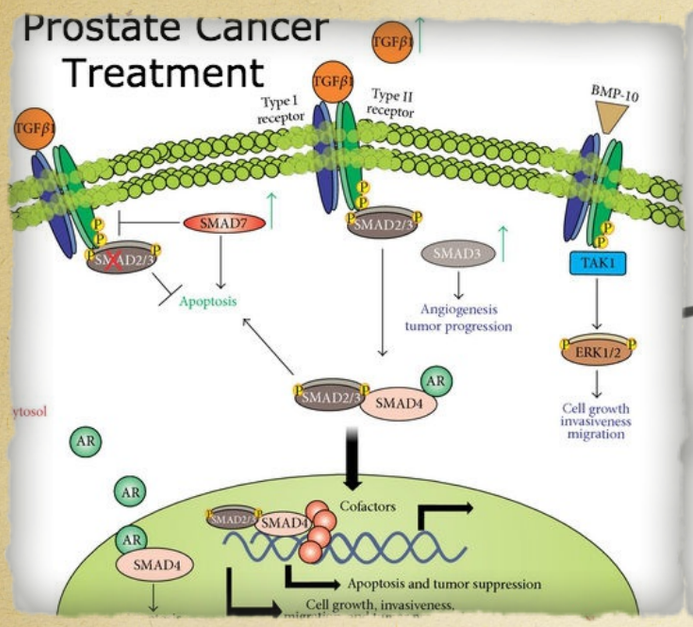
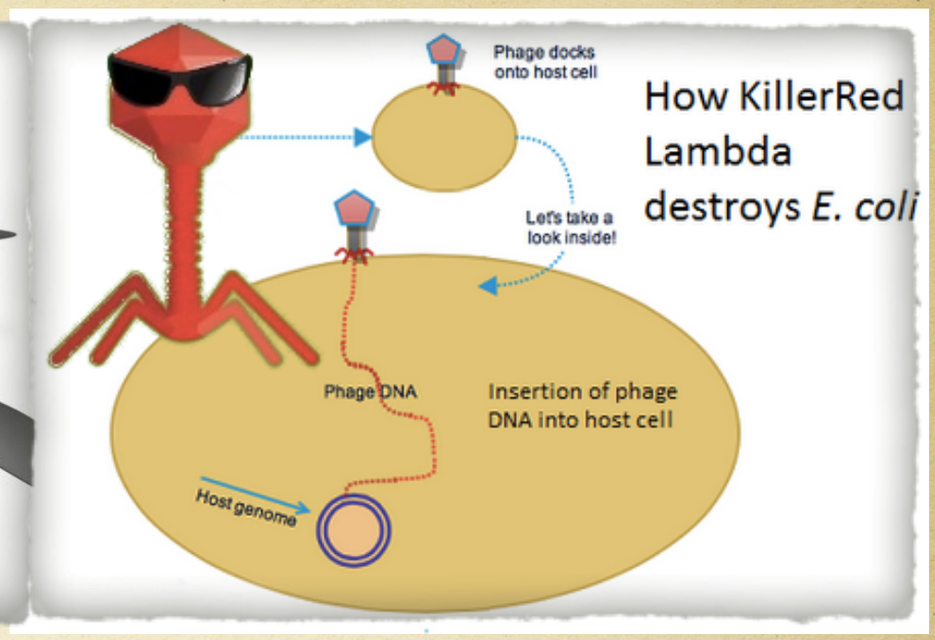




SReach:

Combing SMT-based Model Checking and Statistical Tests*

Presented by Qinsi Wang

EMC 2014 Symposium

**Qinsi Wang, Paolo Zuliani, Soonho Kong, Sicun Gao, and Edmund Clarke. SReach: Combining Statistical Tests and Bounded Model Checking for Nonlinear Hybrid Systems with Parametric Uncertainty*

Probabilistic Bounded Reachability Analysis of Stochastic Hybrid Systems

Stochastic Hybrid Systems

- Application:
  - Biology (e.g. the killerred biological model),
  - Cyber-physical systems (e.g. the quadcopter stabilization control system),
  - Financial models (e.g. insurance pricing systems), and so on.
- Formalism: Probabilistic hybrid automata (PHA), Stochastic hybrid automata (SHA), General Stochastic hybrid systems (GSHS), ...

