Probabilistic Model Checking of Randomised Distributed Protocols using PRISM

Marta Kwiatkowska

University of Birmingham

VPSM PhD School, Copenhagen, October 2006

Part III

Case Studies

PRISM case studies

- Communication and multimedia protocols
  - Bluetooth device discovery [DKNP06]
  - IEEE 1394 FireWire root contention [KNS03]
  - IPv4 Zeroconf protocol [KNPS06]
  - IEEE 802.3 CSMA/CD protocol [DFH+04]
  - IEEE 802.11 WiFi wireless LANs [KNS02]
  - Zigbee (IEEE 802.15.4) protocol [Fru06]

www.cs.bham.ac.uk/~dxp/prism/casestudies

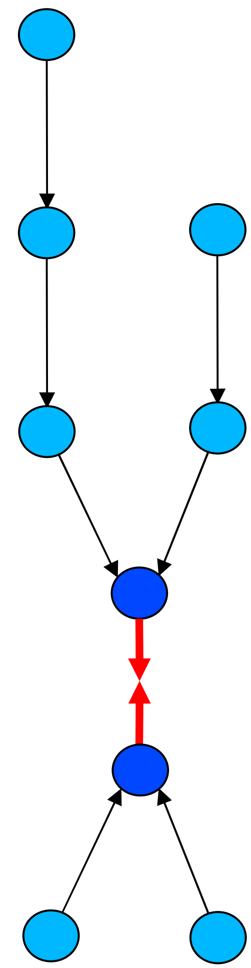
PRISM case studies

- Security systems/protocols
  - Probabilistic Contract Signing [NS06]
  - Crowds Protocol (anonymity) [Shm04]
  - Probabilistic Fair Exchange [NS06]
  - PIN Cracking Schemes [Ste06]
  - Negotation frameworks [BFW06]
  - Quantum cryptography [NPBG05]

www.cs.bham.ac.uk/~dxp/prism/casestudies

PRISM case studies

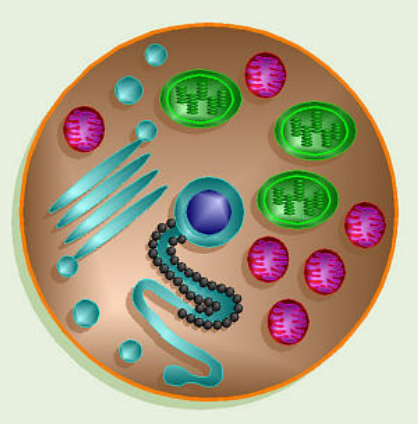
- Randomised distributed algorithms for:
  - Byzantine Agreement [KN02]
  - Consensus [KNS01]
  - Self-stabilisation
  - Leader election
  - Mutual exclusion
  - Two Process Wait-Free Test-and-Set



www.cs.bham.ac.uk/~dxp/prism/casestudies

PRISM case studies

- Analysis of behaviour/performance/reliability of:
  - Biological processes – signalling/cell cycle pathways [HKN+06]
  - Dynamic power management systems [NPK+05]
  - Dynamic voltage scaling algorithms [KNP05]
  - Manufacturing/control systems [KNP06,GF06]
  - Nanotechnology - NAND multiplexing [NPKS05]
  - Groupware protocols ("thinkteam") [BML05]



www.cs.bham.ac.uk/~dxp/prism/casestudies

Bluetooth device discovery

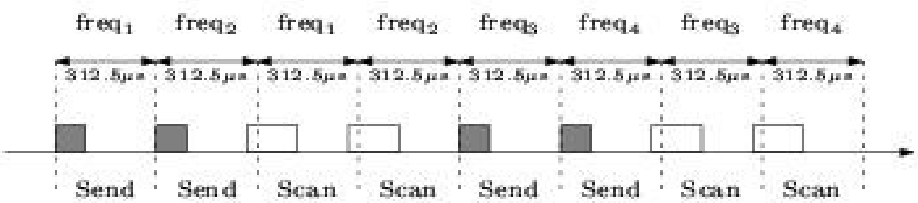
- Bluetooth: short-range low-power wireless protocol
  - widely available in phones, PDAs, laptops, ...
  - personal area networks (PANs)
  - open standard, specification freely available
- Uses frequency hopping scheme
  - to avoid interference (uses unregulated 2.4GHz band)
  - pseudo-random selection over 32 of 79 frequencies
- Network formation
  - piconets (1 master, up to 7 slaves)
  - self-configuring: devices discover themselves

Bluetooth device discovery

- States of a Bluetooth device:
  - Standby: default operational state
  - Inquiry: device discovery
    - master looks for devices, slaves listens for master
  - Page: establish connection - synchronise clocks, etc.
  - Connected: device ready to communicate in a piconet
- Device discovery
  - manadatory first step before any communication possible
  - “page” reuses information from “inquiry” so is much faster
  - power consumption much higher for “page”
  - performance crucial

Master (sender) behaviour

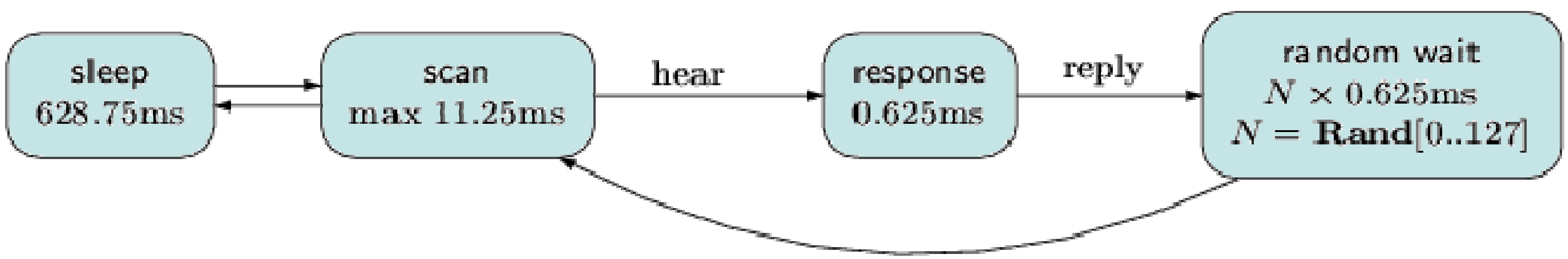
- 28 bit free-running clock CLK, ticks every 312.5μs
- Frequency hopping sequence determined by clock:
  - freq = [CLK_{16-12}+k+ ($CLK_{4-2,0}$-CLK_{16-12}) mod 16] mod 32
  - 2 trains of 16 frequencies (determined by offset k), 128 times each, swap between every 2.56s
- Broadcasts inquiry packets on two consecutive frequencies, then listens on the same two



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
17	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	2	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	2	3	20	21	22	23	24	25	26	27	28	29	30	31	32
17	18	19	20	5	6	7	8	9	10	11	12	13	14	15	16
17	18	19	20	21	6	7	8	9	10	11	12	13	14	15	16
1	2	3	4	5	6	23	24	25	26	27	28	29	30	31	32
1	2	3	4	5	6	7	24	25	26	27	28	29	30	31	32
17	18	19	20	21	22	23	24	9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24	25	10	11	12	13	14	15	16
1	2	3	4	5	6	7	8	9	10	27	28	29	30	31	32
1	2	3	4	5	6	7	8	9	10	11	28	29	30	31	32
17	18	19	20	21	22	23	24	25	26	27	28	13	14	15	16
17	18	19	20	21	22	23	24	25	26	27	28	29	14	15	16
1	2	3	4	5	6	7	8	9	10	11	12	13	14	31	32
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	32
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
17	18	3	4	5	6	7	8	9	10	11	12	13	14	15	16
17	18	19	4	5	6	7	8	9	10	11	12	13	14	15	16
1	2	3	4	21	22	23	24	25	26	27	28	29	30	31	32
1	2	3	4	5	22	23	24	25	26	27	28	29	30	31	32
17	18	19	20	21	22	7	8	9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	8	9	10	11	12	13	14	15	16
1	2	3	4	5	6	7	8	25	26	27	28	29	30	31	32
1	2	3	4	5	6	7	8	9	26	27	28	29	30	31	32
17	18	19	20	21	22	23	24	25	26	11	12	13	14	15	16
17	18	19	20	21	22	23	24	25	26	27	12	13	14	15	16
1	2	3	4	5	6	7	8	9	10	11	12	29	30	31	32
1	2	3	4	5	6	7	8	9	10	11	12	13	30	31	32
17	18	19	20	21	22	23	24	25	26	27	28	29	30	15	16
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	16

Slave (receiver) behaviour

- Listens (scans) on frequencies for inquiry packets
  - must listen on right frequency at right time
  - cycles through frequency sequence at much slower speed (every 1.28s)



- On hearing packet, pause, send reply and then wait for a random delay before listening for subsequent packets
  - avoid repeated collisions with other slaves

