Statistical Model Checking for Distributed Probabilistic-Control Hybrid Automata with Smart Grid Applications

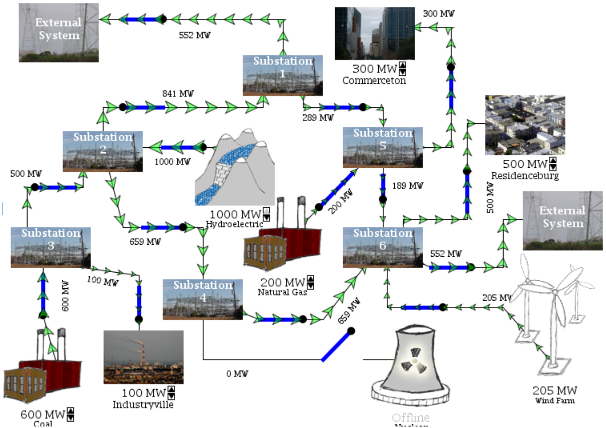
João Martins[1,2] André Platzer[1] João Leite[2]

[1] Computer Science Department,
Carnegie Mellon University, Pittsburgh PA

[2] CENTRIA and Departamento de Informática,
FCT, Universidade Nova de Lisboa

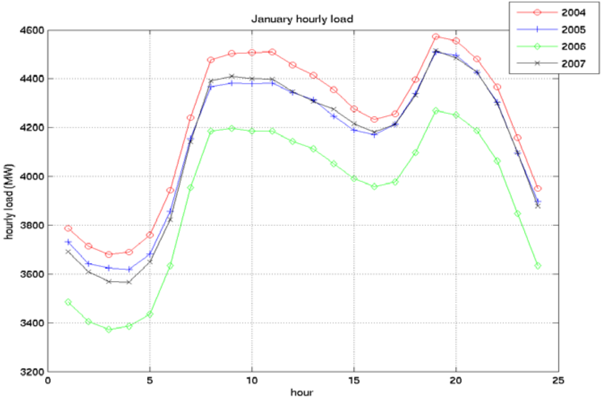
13th International Conference on Formal Engineering Methods

Summary



The Grid is a hierarchical "graph" with sources and sinks

Image from the TCIPG Education applet



Power consumption follows well-known patterns

Image from The Impact of Daylight Savings Time on Electricity Consumption in Indiana, J. Basconi, J. Kantor

- Smart Meters + Smart Appliances
- The Grid predicts load, becomes more stable, cost-effective, energy-efficient, secure, resilient

Even today, utilities deploy networks that transmit several thousands of bits... **per day**.

Even today, utilities deploy networks that transmit several thousands of bits... **per day**.

- Is reliability the most significant factor for the Grid?
- How about bandwidth? RTT?

Even today, utilities deploy networks that transmit several thousands of bits... **per day**.

- Is reliability the most significant factor for the Grid?
- How about bandwidth? RTT?

Deployment and testing of technologies is extremely expensive.

Answer: formal verification

Test, evaluate and tweak technologies - then deploy.

Model

What are the properties of the Smart Grid?

- It's a *cyber-physical* system
- It's a *distributed* system
- It is a *stochastic* system

Model

What are the properties of the Smart Grid?

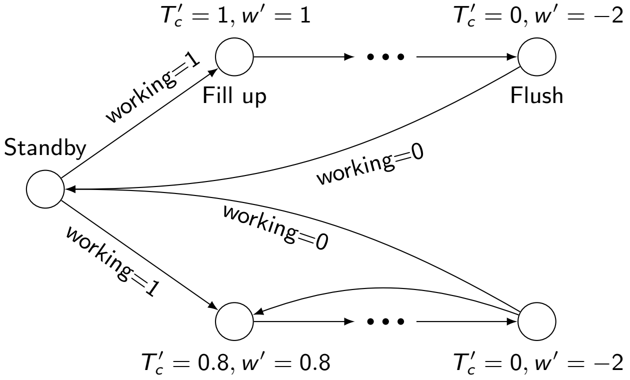
- It's a *cyber-physical* system
- It's a *distributed* system
- It is a *stochastic* system

Plan:

1. Develop hybrid, distributed and probabilistic model
2. Develop logic for stating properties
3. Verify properties using existing statistical model-checking techniques

