Formal Verification and Simulation for Performance Analysis for Probabilistic Broadcast Protocols

Ansgar Fehnker

joint work with Peng Gao

National ICT Australia and University of New South Wales

The PEWNA project

Wireless sensor networks

Aggregate of small, portable devices

- battery-operated computing power
- gather sensor information in a distributed fashion
- wireless communications
- multi-hop communication

The PEWNA project

Project Goals

- Notations and tools for wireless network protocols.
- Model checking techniques for performance evaluation.
- Abstraction techniques to scale model checking techniques.

The Content

- Method
- Problem
- Protocol
- Results
- Future work

The Method

Model Checking

Model Checking

Basic Idea

- Given a model of the system
  - Kripke structure, FSM, Petri net, Probabilistic system…
- Given a formal specification
  - LTL, CTL, mu-calculus, PCTL, …
  - another simpler model
- Calculate whether model satisfies specification

Model Checking

Characteristics

≠ testing
Explores the behavior of a system model

≠ simulation
Explores (symbolically) all infinite behaviours

≠ theorem proving
Proves correctness automatically

Model Checking

PRISM

A Probabilistic Symbolic Model Checker (Uni Birmingham)

Supports:
- Discrete-Time Markov Chains (DTMCs)
- Continuous-Time Markov Chains (CTMCs)
- Markov Decision Processes (MDPs)

Checks:
- Probabilistic temporal logic (PCTL)
- Example: node 7 receives a message with probability > 0.8

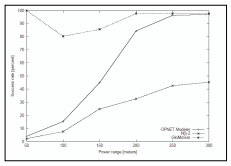
The problem

Simulation is imperfect

Related Work

On the Accuracy of MANET Simulators
Cavin, Sasson and Schiper (2002)

- Different simulators give different answers
- Even for simple protocols
- Semantics defined by simulator.
- Low level details as important as high level protocol



Related Work

Experimental Evaluation of Wireless Simulation Assumptions
Kotz, Newport et al. (2004)

- Assumptions render results useless
- Common assumptions:
  1. The earth is flat
  2. The transmission area is circular.
  3. All radios have equal range.
  4. If I can hear you, you can hear me.
  5. If I can hear you at all, I can hear you perfectly.
  6. Signal strength is a simple function of distance.

The Problem

Common Solution

- More details
- Hardware in the loop simulation
- Precise specific assumptions
- Specific results for very specific instances of a system

What about the general properties of a protocol?

The Problem

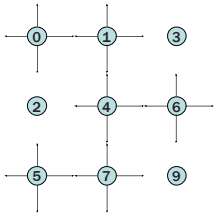
Our Approach

- More abstract model
- Formal model with well defined semantics
- Precise assumptions
- General results for a generic protocol

The Protocol

Gossiping and Flooding

Flooding and Gossiping



Gossiping protocol
- listen to medium
- if you receive a message
  - send message with probability p
- go to sleep

Properties of gossiping
- simple
- reduced redundancy
- reduced collisions
- improves efficiency

Common assumption

Common assumptions

- Absence of collisions
- Perfectly synchronous execution
- No clock drift
- Perfect medium

The basic model

Formal Model

Prism Model

- Well defined semantics,
- Abstraction from low level detail

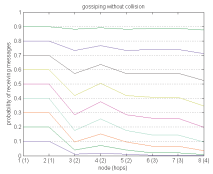
```
module node4
act4:  bool init true;
send4: bool init false;

[tick]  act4 & !send4 & (send1|send3|send5|send7)
        -> psend: (act4'=true)&(send4'=true) + (1-psend):(act4'=false)&(send4'=false);
[tick]  act4 & !send4 & !(send1|send3|send5|send7)-> (act4'=true) & (send4'=false);
[tick]  act4 &  send4 -> (act4'=false) & (send4'=false);
[tick] !act4          -> (act4'=false) & (send4'=false);
endmodule
```

PRISM model of gossiping protocol

Performance Evaluation

Model checking results

- PRISM results for gossiping w/o collision
- Results are exact probabilities rather than approximations



Common assumption

Common assumptions

- *Absence of collisions*
- **Perfectly synchronous execution**
- No clock drift
- *Perfect medium*

Synchronization

Model protocols as formal models

- Clean understanding of concurrency
- Simulation in contrast often implicitly synchronous

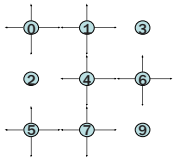
```
for i=find(sending) %for all nodes that are sending

   if(rand <= p)
        for j=1:4
            if(C(i,j) ~= 0) % if it has a neighbour
                        receive(C(i,j)) = 1; %then neighbour receives a msg
            end
        end;
   end;
end;
```

Matlab code for Monte-Carlo simulation: *implicitly synchronous*

Synchronization

Model protocols as formal models

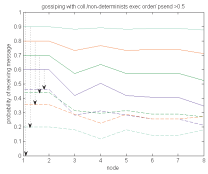
- Clean understanding of concurrency



Synchronization

Model protocols as formal models

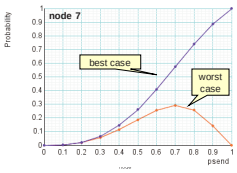
- Model with non-deterministic execution order
- Minimal and maximal probability cover all execution orders



Synchronization

Model protocols as formal models

- Model with non-deterministic execution order
- Minimal and maximal probability cover all execution orders



Common assumption

Common assumptions

- *Absence of collisions*
- Perfectly synchronous execution
- **No clock drift**
- *Perfect medium*

Timing

Unreliable timing

- Modelled as probabilistic waiting
- Compared different variants of the timing model

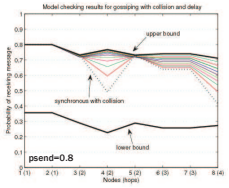
```
[tick] active4=1 & send4=0 & send1+send3+send5+send7 =1
                           ->   (1-psend):        (active4'=0)&(send4'=0)
                           +    psend*(1-pdelay): (active4'=1)&(send4'=1)
                           +    psend*   pdelay:  (active4'=2)&(send4'=0);
[tick] active4=2 ->(1-pdelay): (active4'=1)&(send4'=1)
                           +    pdelay:  (active4'=2)&(send4'=0);
```

PRISM model of simple delay

Timing

Unreliable timing

- Effect of collisions vanishes with an increasing delay
- Upper and lower bound provided by non-deterministic model



Intermediate Summary

Results

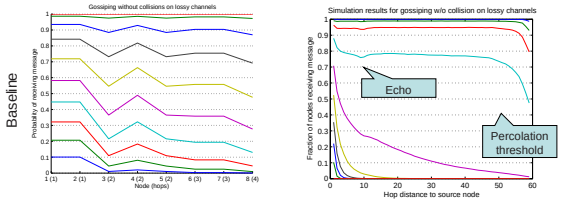
- Prism model for gossiping protocols
- Well defined semantics
- Abstraction from low level detail
- Verification produces exact answers
- Deals with non-determnism by under specification
- Results for small networks (up to 20 nodes)

What about large networks (1000 nodes)?

Matlab model

Monte-Carlo simulation

- Derived a simulation model (manually) from PRISM model
- Observed effects that are only visible for large models/models



Summary

Verification **vs** Simulation

Verification
- all infinite behaviour
- for a generic protocol
- for small networks

Simulation
- a few finite behaviours
- for a specific instance
- for large networks

Observations

Verification **and** Simulation

- Simulation complements model checking
- Formal model serves as "golden" model
- We are currently maintaining two models (artefacts)
- Future work: PRISM to of-the-shelf simulator translation

The End

Thanks