SReach considers …

- Hybrid automata with parametric uncertainty
  - unknown parameters, individual differences, noisy data, learning errors, …
- Probabilistic hybrid automata
  - transitions with discrete probability distributions
  - transitions with continuous distributions, but discrete choices

SReach can handle…

- Probabilistic bounded reachability problems
- One of the elementary questions for the quantitative analysis of stochastic hybrid systems is to compute the probability of reaching a certain set of states.
- More specifically, model validation, parameter estimation, and sensitivity analysis

SReach answers ...

- SReach can answer two types of questions:
  - (1) Does the model satisfy a given reachability property with probability greater than a certain threshold? hypothesis testing
  - (2) What is the probability that the model satisfies a given reachability property? statistical estimation

SReach's algorithm

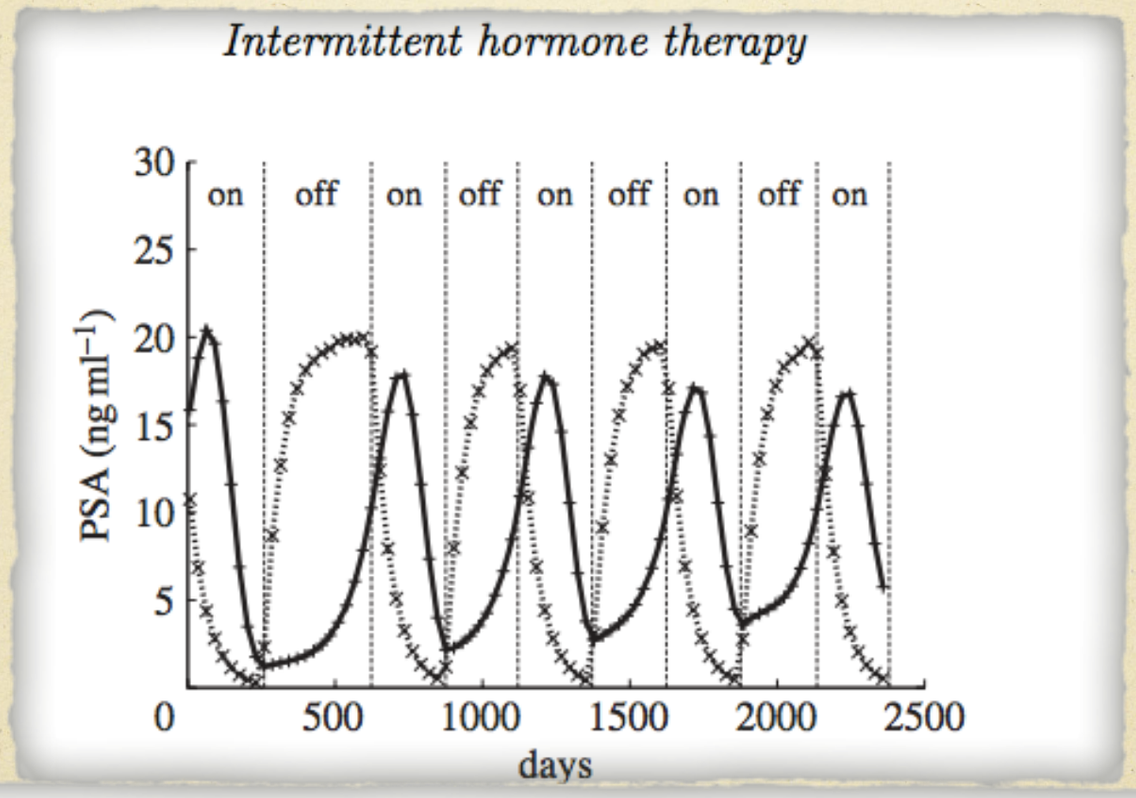
Algorithm 1 SReach for Hybrid Automata with Parametric Uncertainty

function SReach(MP, ST, δ, k)
    $Succ \leftarrow 0$ ▷ number of δ-sat samples
    $N \leftarrow 0$ ▷ total number of samples
    $RV \leftarrow$ ExtractRV(MP) ▷ get the RVs from the probabilistic model
    repeat
        $S_i \leftarrow \text{Sim}(RV)$ ▷ sample the parameters
        $M_i \leftarrow \text{Gen}(MP, S_i)$ ▷ generate a dReach model
        $Res \leftarrow \text{dReach}(M_i, \delta, k)$ ▷ call dReach to solve k-step δ-reachability
        if $Res = \delta$-sat **then**
            $Succ \leftarrow Succ + 1$
        end if
        $N \leftarrow N + 1$
    until $ST.done(Succ, N)$ ▷ perform statistical test
    return $ST.output$
end function

SReach uses ...