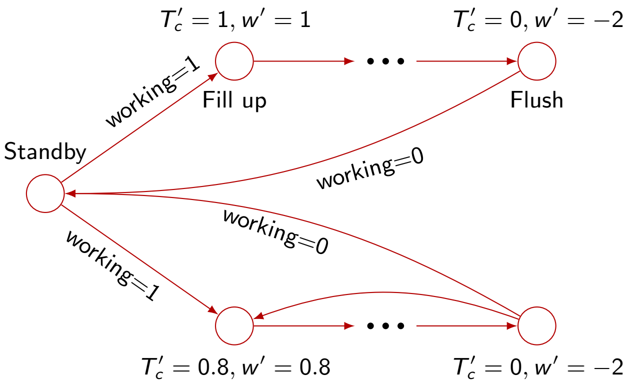
Summary

DTHA [12]: Washing machine



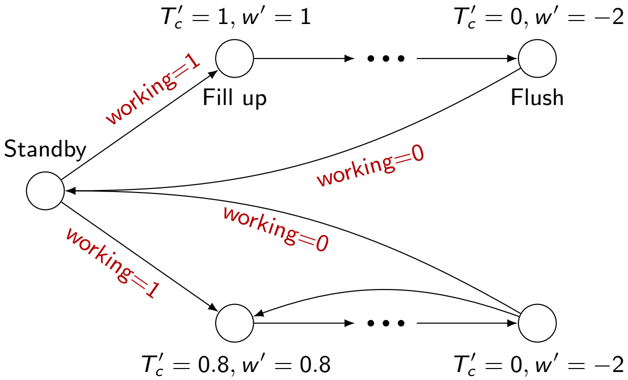
T_c is total water consumed, w is water currently in the machine

Washing machine



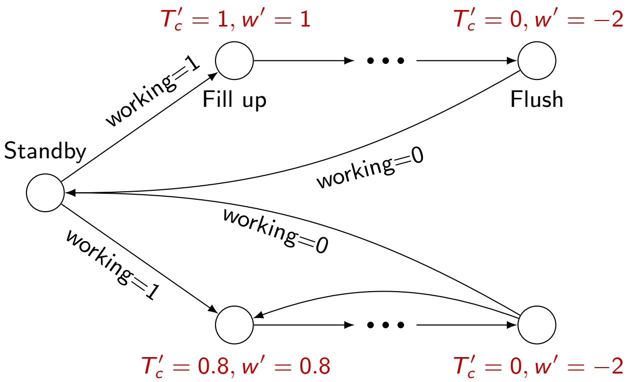
Control graph $\langle Q, E \rangle$

Washing machine



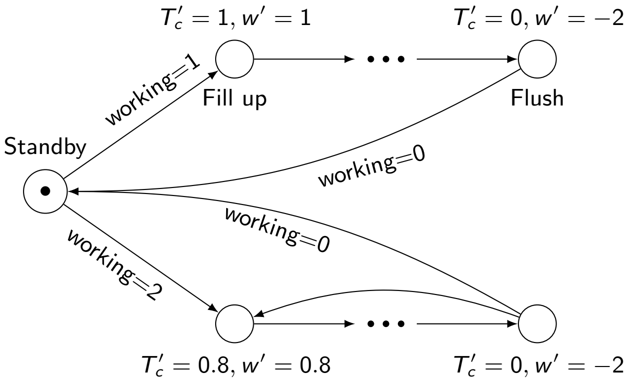
Jump relation $\mathtt{jump}_e : \mathbb{R}^n \times \mathbb{R}^n$

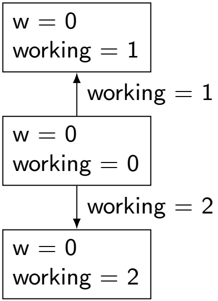
Washing machine



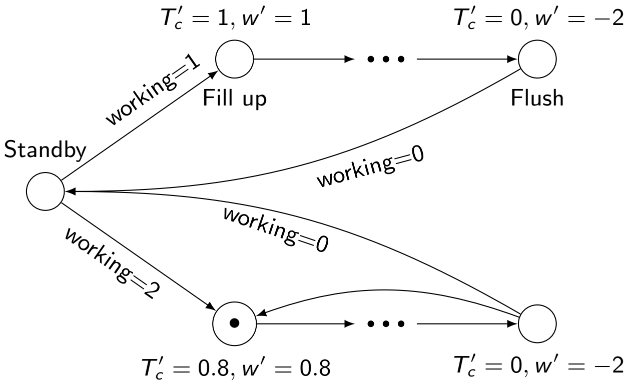
Flows $\varphi_q : \mathbb{R}_{\geq 0} \times \mathbb{R}^d \to \mathbb{R}^d$