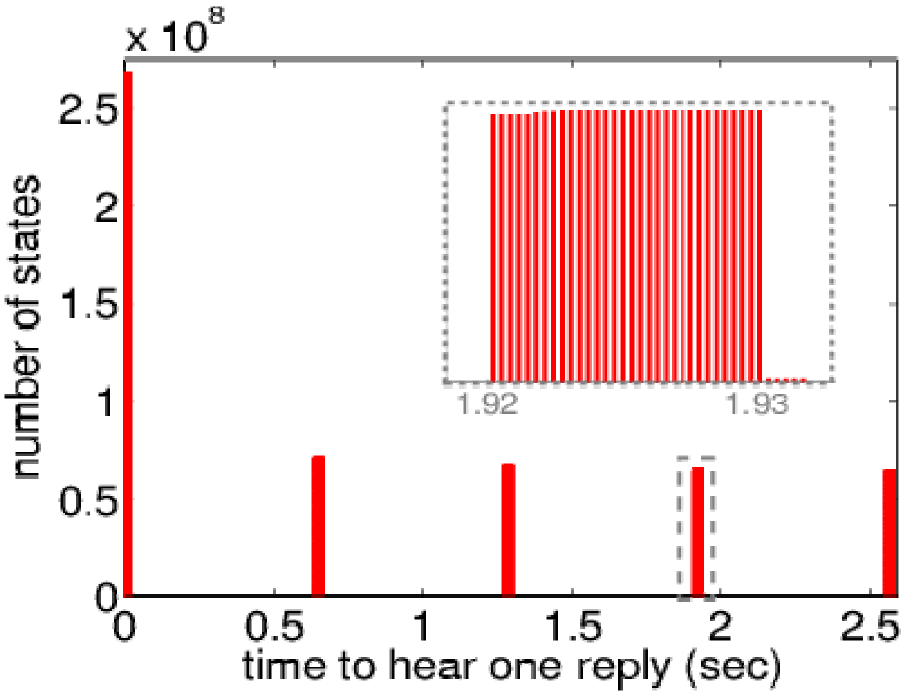
Bluetooth – PRISM model

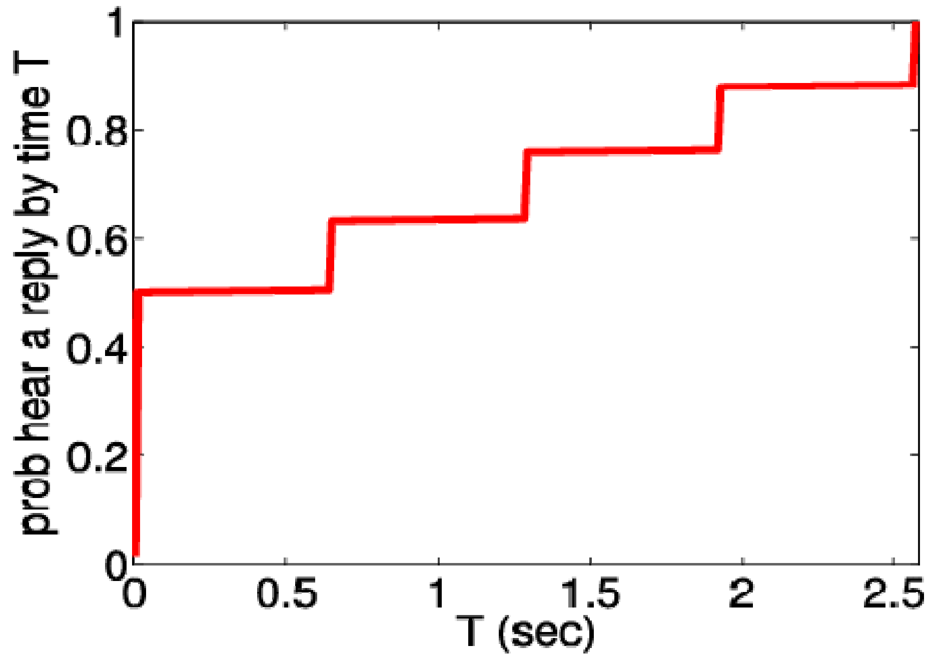
- Modelling in PRISM [DKNP06]
  - model one sender and one receiver
  - synchronous (clock speed defined by Bluetooth spec)
  - randomised behaviour – use DTMC
  - model at lowest-level (one clock-tick = one transition)
  - use real values for delays, etc. from Bluetooth spec
- Modelling challenges
  - complex interaction between sender/receiver
  - combination of short/long time-scales – cannot scale down
  - sender/receiver not initially synchronised, huge number of possible initial configurations (17,179,869,184)

Bluetooth - Results

- Huge DTMC – initially, model checking infeasible
  - partition into 32 scenarios, i.e. 32 separate DTMCs
  - on average, approx. 3.4×10^9 states, 536,870,912 initial
  - can be built/analysed with PRISM's MTBDD engine
- Compute:
  - R=? [F replies=K {"init"}{max}]
  - "worst-case expected time to hear K replies over all possible initial configurations"
  - also look at:
    - how many initial states for each possible expected time
    - cumulative distribution function assuming equal probability for each initial state

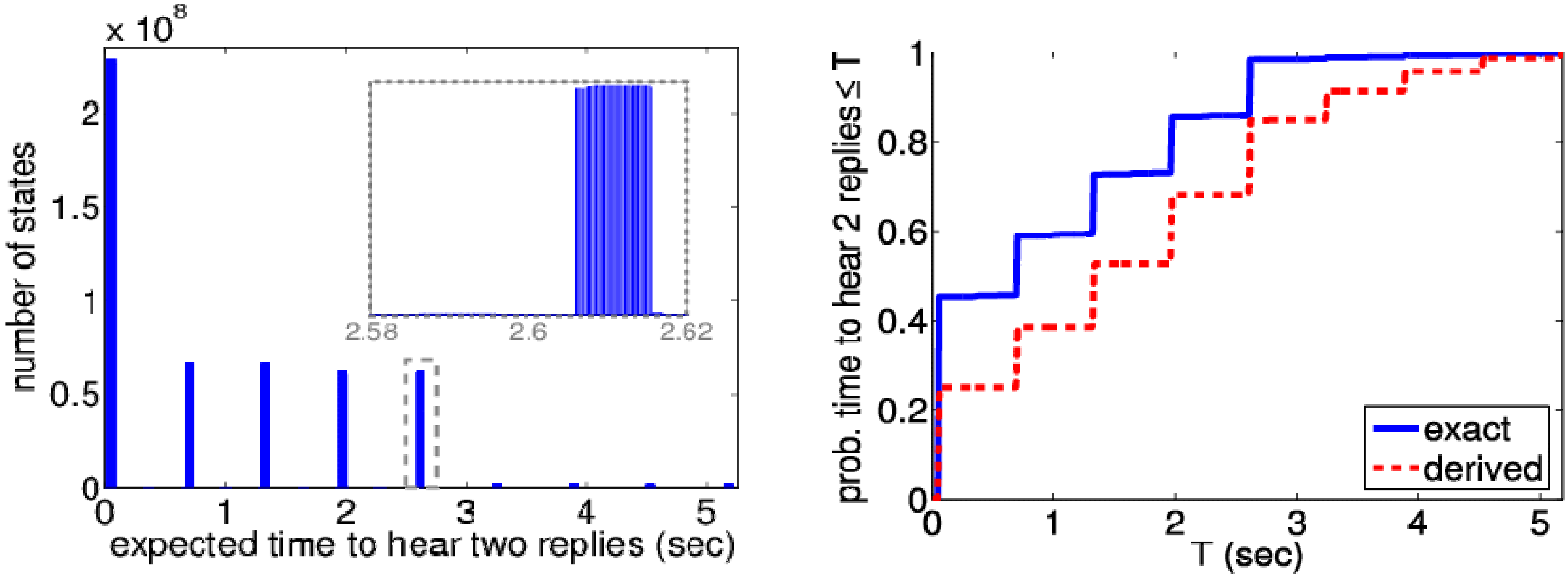
Bluetooth - Time to hear 1 reply





- worst-case expected time = 2.5716 sec
- in 921,600 possible initial states
- best-case = 635 μs

Bluetooth - Time to hear 2 replies



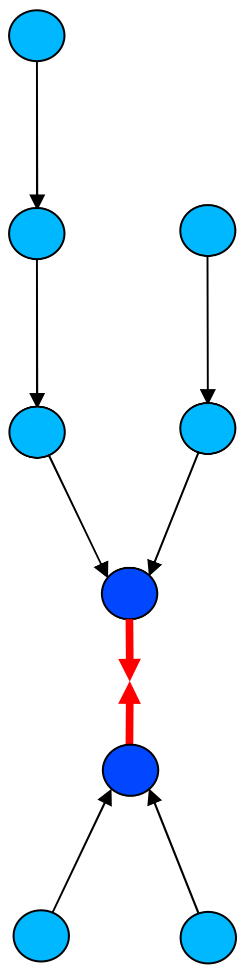
- worst-case expected time = 5.177 sec
- in 444 possible initial states
- compare actual CDF with derived version which assumes times to reply to first/second messages are independent

Bluetooth - Results

- Other results: (see [DKNP06])
  - compare versions 1.2 and 1.1 of Bluetooth, confirm 1.1 slower
  - power consumption analysis (using costs + rewards)
- Conclusions:
  - successful analysis of complex real-life model, actual parameters
  - exhaustive analysis: best-/worst-case values
    - can pinpoint scenarios which give rise to them
    - not possible with simulation approaches
  - model still relatively simple
    - consider multiple receivers?
    - combine with simulation?