- The following statistical tests:
- Hypothesis testing methods: Lai's test, Bayes factor test, Bayes factor test with indifference region, and Sequential probability ratio test.
- Statistical estimation methods: Chernoff-Hoeffding bound, Bayesian interval estimation with beta prior, and Direct sampling.

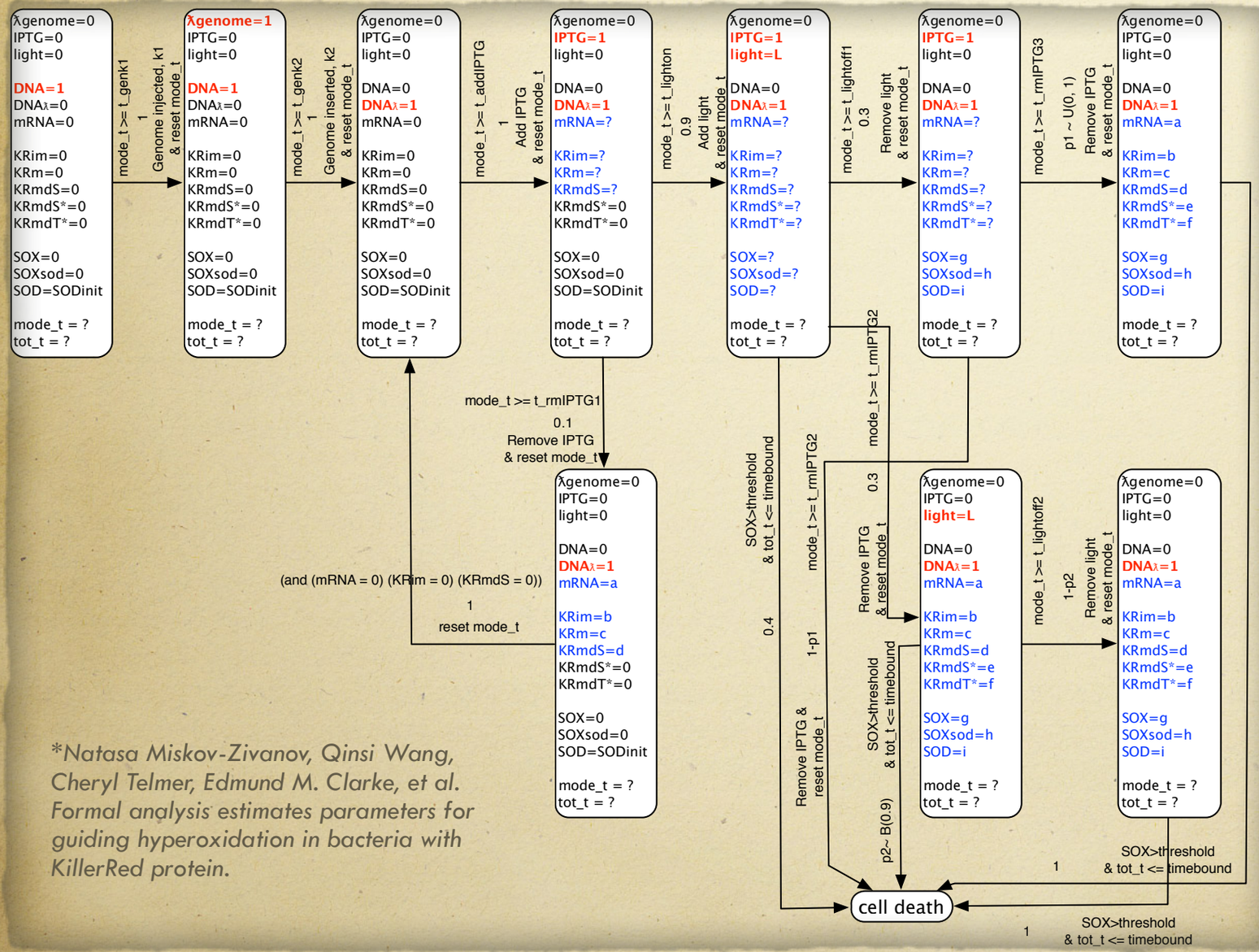
Hybrid automaton with parametric uncertainty



Model	#RVs	r_0	r_1	Est_P	#S_S	#T_S	T_pre_S(s)
PCT1	6	5.0	10.0	0.04	0	227	0.145
PCT2	6	7.0	11.0	0.591	2144	3628	432.491
PCT3	6	10.0	15.0	0.996	227	227	692.861

Table 1. #RVs = the number of random variables in the model, δ is the precision for the δ-decision procedure, #S_S = the number of samples with which dReal returns δ-Sat, #T_S = the number of total samples, r_0 is the lower threshold of the serum PSA level, r_1 is the upper threshold, Est_P indicates the estimated probability of the given model meeting the bounded reachability property, and T_per_S(s) gives the average CPU time costed by each sample in seconds.