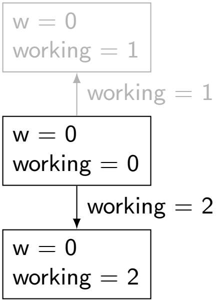
Washing machine: scheduler



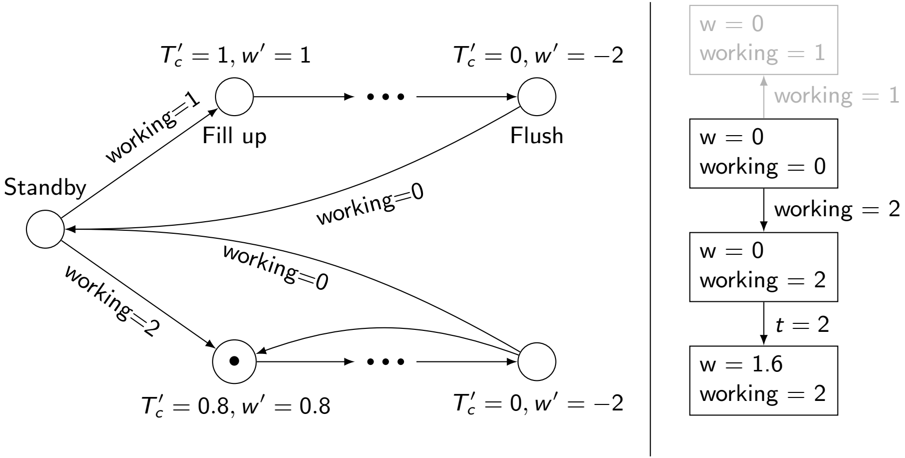


Washing machine: scheduler

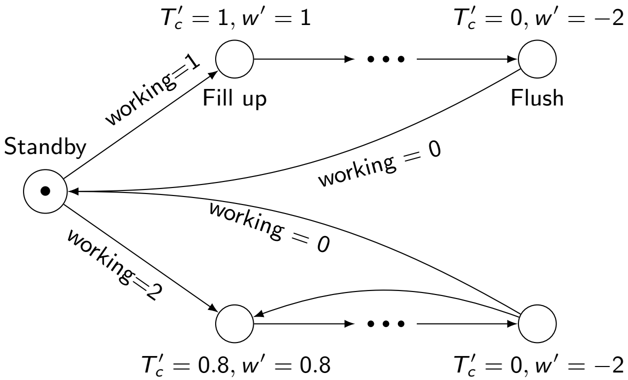


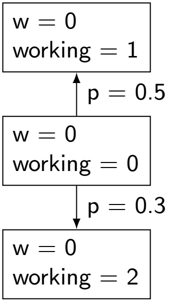


Washing machine: scheduler



Washing machine: probabilistic scheduler





Multiple washing machines?

Multiple washing machines?

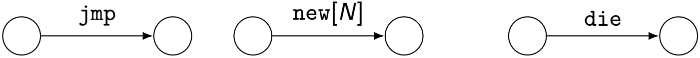
What if they leave?

What if they leave?




Actions

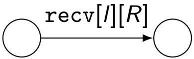
Washing machines behave like Petri Net markings.



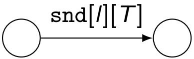
jump$_e$	create new entity given by N	makes entity disappear

Actions

They can also communicate *asynchronously*.



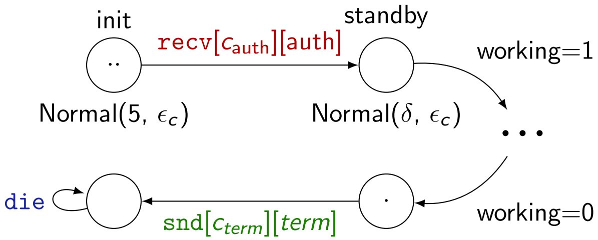
Channel l, reacts with R



Channel l, message content T

Example: apartment laundry

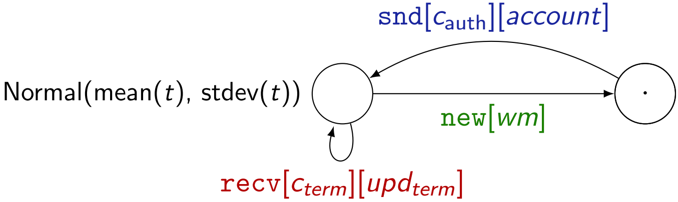
Washing machines first initialise, then wait for authorisation to start working



They announce when they finish, and exit the system.

Example: apartment laundry

The central unit keeps track of working machines, and starts and enables them



Summary

Bounded LTL cannot deal with changing number of variables.

Definition (Syntax of QBLTL)

Formulae of QBLTL are given by the following grammar, with $* \in \{+, -, \div, \times, \hat{}\}$ and $\sim \in \{\leq, \geq, =\}$:

$\theta ::= c \mid \theta_1 * \theta_2 \mid \pi_i(e) \mid \mathbb{E}(e) \mid \mathbf{ag}[e](\theta)$, with $i \in \mathbb{N}, c \in \mathbb{Q}$

$\phi ::= \mathbb{E}(e) \mid \theta_1 \sim \theta_2 \mid \phi_1 \vee \phi_2 \mid \neg\phi_1 \mid \phi_1 \, \mathbf{U}^t \phi_2 \mid \exists e.\phi_1$

Bounded LTL cannot deal with changing number of variables.