IEEE 1394 (FireWire) root contention

- Serial bus for networking multimedia devices
  - "hot-pluggable" - add/remove devices (nodes) at any time
- Root contention protocol
  - leader election algorithm, when nodes join/leave
  - nodes send messages: "be my parent"
  - root contention: when nodes contend leadership
  - random choice: "fast"/"slow" delay before retry
- Properties of interest
  - time taken for leader election
  - effect of using biased coin - conjecture [Stoelinga]



FireWire - PRISM model

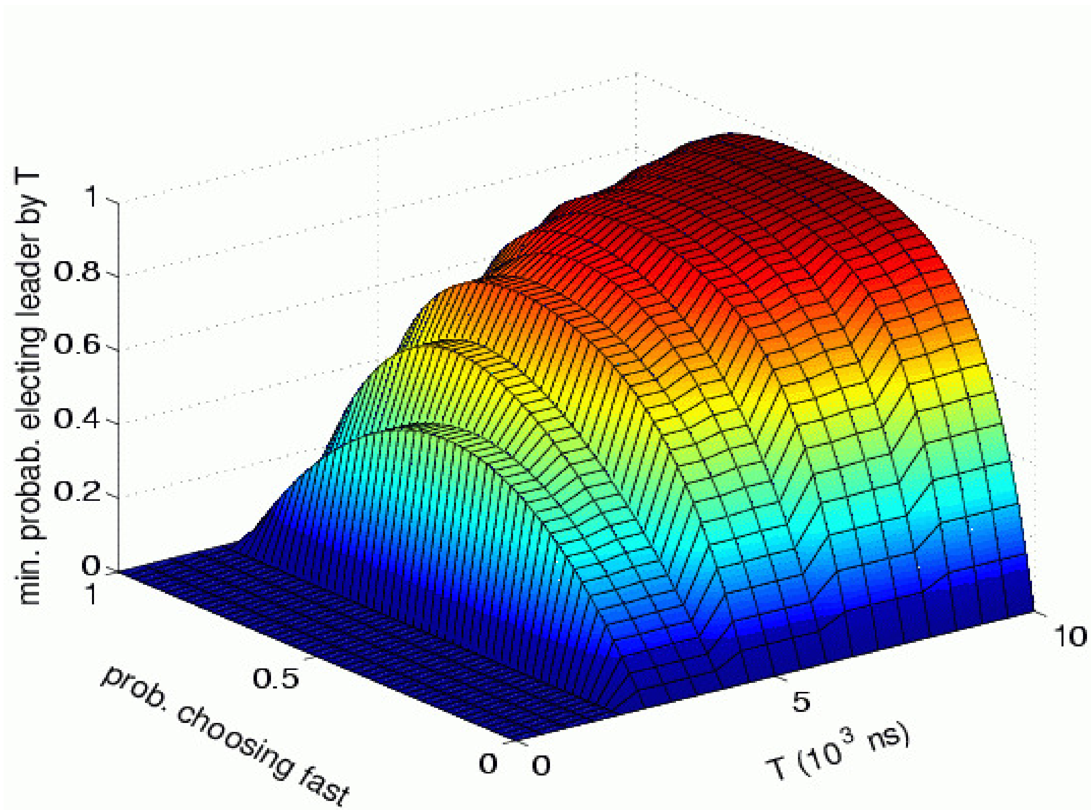
- Based on probabilistic timed automata (PTA) model
  - by Stoelinga et al. [SV99], [SS01]
  - infinite state (real-time)
  - digital clocks approach [KNS03] reduces to…

- PRISM model: finite-state MDP
  - concurrency: messages between nodes and wires
  - underspecification of delays (upper/lower bounds)
  - probability: coin toss
  - max. model size: 170 million states
  - analysed using PRISM's MTBDD engine

FireWire - Properties

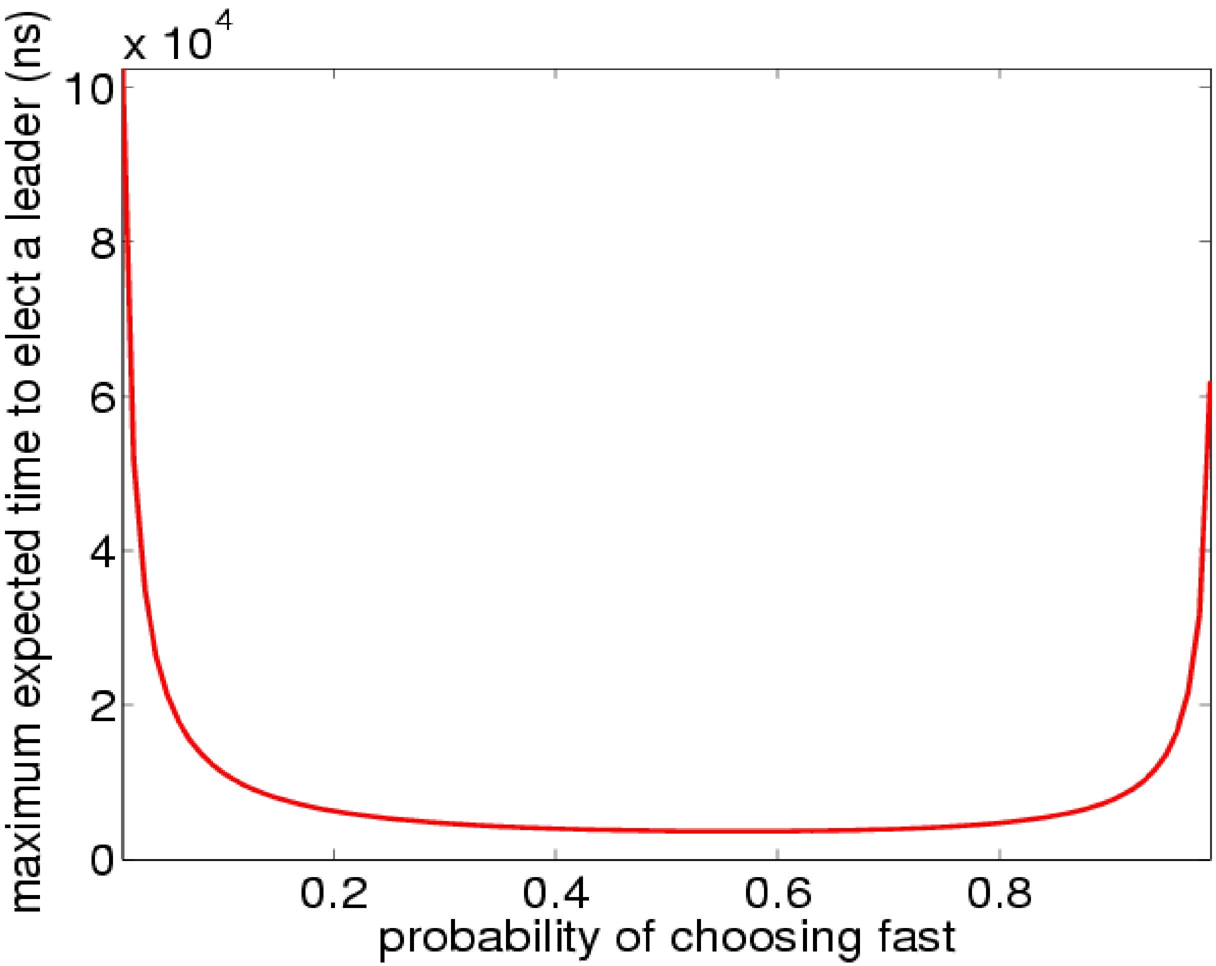
- "minimum probability that a leader is elected by time T"
  - add variable t to count elapsed time
  - Pmin=? [$t \leq T$ U "elected"]
  - vary: T, coin bias: probability of choosing "fast"
- "maximum expected time to elect a leader"
  - add timing costs
  - Rmax=? [F "elected"]
  - vary: coin bias