Probabilistic Hybrid Automaton



**Natasa Miskov-Zivanov, Qinsi Wang, Cheryl Telmer, Edmund M. Clarke, et al. Formal analysis estimates parameters for guiding hyperoxidation in bacteria with KillerRed protein.*

- 1. Will the killerred kill bacteria cells within a certain time with probability no less then 0.95?
- 2. Will the time duration that keeps the light on impact the whole process significantly?
- What will be relation between the time to turn on the light and the time needed to kill bacteria cells?
- …

Experimental results

Model	#RVs	EPI_TO1	EPI_TO2	#S_S	#T_S	Est_P	A_T(s)	T_T(s)
Cd_to1_s	1	U(6.1e-3, 7e-3)	6	227	227	0.996	0.362	82.174
Cd_to1_uns	1	U(5.5e-3, 5.9e-3)	6	0	227	0.004	0.124	28.148
Cd_to2_s	1	400	U(0.131, 6)	227	227	0.996	0.361	81.947
Cd_to2_uns	1	400	U(0.1, 0.129)	0	227	0.004	0.139	31.552
Cd_to12_s	2	N(400, 1e-4)	N(6, 1e-4)	227	227	0.996	0.373	84.671
Cd_to12_uns	2	N(5.5e-3, 10e-6)	N(0.11, 10e-5)	0	227	0.004	0.131	29.737

Table 2: #RVs = number of random variables in the model, #S_S = number of δ-sat samples, #T_S = total number of samples, Est_P = estimated probability of property, A_T(s) = average CPU time of each sample in seconds, and T_T(s) = total CPU time for all samples in seconds.

Benchmark	#Ms	K	#ODEs	#Vs	#RVs	δ	Est_P	#S_S	#T_S	A_T(s)	T_T(s)
BBK1	1	1	2	14	3	0.001	0.754	5372	7126	0.086	612.836
BBK5	1	5	2	38	3	0.001	0.059	209	3628	0.253	917.884
BBwDv1	2	2	4	20	4	0.001	0.208	2206	10919	0.080	873.522
BBwDv2K2	2	2	4	20	3	0.001	0.845	7330	8669	0.209	1811.821
BBwDv2K8	2	8	4	56	3	0.001	0.207	2259	10901	0.858	9353.058
Tld	2	7	2	33	4	0.001	0.996	227	227	0.213	48.351
Ted	2	7	4	50	4	0.001	0.996	227	227	12.839	2914.448
DTldK3	2	3	4	26	2	0.001	0.996	227	227	0.382	86.714
DTldK5	2	5	4	38	2	0.001	0.161	1442	8961	0.280	2509.078
W4mv1	4	3	8	26	6	0.001	0.381	5953	15639	0.238	3722.082
W4mv2K3	4	3	8	26	6	0.001	0.996	227	227	0.673	152.771
W4mv2K7	4	7	8	50	6	0.001	0.004	0	227	0.120	27.240
DWK1	2	1	4	14	5	0.001	0.996	227	227	0.171	38.817
DWK3	2	3	4	26	5	0.001	0.996	227	227	0.215	48.806
DWK9	2	9	4	62	5	0.001	0.996	227	227	5.144	1167.688
Que	3	2	3	13	4	0.001	0.228	2662	11677	0.095	1109.315
3dOsc	3	2	18	48	2	0.001	0.996	227	227	8.273	1877.969
QuadC	1	0	14	44	6	0.001	0.996	227	227	825.641	187420.507

Table 3: #Ms = number of modes, K indicates the unfolding steps, #ODEs = number of ODEs in the model, #Vs = number of total variables in the unfolded formulae, #RVs = number of random variables in the model, δ = precision used in **dReach**, #S_S = number of δ-sat samples , #T_S = total number of samples, Est_P = estimated probability of the property, A_T(s) = average CPU time of each sample in seconds, and T_T(s) = total CPU time for all samples in seconds.

Future work

- General Stochastic Hybrid Systems
  - probabilistic jumps with continuous distributions
  - stochastic flows: stochastic differential equations
- Propose and implement a new theory solver for a subset of probability theory

Thanks, Ed!

- https://github.com/dreal/SReach
- paralleled version will be released soon!