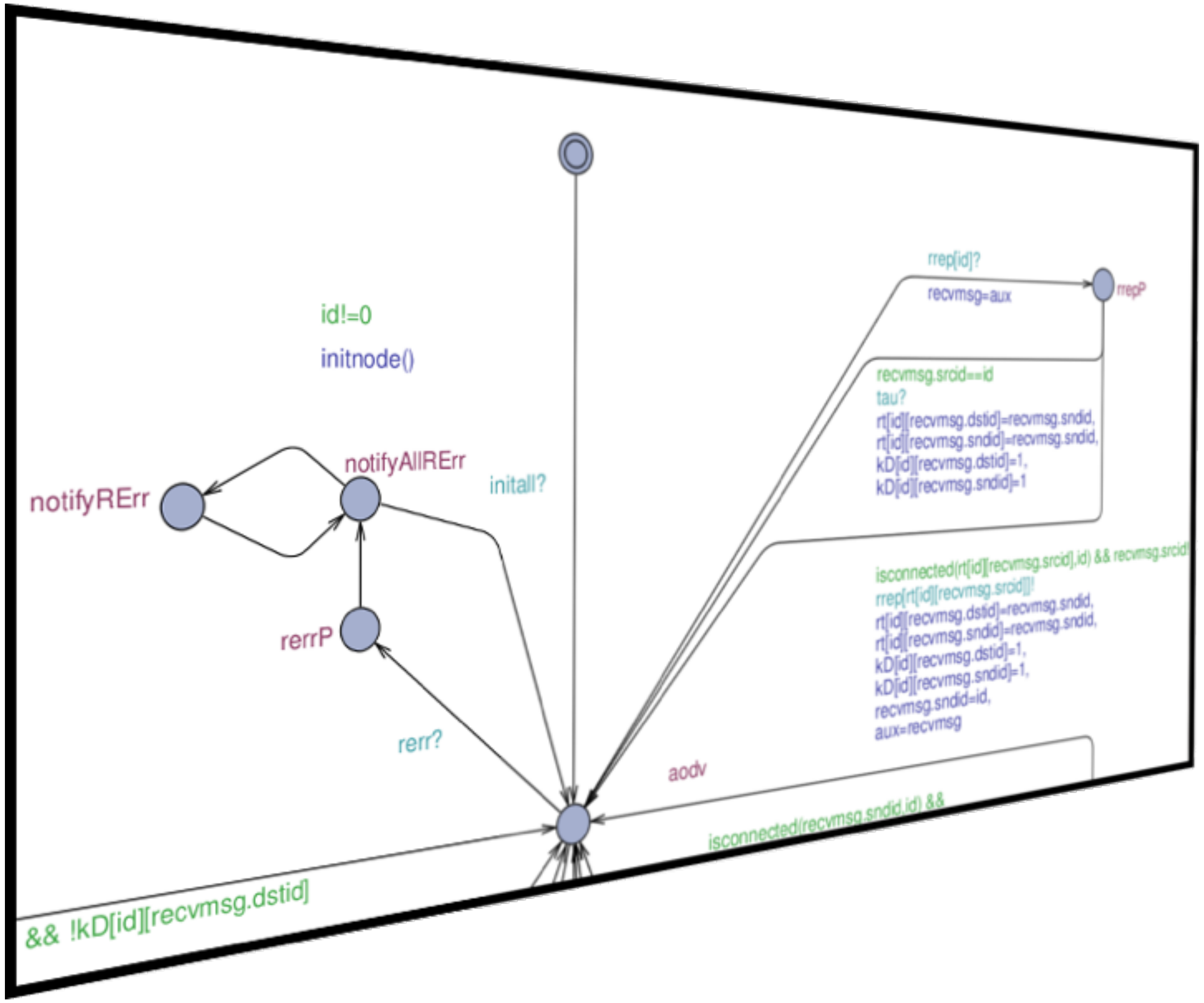
A Bit of Theoretical Results

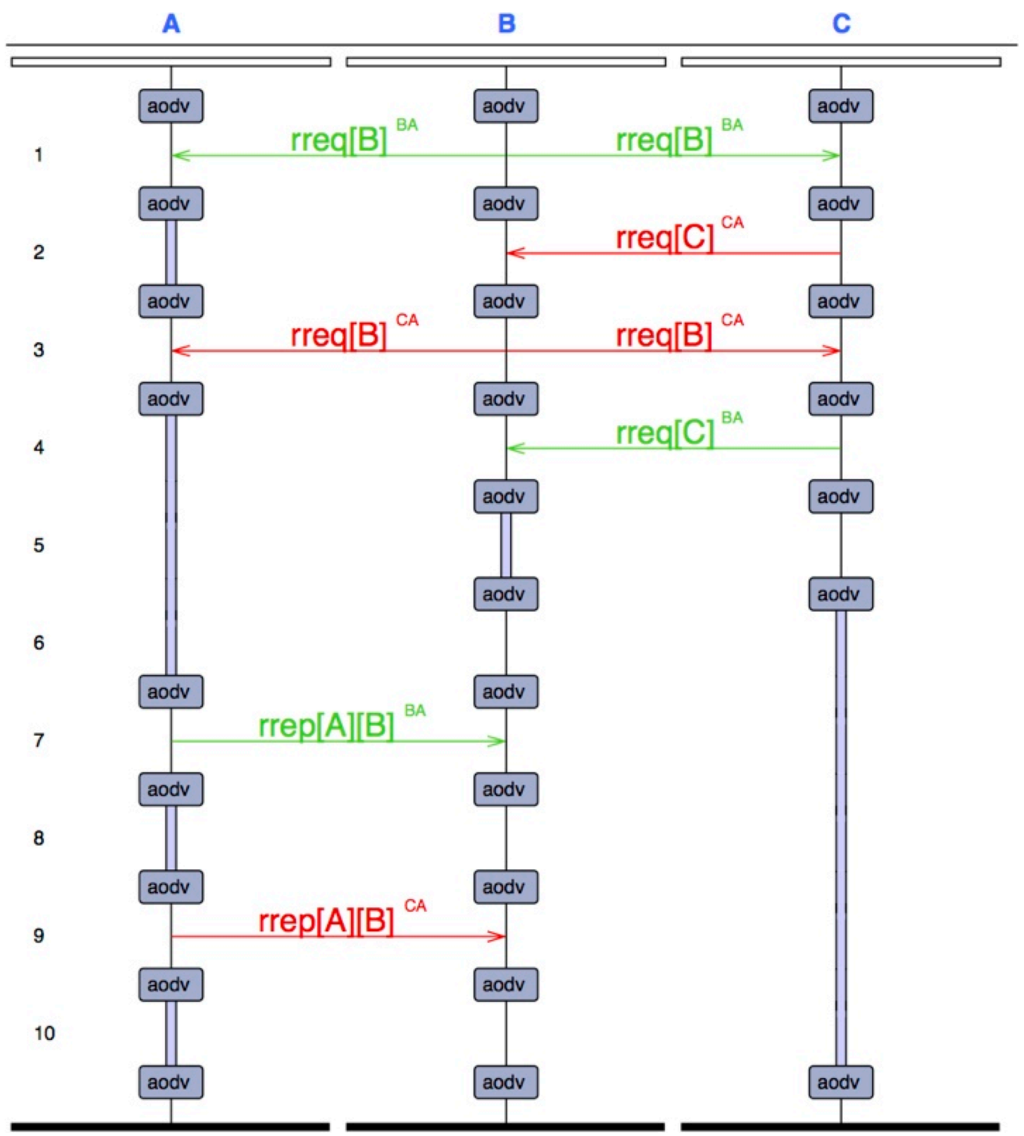
- process algebra is blocking (our model is non-blocking)
- process algebra is isomorphic to one without data structure --- a process for every substitution instance
- resulting algebra is in *de Simone* format
  (by this strong bisimulation is a congruence)
- both parallel operators are associative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

Case Study

- AODV Routing Protocol
- Achievements
  - full concise specification of AODV (RFC 3561) (no time)
  - verified/disproved properties
    - route discovery
    - packet delivery
    - loop freedom
      - first (correct) proof
      - disproved loop freedom for variants of AODV (as implemented in at least open source implementation)
  - 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

UPPAAL Model Checker

- Well established model checker
- Uses networks of timed automata
- Has been used for protocol verification