FireWire - Results



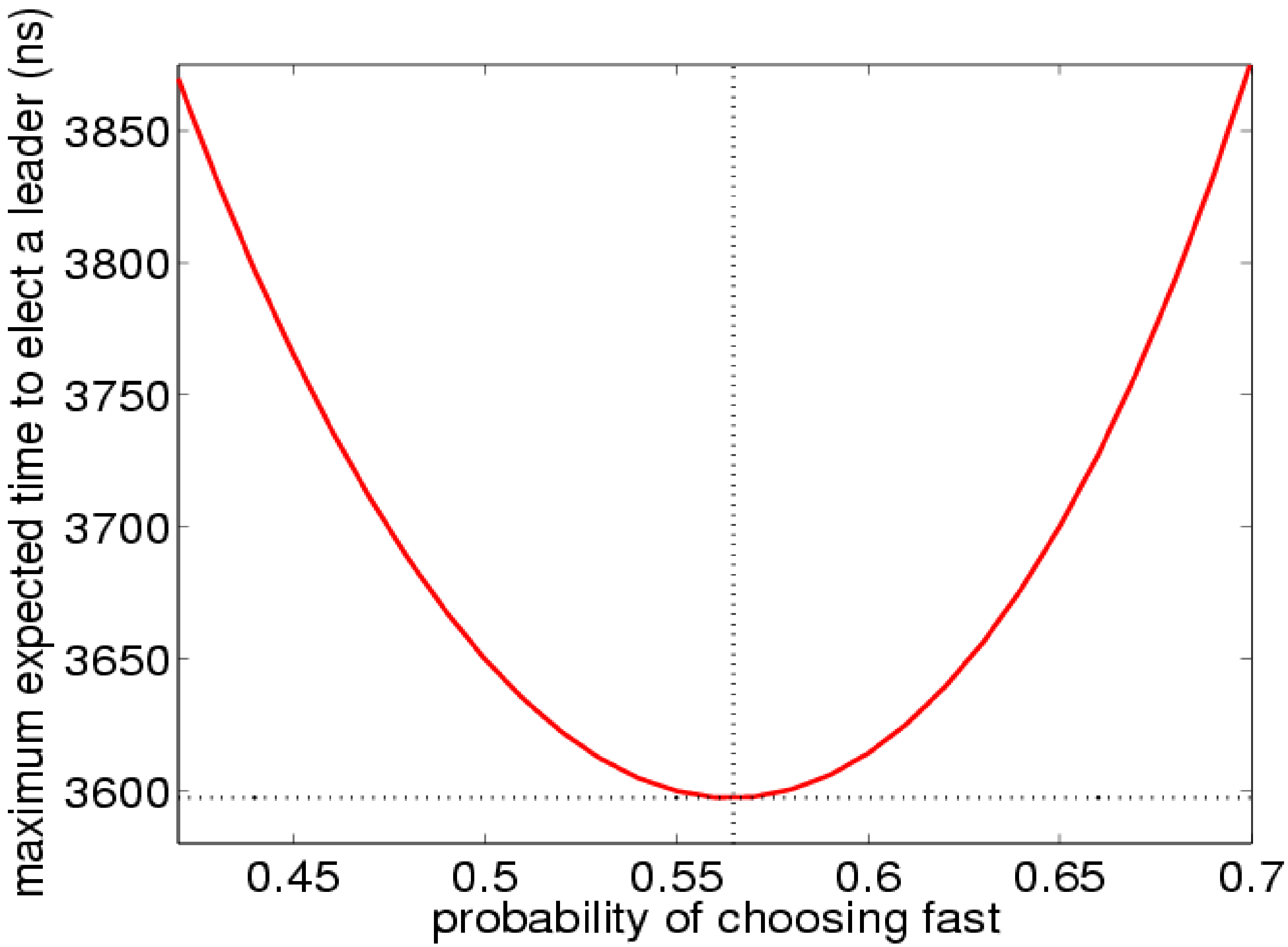
"minimum probability of electing leader by time T"

FireWire - Results



"maximum expected time to elect a leader"

FireWire - Results



"maximum expected time to elect a leader"

Biased coin is beneficial

Contract signing

- Two parties want to agree on a contract
  - each will sign if the other will sign, but do not trust each other
  - there may be a trusted third party (judge)

    but it should only be used if something goes wrong
- In real life: contract signing with pen and paper
  - sit down and write signatures simultaneously
- On the Internet…
  - how to exchange commitments on an asynchronous network?
  - "partial secret exchange protocol" due to

    Even, Goldreich and Lempel [EGL85]

Contract signing – EGL protocol

- Partial secret exchange protocol for 2 parties (**A** and **B**)
- **A** (**B**) holds 2N secrets $a_1,\ldots,a_{2N}$ ($b_1,\ldots,b_{2N}$)
  - a secret is a binary string of length L
  - secrets partitioned into pairs: e.g. $\{(a_i, a_{N+i}) \mid i=1,\ldots,N\}$
  - **A** (**B**) committed if **B** (**A**) knows one of **A**'s (**B**'s) pairs
- Uses "1-out-of-2 oblivious transfer protocol" **OT(S,R,x,y)**
  - **S** sends **x** and **y** to **R**
  - **R** receives **x** with probability ½ otherwise receives **y**
  - **S** does not know which one **R** receives
  - if **S** cheats then **R** can detect this with probability ½

Contract signing – EGL protocol

(step 1)

for (i=1,...,N)

OT(A,B,a_i,a_{N+i})

OT(B,A,b_i,b_{N+i})

(step 2)

for (i=1,...,L) (where **L** is the bit length of the secrets)

for (j=1,...,2N)

A transmits bit **i** of secret a_j to **B**

for (j=1,...,2N)

B transmits bit **i** of secret b_j to **A**

Contract signing - Results

- Modelled in PRISM as a DTMC (no concurrency) [NS06]
- Discovered a weakness in the protocol:
  - party **B** can act maliciously by quitting the protocol early
  - this behaviour not considered in the original analysis
- More details:
  - if **B** stops participating in the protocol as soon as he/she has obtained at least one of **A** pairs, then, with probability 1, at this point:
    - **B** possesses a pair of **A**'s secrets
    - **A** does not have complete knowledge of any pair of **B**'s secrets
  - Protocol is therefore not fair under this attack:
    - **B** has a distinct advantage over **A**

Contract signing - Results

- The protocol is unfair because in step 2: **A** sends a bit for each of its secret before **B** does.
- Can we make this protocol fair by changing the message sequence scheme?
- Since the protocol is asynchronous the best we can hope for is with probability ½ **B** (or **A**) gains this advantage
- We consider 3 possible alternate message sequence schemes…

Contract signing: EGL2

(step 1)

…

(step 2)

for (i=1,…,L)

for (j=1,…,N) A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

for (j=1,…,N) B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

for (j=N+1,…,2N) A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

for (j=N+1,…,2N) B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

Contract signing: EGL3

(step 1)

…

(step 2)

for (i=1,…,L) for (j=1,…,N)

A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

for (i=1,…,L) for (j=N+1,…,2N)

A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

Contract signing: EGL4

(step 1)

…

(step 2)

for (i=1,…,L)

A transmits bit **i** of secret $\mathbf{a_1}$ to **B**

for (j=1,…,N) B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

for (j=2,…,N) A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

for (i=1,…,L)

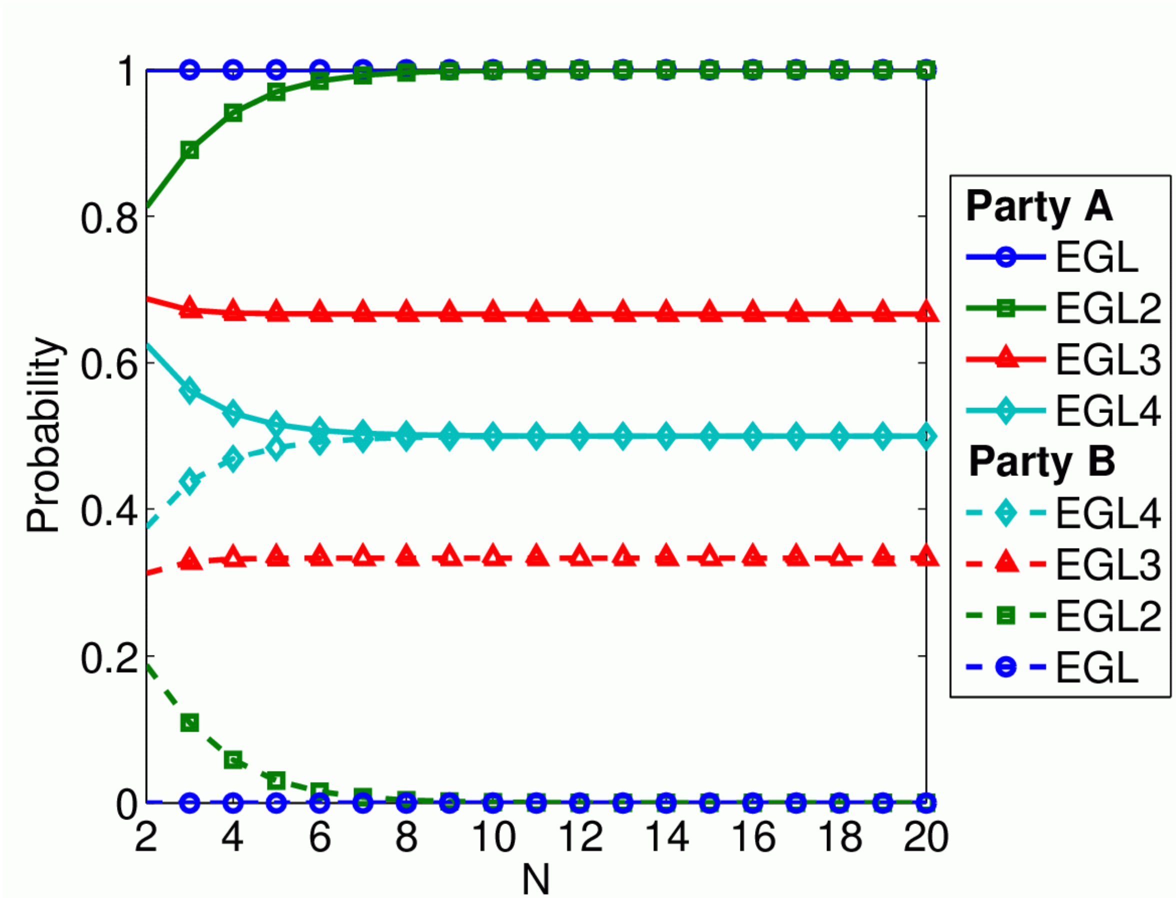
A transmits bit **i** of secret $\mathbf{a_{N+1}}$ to **B**