Definition (Syntax of QBLTL)

Formulae of QBLTL are given by the following grammar, with $* \in \{+, -, \div, \times, \hat{}\}$ and $\sim \in \{\leq, \geq, =\}$:

$\theta ::= c \mid \theta_1 * \theta_2 \mid \pi_i(e) \mid \mathsf{E}(e) \mid \mathbf{ag}[e](\theta)$, with $i \in \mathbb{N}, c \in \mathbb{Q}$

$\phi ::= \mathsf{E}(e) \mid \theta_1 \sim \theta_2 \mid \phi_1 \vee \phi_2 \mid \neg\phi_1 \mid \phi_1 \mathbf{U}^t \phi_2 \mid \exists e.\phi_1$

Bounded LTL cannot deal with changing number of variables.

Definition (Syntax of QBLTL)

Formulae of QBLTL are given by the following grammar, with
$* \in \{+, -, \div, \times, \hat{}\}$ and $\sim \in \{\leq, \geq, =\}$:
$\theta ::= c \mid \theta_1 * \theta_2 \mid \pi_i(e) \mid \mathsf{E}(e) \mid \mathbf{ag}[e](\theta)$, with $i \in \mathbb{N}, c \in \mathbb{Q}$
$\phi ::= \mathsf{E}(e) \mid \theta_1 \sim \theta_2 \mid \phi_1 \vee \phi_2 \mid \neg\phi_1 \mid \phi_1 \, \mathbf{U}^t \phi_2 \mid \exists e.\phi_1$

Bounded LTL cannot deal with changing number of variables.

Definition (Syntax of QBLTL)

Formulae of QBLTL are given by the following grammar, with $* \in \{+, -, \div, \times, \hat{}\}$ and $\sim \in \{\leq, \geq, =\}$:

$\theta ::= c \mid \theta_1 * \theta_2 \mid \pi_i(e) \mid \mathbb{E}(e) \mid \mathbf{ag}[e](\theta)$, with $i \in \mathbb{N}, c \in \mathbb{Q}$

$\phi ::= \mathbb{E}(e) \mid \theta_1 \sim \theta_2 \mid \phi_1 \vee \phi_2 \mid \neg\phi_1 \mid \phi_1 \mathbf{U}^t \phi_2 \mid \exists e.\phi_1$

Bounded LTL cannot deal with changing number of variables.

Definition (Syntax of QBLTL)

Formulae of QBLTL are given by the following grammar, with $* \in \{+, -, \div, \times, \hat{}\}$ and $\sim \in \{\leq, \geq, =\}$:

$\theta ::= c \mid \theta_1 * \theta_2 \mid \pi_i(e) \mid \mathrm{E}(e) \mid \mathbf{ag}[e](\theta)$, with $i \in \mathbb{N}, c \in \mathbb{Q}$

$\phi ::= \mathrm{E}(e) \mid \theta_1 \sim \theta_2 \mid \phi_1 \vee \phi_2 \mid \neg\phi_1 \mid \phi_1 \, \mathbf{U}^t \phi_2 \mid \exists e.\phi_1$

Bounded LTL cannot deal with changing number of variables.

Definition (Syntax of QBLTL)

Formulae of QBLTL are given by the following grammar, with $* \in \{+, -, \div, \times, \hat{}\}$ and $\sim \in \{\leq, \geq, =\}$:

$\theta ::= c \mid \theta_1 * \theta_2 \mid \pi_i(e) \mid \mathsf{E}(e) \mid \mathbf{ag}[e](\theta)$, with $i \in \mathbb{N}, c \in \mathbb{Q}$

$\phi ::= \mathsf{E}(e) \mid \theta_1 \sim \theta_2 \mid \phi_1 \vee \phi_2 \mid \neg\phi_1 \mid \phi_1 \, \mathbf{U}^t \phi_2 \mid \exists e.\phi_1$

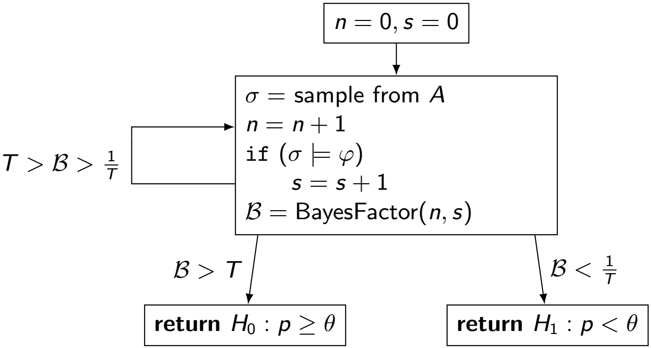
Lemma

QBLTL (like BLTL) has bounded simulation traces.