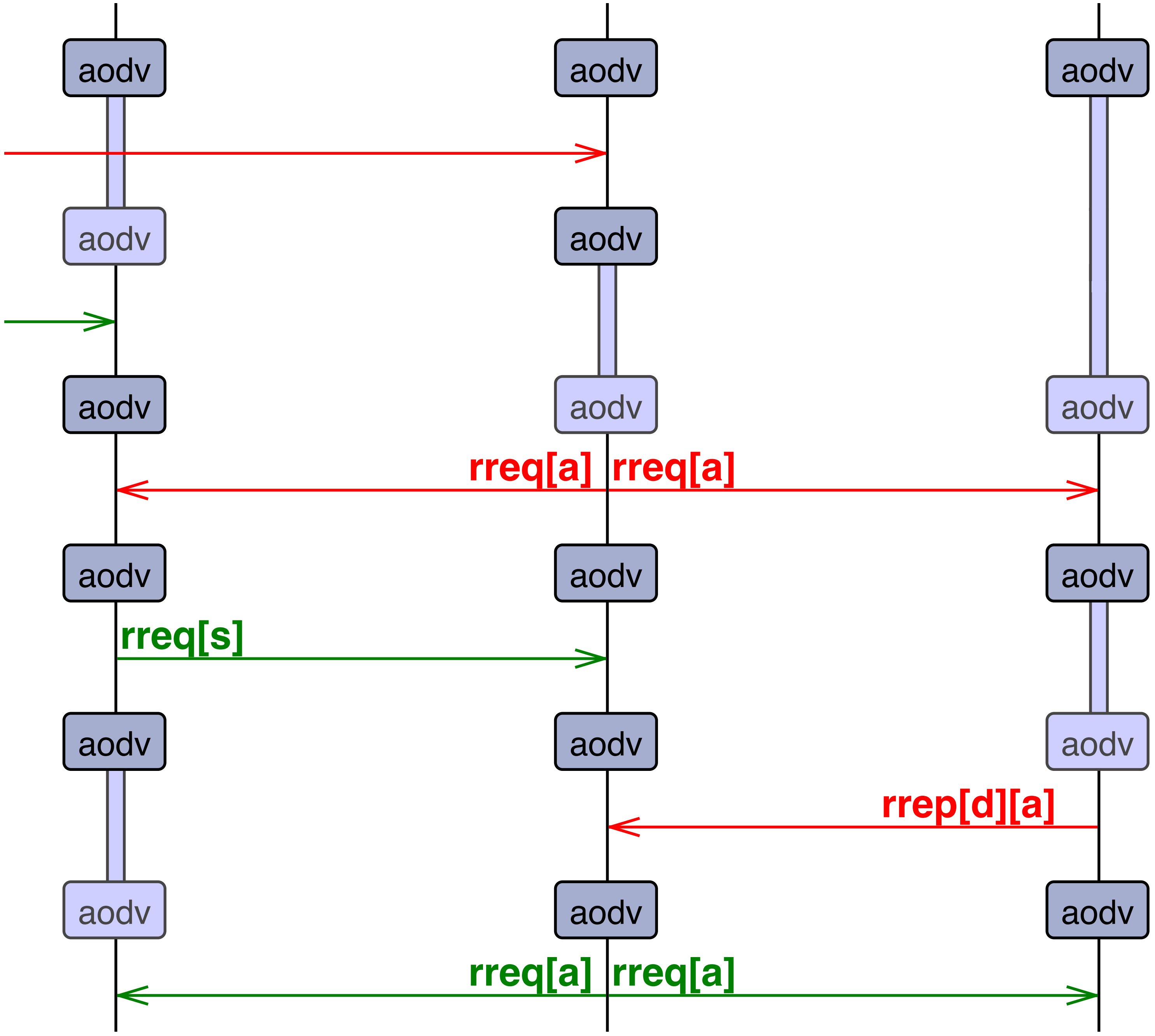
- Synchronisation mechanisms
  - binary handshake synchronisation (unicast communication)
  - broadcast synchronisation (broadcast communication)
- Common data structures
  - arrays, structs, ...
  - C-like programming language
- Provides mechanisms for time and probability

Experiments Set-Up

- Exhaustive search
  - various properties
  - all different 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

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

Routing Algebra – Elements, Operators

- Matrices over routing table entries

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

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)

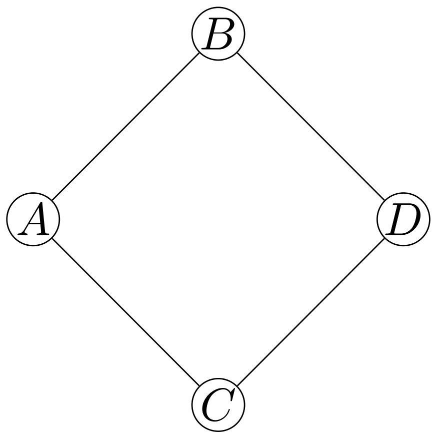
$$(\texttt{nhip}\,,\,\texttt{hops})$$

- 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



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

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

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**