for (j=N+1,…,2N) B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

for (j=N+2,…,2N) A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

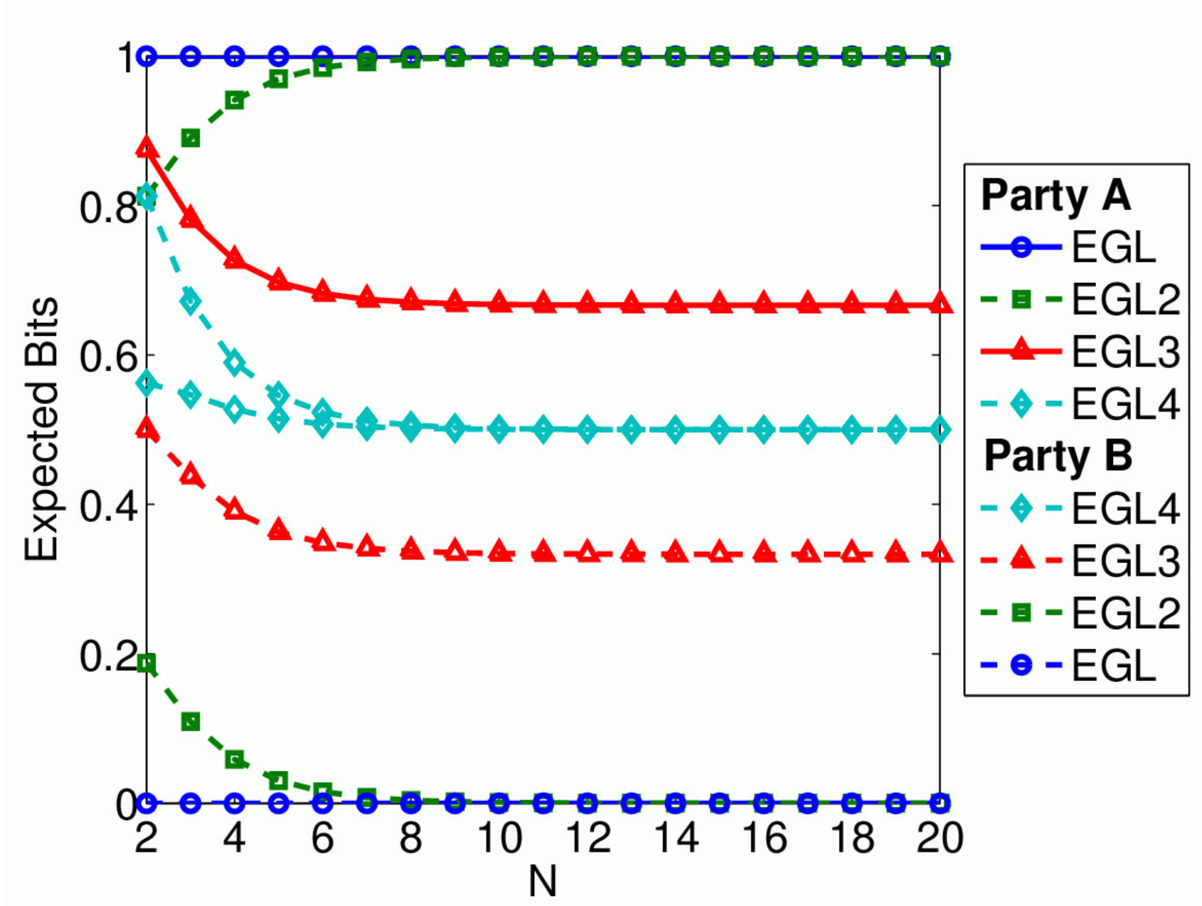
Contract signing - Results

- Probability that the other party gains knowledge first (the chance that the protocol is unfair)



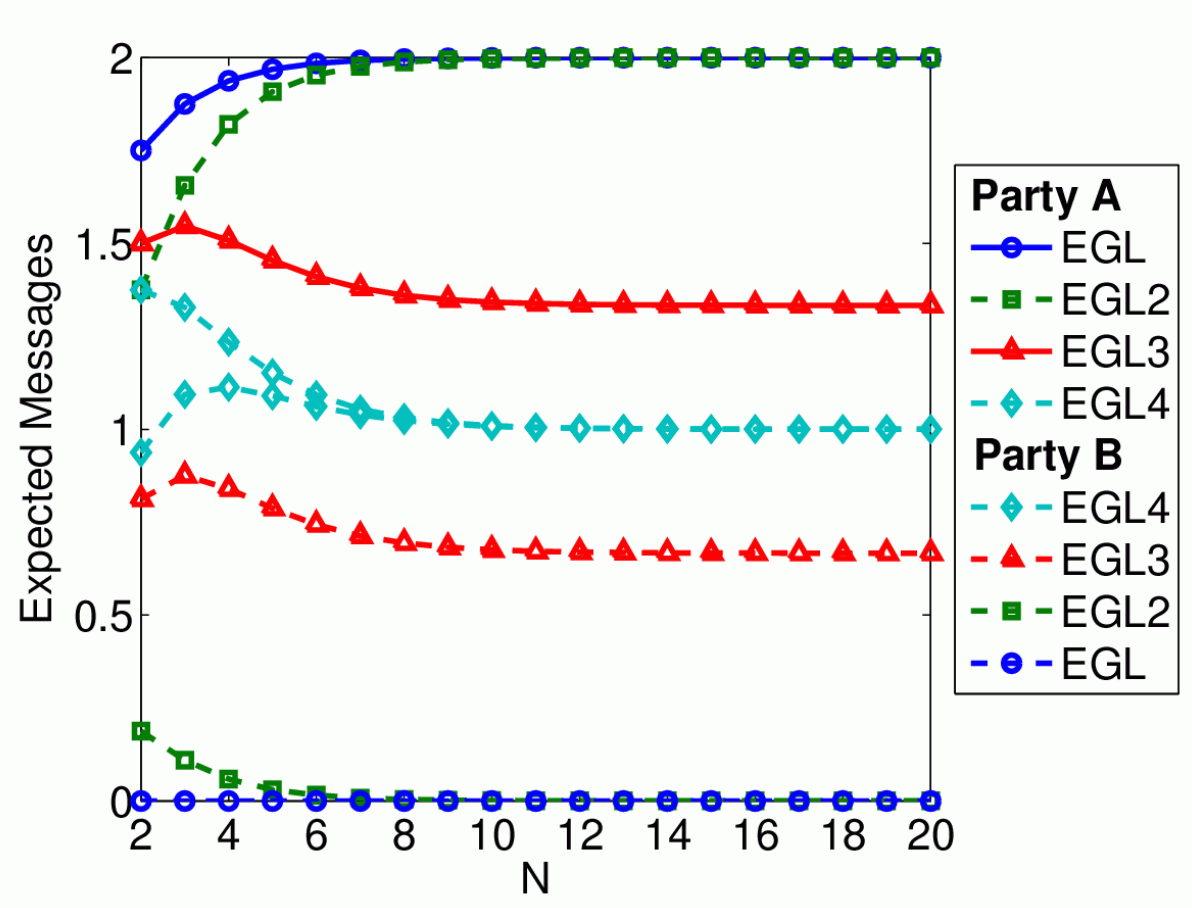
Contract signing - Results

- Expected bits a party requires to know a pair once the other knows a pair (quantifies how unfair the protocol is)