Why Statistical Model Checking?

- Estimates probabilities of properties holding (they will never hold *always*)
- Very efficient
- Model is very hard to analyse (dynamic, unbounded number of entities, asynchronous messages, etc)
- Plug'n'play, e.g. black-box model
- Successfully used in many applications (cf. Edmund Clarke's work)

Statistical Model Checking: Hypothesis Testing [12]



Statistical Model Checking

$$\mathcal{B} = \frac{P(d|H_0)}{P(d|H_1)}$$

A *large* Bayes Factor $\mathcal{B}$ is *evidence* for $H_0 : p > \theta$.
A *small* Bayes Factor $\mathcal{B}$ is *evidence* for $H_1 : p \leq \theta$.

Theorem (Error bounds for Hypothesis Testing [12])

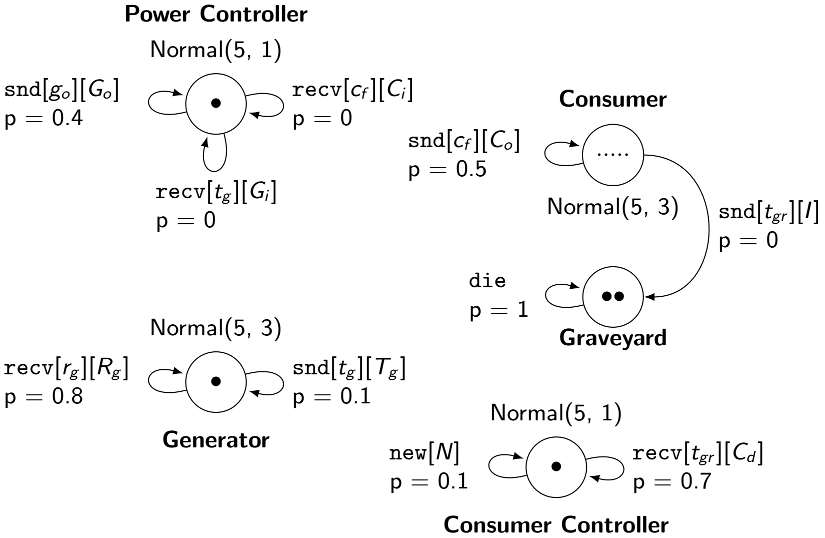
For any discrete random variable and prior, the probability of a Type I-II error for the Bayesian hypothesis testing algorithm 2 is bounded above by $\frac{1}{T}$*, where* T *is the Bayes Factor threshold given as input.*

A more sophisticated Interval Estimation Algorithm estimates p.

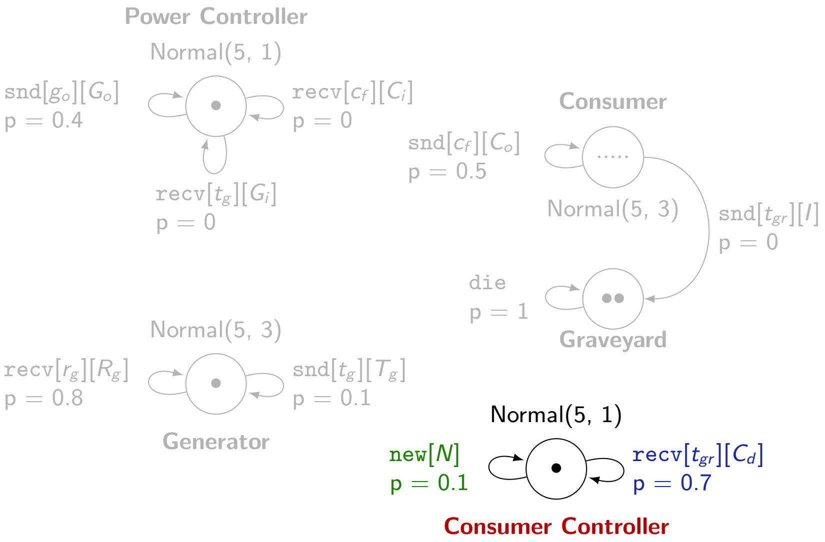
Summary

Even today, utilities deploy networks that transmit several thousands of bits per day (low bandwidth).