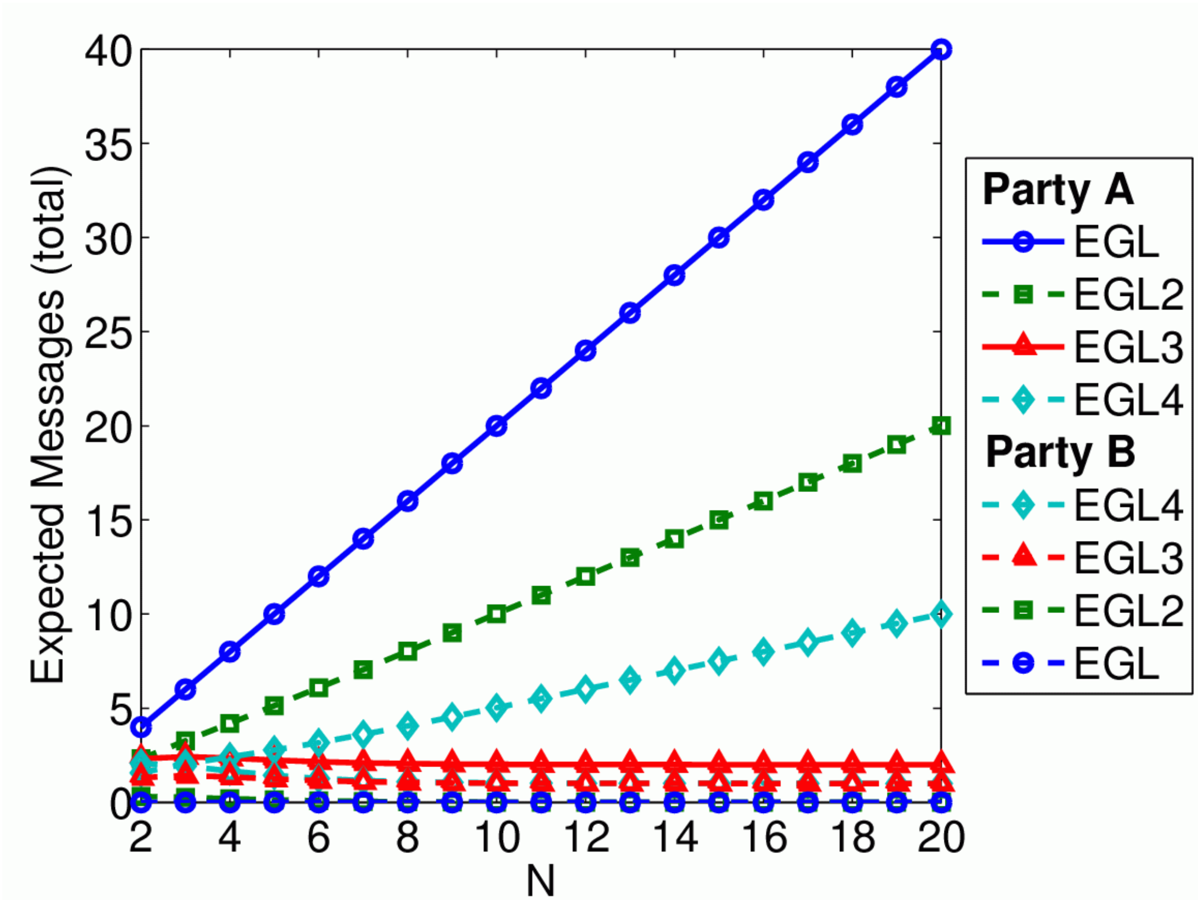
Contract signing - Results

- Expected messages a party must receive to know a pair once the other knows a pair (measures the influence the other party has on the fairness, since it can try and delay these messages)



Contract signing - Results

- Expected messages that need to be sent for a party to know a pair once the other party knows a pair (measures the duration of unfairness)



Contract signing - Results

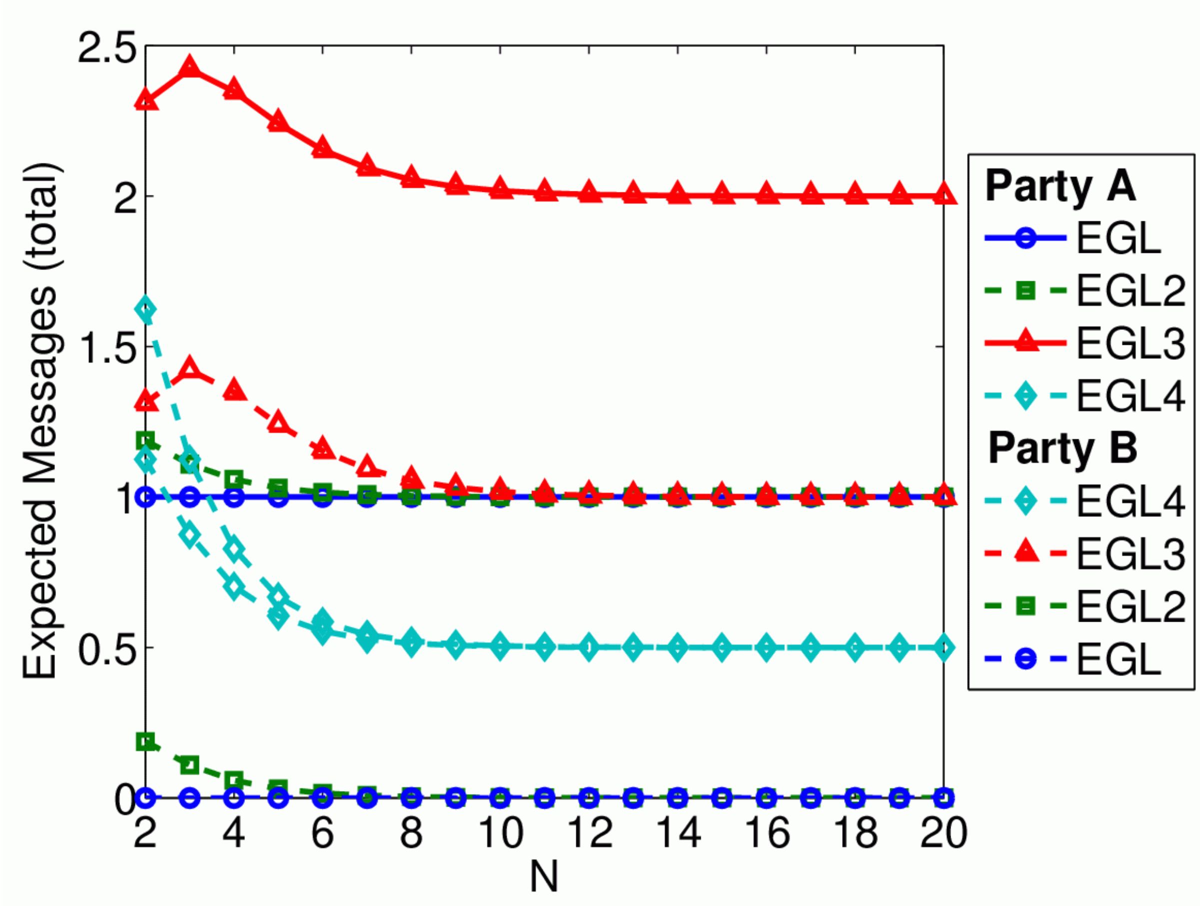
- Results show EGL4 is the ‘fairest’ protocol
- Except for duration of fairness measure:

  Expected messages that need to be sent for a party to know a pair once the other party knows a pair

  - this value is larger for **B** than for **A**
  - in fact, as **N** increases, it increases for **B**, decreases for **A**
- Solution: if a party sends a sequence of bits in a row (without the other party sending messages in between), require that the party send these bits as as a single message

Contract signing - Results

- Expected messages that need to be sent for a party to know a pair once the other party knows a pair (measures the duration of unfairness)



IPv4 Zeroconf protocol

- IPv4 ZeroConf protocol
  - New IETF standard for dynamic network self-configuration
  - Link-local (no routers within the interface)
  - No need for an active DHCP server
  - Aimed at home networks, wireless ad-hoc networks, hand-held devices
  - “Plug and play”
- Self-configuration
  - Performs assignment of IP addresses
  - Symmetric, distributed protocol
  - Uses random choice and timing delays

IPv4 Zeroconf Standard

- Select an IP address out of 65024 at random
- Send a probe querying if address in use, and listen for 2 seconds
  - If positive reply received, restart
  - Otherwise, continue sending probes and listening (2 seconds)
- If K probes sent with no reply, start using the IP number
  - Send 2 packets, at 2 second intervals, asserting IP address is being used
  - If a conflicting assertion received, either:
    - defend (send another asserting packet)
    - defer (stop using the IP address and restart)

Will it work?