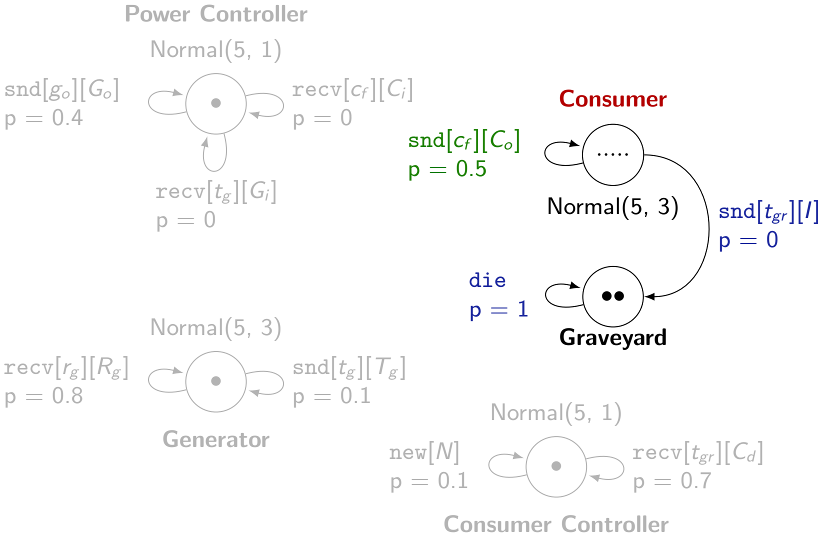
Can we evaluate the impact of network reliability?



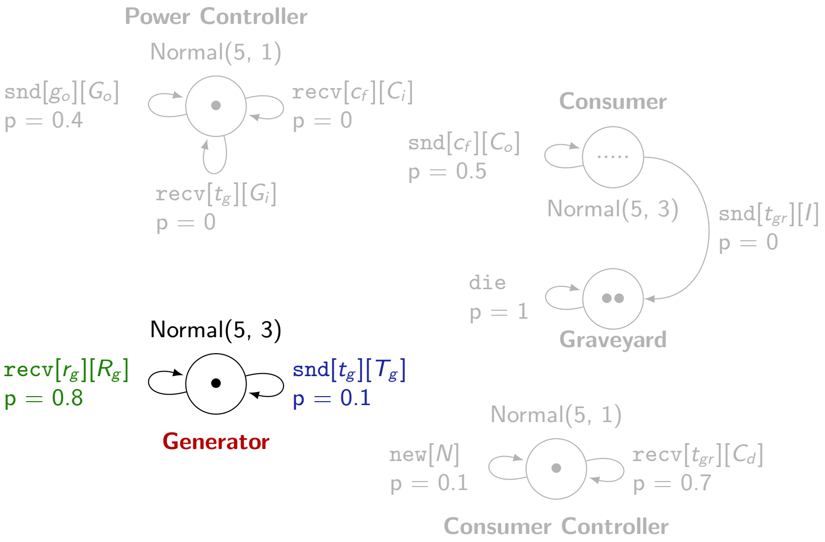
Power Controller
Normal(5, 1)
snd[g_o][G_o]
p = 0.4
recv[c_f][C_i]
p = 0
recv[t_g][G_i]
p = 0
Consumer
snd[c_f][C_o]
p = 0.5
.....
Normal(5, 3)
snd[t_{gr}][I]
p = 0
die
p = 1
Graveyard
Normal(5, 3)
recv[r_g][R_g]
p = 0.8
snd[t_g][T_g]
p = 0.1
Generator
Normal(5, 1)
new[N]
p = 0.1
recv[t_{gr}][C_d]
p = 0.7
Consumer Controller



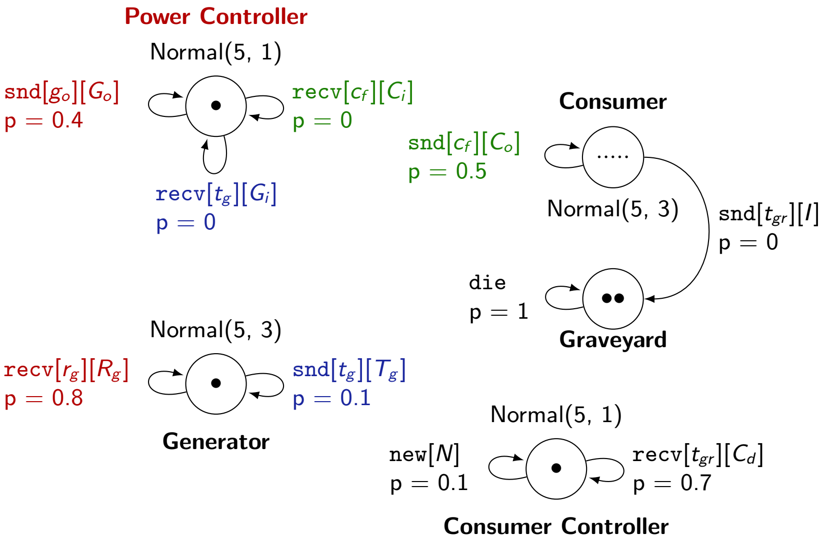
Creates classes of consumers. Keeps track of them.



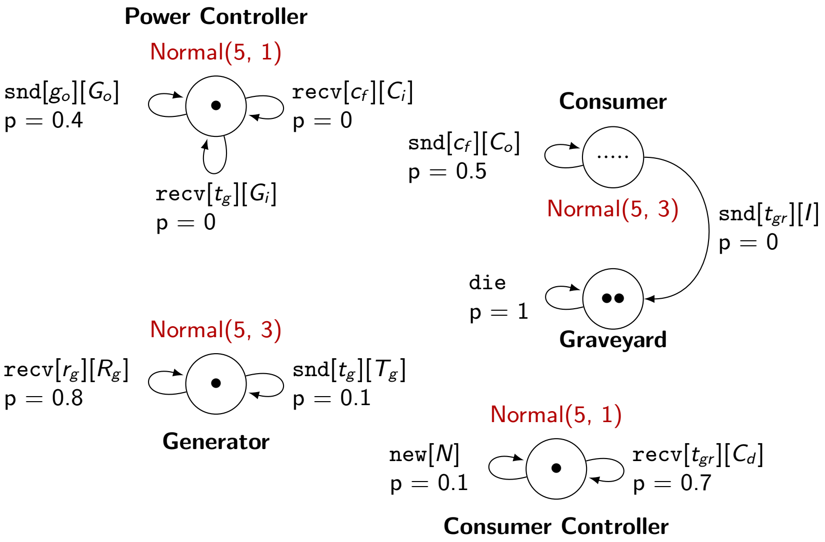
Consumers feedback consumption. They announce death and exit.



The generator receives control messages and sends output info.



The power controller ties it all together.



Messages get sent periodically.

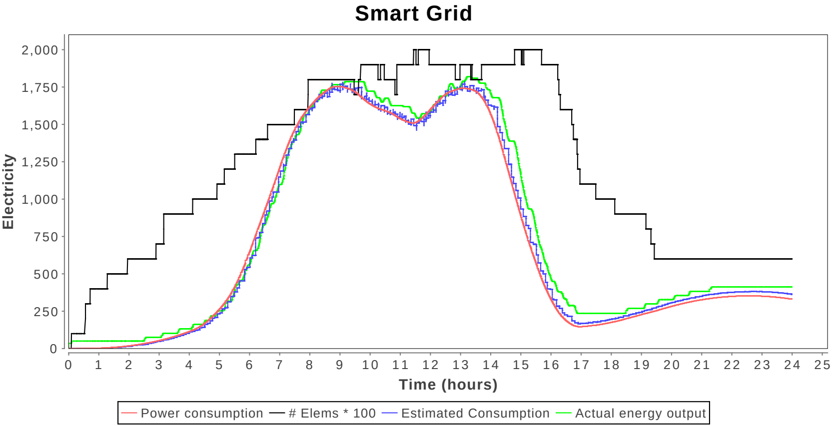


Figure: Sample run from the modelled system.

Properties

Property (1): the output of the generator is always within 400 units of energy of actual demand

$$\mathbf{G}^{1440}|\sum[e](Gen(e)\cdot\pi_{\text{output}}(e))-\sum[e](Cons(e)\cdot\pi_{\text{consumption}}(e))|<400$$

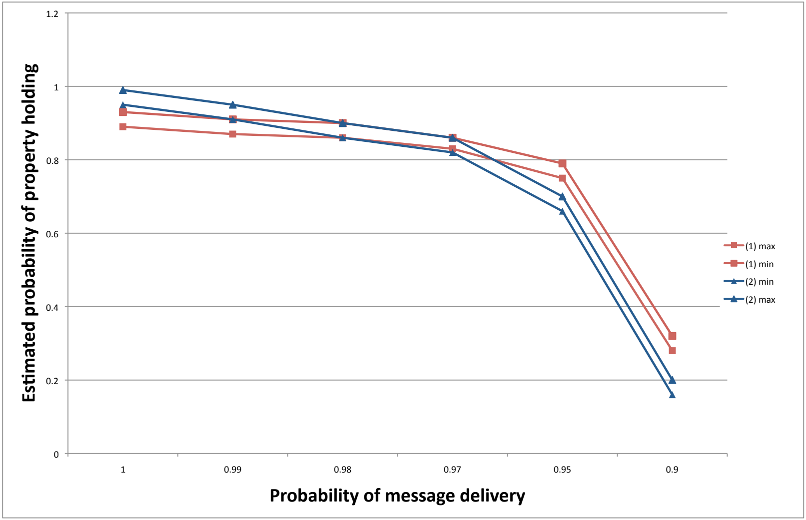
Property (2): the PC's estimate of power consumption is not too far from the truth.