- Possible problem…
  - IP number chosen may be already in use, but:
    - Probes or replies may get lost or delayed (host too busy)
- Issues:
  - Self-configuration delays may become unacceptable
    - Would you wait 8 seconds to self-configure your PDA?
  - No justification for parameters
    - for example K=4 in the standard
- Case studies:
  - DTMC and Markov reward models, analytical [BvdSHV03,AK03]
  - TA model using UPPAAL [ZV02]
  - PTA model with digital clocks using PRISM [KNPS06]

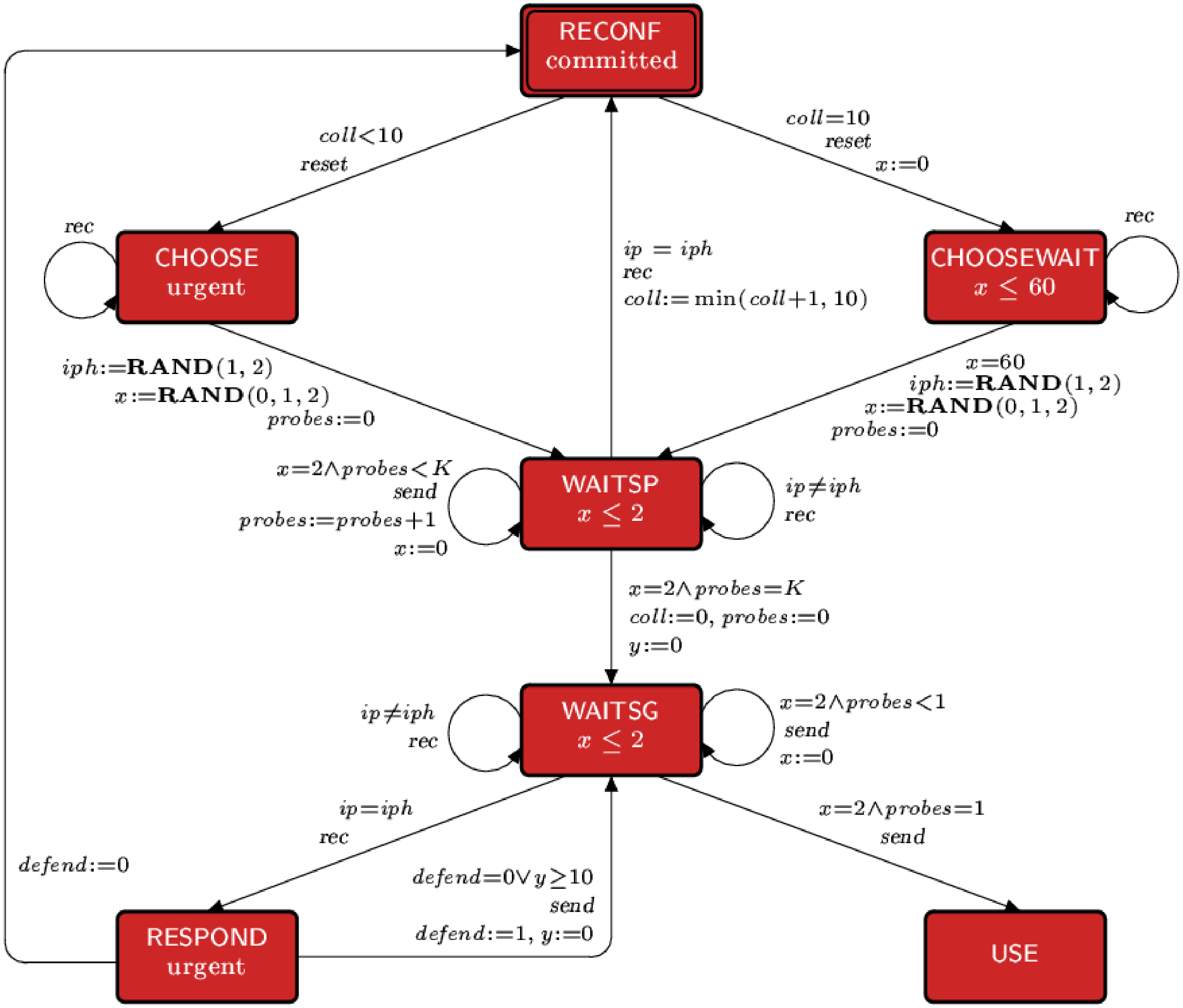
The IPv4 Zeroconf protocol model

- Modelled using Probabilistic Timed Automata (with digital clocks)

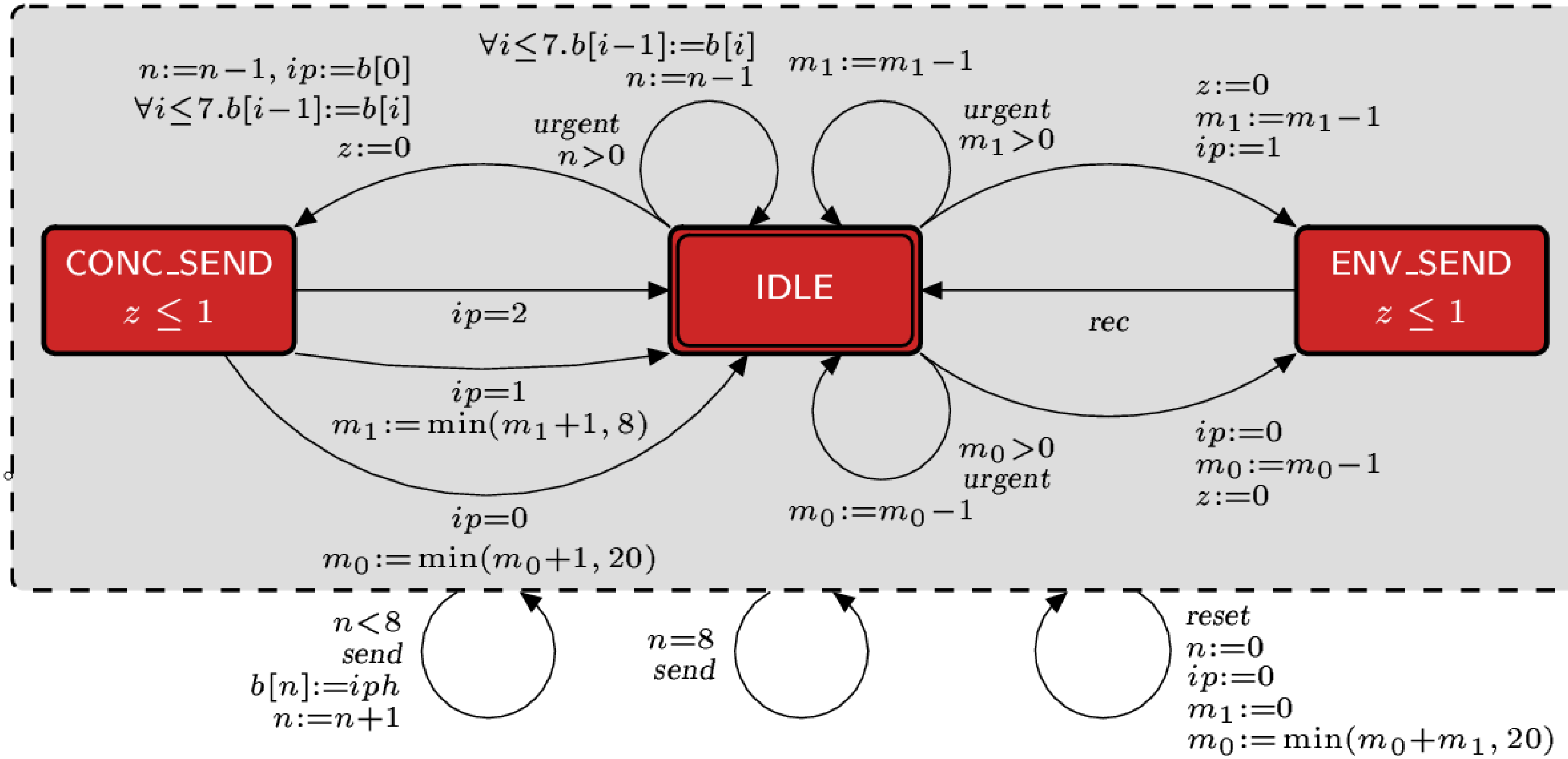
- Parallel composition of two PTAs:
  - one (joining) host, modelled in detail
  - environment (communication medium + other hosts)

- Variables:
  - K (number of probes sent before the IP address is used)
  - the probability of message loss
  - the number of other hosts already in the network