$$\mathbf{G}^{1440}|\sum[e](Gen(e)\cdot\pi_{\text{output}}(e))-$$

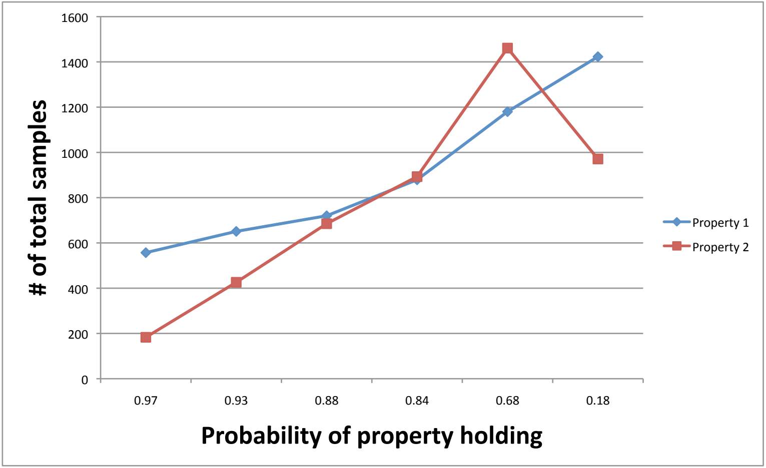
$$-\sum[e](PC(e)\cdot(\pi_0(e)+...+\pi_{19}(e)))|<250$$

We estimate that Property (1) is harder than (2).

Results: probability of satisfying properties



Results: number of traces required



Case Study Conclusion

- The algorithm is efficient (# of traces required)
- Reliability becomes a major concern *only below a certain threshold*
- Utilities can easily visualise the cost/benefit relation
- Once the model has been implemented, it can be tweaked and retested quickly

Summary

Contributions

- Developed a formal model for distributed, probabilistic cyber-physical systems
- Extended BLTL to QBLTL
- Ensured SMC could be applied
- Implemented the above in a Java library
- Studied network reliability in a simplified Smart Grid model
- Quickly revealed important relations in a non-trivial system

Edmund M. Clarke, Alexandre Donzé, and Axel Legay.
Statistical model checking of mixed-analog circuits with an application to a third order delta-sigma modulator.
In *Haifa Verification Conference*, pages 149–163, 2008.

Edmund M. Clarke, Alexandre Donzé, and Axel Legay.
On simulation-based probabilistic model checking of mixed-analog circuits.
Formal Methods in System Design, 36(2):97–113, 2010.

Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla.
Automatic verification of finite-state concurrent systems using temporal logic specifications.
ACM Trans. Program. Lang. Syst., 8(2), 1986.

Edmund M. Clarke, James R. Faeder, Christopher James Langmead, Leonard A. Harris, Sumit Kumar Jha, and Axel Legay.
Statistical model checking in biolab: Applications to the automated analysis of t-cell receptor signaling pathway.
In *CMSB*, pages 231–250, 2008.

Sumit Kumar Jha, Edmund M. Clarke, Christopher James Langmead, Axel Legay, André Platzer, and Paolo Zuliani.
A bayesian approach to model checking biological systems.
In *CMSB*, 2009.

Nancy A. Lynch.
Input/Output automata: Basic, timed, hybrid, probabilistic, dynamic, ...
In *CONCUR*, pages 187–188, 2003.

José Meseguer and Raman Sharykin.
Specification and analysis of distributed object-based stochastic hybrid systems.
In *HSCC*, 2006.

Ying-Chih Wang, Anvesh Komuravelli, Paolo Zuliani, and Edmund M. Clarke.
Analog circuit verification by statistical model checking.
In *ASP-DAC*, pages 1–6, 2011.

E. Yahav, T. Reps, and M. Sagiv.
LTL model checking for systems with unbounded number of dynamically created threads and objects.
Technical Report TR-1424, Computer Sciences Department, University of Wisconsin, 2001.

Håkan L. S. Younes, Edmund M. Clarke, and Paolo Zuliani.
Statistical verification of probabilistic properties with unbounded until.
In *SBMF*, 2010.

Håkan L. S. Younes and Reid G. Simmons.
Statistical probabilistic model checking with a focus on time-bounded properties.
Inf. Comput., 204(9):1368–1409, 2006.

Paolo Zuliani, André Platzer, and Edmund M. Clarke.
Bayesian statistical model checking with application to Simulink/Stateflow verification.
In *HSCC*, pages 243–252, 2010.

Thank you, questions?