Modelling the host



Modelling the environment

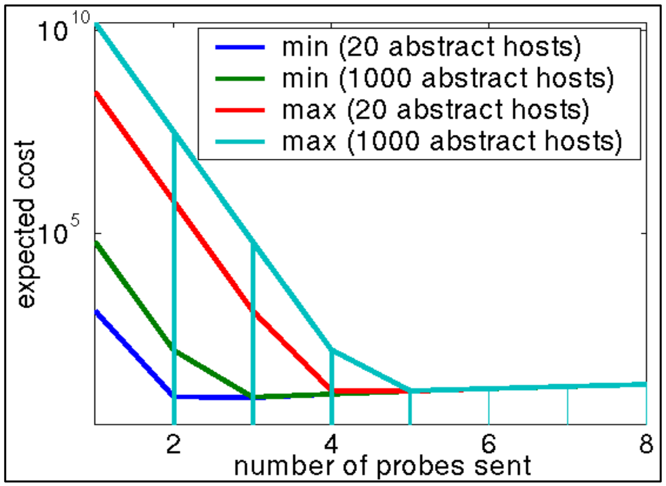


Expected costs

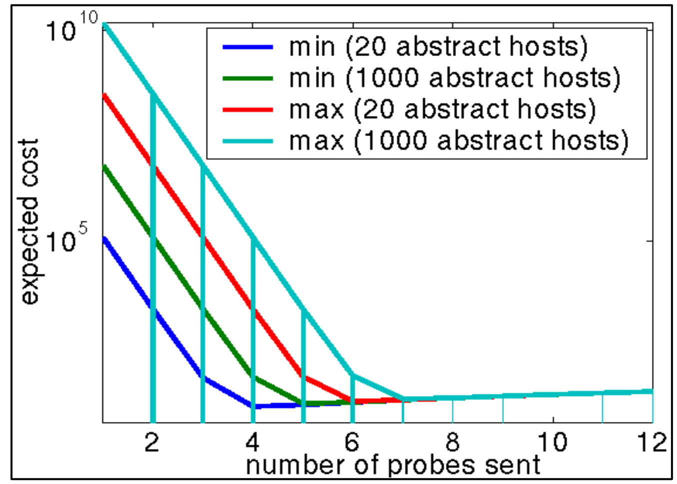
- Compute minimum/maximum expected cost accumulated before obtaining a valid IP address?
- Costs:
  - Time should be costly: the host should obtain a valid IP address as soon as possible
  - Using an IP address that is already in use should be very costly: minimise probability of error
- Cost pair: (r,e)
  - r=1 (t time units elapsing corresponds to a cost of t)
  - $e=10^{12}$ for the event corresponding to using an address which is already in use
  - e=0 for all other events

Results for IPv4 Zeroconf

Prob. of message loss = 0.001



Prob. of message loss = 0.01



- Sending a high number of probes increases the cost
  - increases delay before a fresh IP address can be used
- Sending a low number of probes increases the cost
  - increases probability of using an IP address already in use
- Similar results to the simpler model of [BvdSHV03]

References

- **[BFW06]** P. Ballarini, M. Fisher and M. Wooldridge. Automated Game Analysis via Probabilistic Model Checking: A Case Study. In Proc. 3rd Workshop on Model Checking and Artificial Intelligence (MoChArt'05), volume 149 of ENTCS, pages 125-137, Elsevier. 2006.
- **[BML05]** M. ter Beek, M. Massink and D. Latella. Towards Model Checking Stochastic Aspects of the thinkteam User Interface. In M. Harrison (editor) *Proc. 12th International Workshop on Design, Specification and Verification of Interactive Systems (DSVIS'05)*, volume 3941 of Lecture Notes in Computer Science, pages 39-50, Springer. 2005.
- **[DFH+04]** M. Duflot, L. Fribourg, T. Hérault, R. Lassaigne, F. Magniette, S. Messika, S. Peyronnet and C. Picaronny. Probabilistic model checking of the CSMA/CD protocol using PRISM and APMC. In Proc. 4th Workshop on Automated Verification of Critical Systems (AVoCS'04), volume 128(6) of ENTCS, pages 195-214, Elsevier Science. 2004.
- **[DKNP06]** M. Duflot, M. Kwiatkowska, G. Norman and D. Parker. A Formal Analysis of Bluetooth Device Discovery. *International Journal on Software Tools for Technology Transfer (STTT)*. To appear. 2006.

References

- **[EGL85]** S. Even, O. Goldreich and A. Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28(6):637-647, 1985
- **[Fru06]** M. Fruth. Probabilistic Model Checking of Contention Resolution in the IEEE 802.15.4 Low-Rate Wireless Personal Area Network Protocol. In Proc. 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISOLA'06). To appear. 2006.
- **[GF06]** J. Greifeneder and J. Frey. Dependability analysis of networked automation systems by probabilistic delay time analysis. In *Proc. 12th IFAC Symposium on Information Control Problems in Manufacturing (INCOM'06),* pages 269-274. May 2006.
- **[HKN+06]** J. Heath, M. Kwiatkowska, G. Norman, D. Parker and O. Tymchyshyn. Probabilistic model checking of complex biological pathways. In C. Priami (editor) *Proc. Computational Methods in Systems Biology (CMSB'06)*, volume 4210 of Lecture Notes in Lecture Notes in Bioinformatics, pages 32-47, Springer Verlag. 2006.

References

- **[KN02]** M. Kwiatkowska and G. Norman. Verifying Randomized Byzantine Agreement. In *Proc. Formal Techniques for Networked and Distributed Systems (FORTE'02)*, volume 2529 of LNCS, pages 194-209, Springer-Verlag. November 2002.
- **[KNP05]** M. Kwiatkowska, G. Norman and D. Parker. Probabilistic Model Checking and Power-Aware Computing. In *In Proc. 7th International Workshop on Performability Modeling of Computer and Communication Systems (PMCCS)*, pages 6-9. September 2005.
- **[KNP06]** M. Kwiatkowska, G. Norman and D. Parker. Controller Dependability Analysis By Probabilistic Model Checking. *Control Engineering Practice*, Elsevier. To appear. 2006.
- **[KNPS06]** M. Kwiatkowska, G. Norman, D. Parker and J. Sproston. Performance Analysis of Probabilistic Timed Automata using Digital Clocks. *Formal Methods in System Design*, 29, pages 33-78, Springer. August 2006.

References

- **[KNS01]** M. Kwiatkowska, G. Norman and R. Segala. Automated Verification of a Randomised Distributed Consensus Protocol Using Cadence SMV and PRISM. In *Proc. CAV'01*, volume 2102 of LNCS, pages 194-206, Springer-Verlag. 2001.
- **[KNS02]** M. Kwiatkowska, G. Norman and J. Sproston. Probabilistic Model Checking of the IEEE 802.11 Wireless Local Area Network Protocol. In Proc. PAPM/PROBMIV'02, volume 2399 of LNCS, pages 169-187. 2002.
- **[KNS03]** M. Kwiatkowska, G. Norman and J. Sproston. Probabilistic Model Checking of Deadline Properties in the IEEE1394 FireWire Root Contention Protocol. *Formal Aspects of Computing*, 14(3), pages 295-318. April 2003.

References

- **[NPBG05]** R. Nagarajan, N. Papanikolaou, G. Bowen and S. Gay. An Automated Analysis of the Security of Quantum Key Distribution. In *Proc. 3rd International Workshop on Security Issues in Concurrency (SecCo'05)*. 2005.
- **[NPKS05]** G. Norman, D. Parker, M. Kwiatkowska and S. Shukla. Evaluating the Reliability of NAND Multiplexing with PRISM. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(10), pages 1629-1637. October 2005.
- **[NPK+05]** G. Norman, D. Parker, M. Kwiatkowska, S. Shukla and R. Gupta. Using Probabilistic Model Checking for Dynamic Power Management. *Formal Aspects of Computing*, 17(2), pages 160-176, Springer-Verlag. August 2005.
- **[NS06]** G. Norman and V. Shmatikov. Analysis of Probabilistic Contract Signing. *Journal of Computer Security*. To appear. 2006.

References

- **[Shm04]** V. Shmatikov. Probabilistic Model Checking of an Anonymity System. *Journal of Computer Security*, 12(3/4), pages 355-377. 2004.
- **[SS01]** D. Simons. and M. Stoelinga. Mechanical verification of the IEEE1394a root contention protocol using Uppaal2k. *International Journal on Software Tools for Technology Transfer* 3(4):469-485, 2001.
- **[Ste06]** G. Steel. Formal Analysis of PIN Block Attacks. *Theoretical Computer Science*. To appear. 2006.
- **[SV99]** M. Stoelinga and F. Vaandrager. Root contention in IEEE 1394. In *Proc. 5th AMAST Workshop on Real-Time and Probabilistic Systems (ARTS'99)*, pp. 53-74, 1999.

Further Reading

- **[DKN+06]** M. Duflot, M. Kwiatkowska, G. Norman, D. Parker, S. Peyronnet, C. Picaronny and J. Sproston. Practical Applications of Probabilistic Model Checking to Communication Protocols. In *Handbook of Formal Methods in Industrial Critical Systems (FMICS)*. To appear. 2006.
- **[Nor04]** G. Norman. Analysing Randomized Distributed Algorithms. In *Validation of Stochastic Systems: A Guide to Current Research*, volume 2925 of Lecture Notes in Computer Science. August 2004.
- **[KNP05d]** M. Kwiatkowska, G. Norman and D. Parker. Quantitative analysis with the probabilistic model checker PRISM. *Electronic Notes in Theoretical Computer Science*, 153(2), pages 5-31, Elsevier. May 2005.
- **[KNP05b]** M. Kwiatkowska, G. Norman and D. Parker. Probabilistic model checking in practice: Case studies with PRISM. *ACM Performance Evaluation Review*, 32(4), pages 16-21. March 2005.