Multi-Objective Parameter Fitting in Parametric Probabilistic Hybrid Automata

— Learning to Mine and Exploit PAC Formal Models —

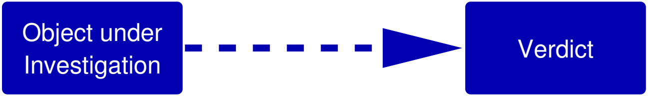
Martin Fränzle[1]

joint work with

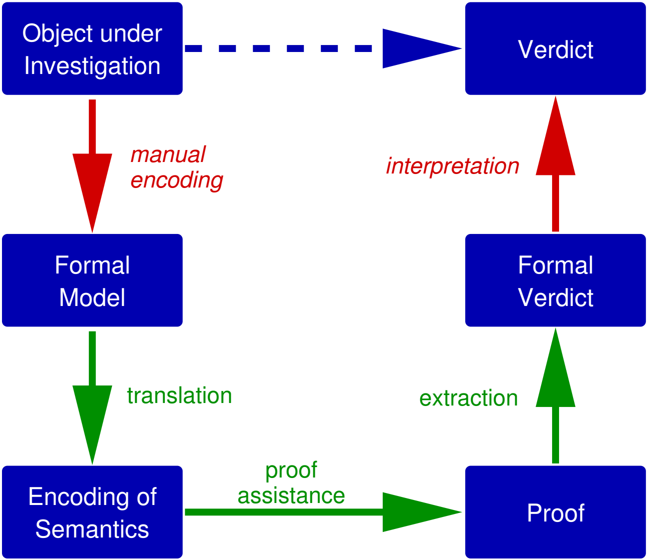
Alessandro Abate (Oxford University, UK),
Sebastian Gerwinn (OFFIS e.V., FRG),
Joost-Pieter Katoen (RWTH Aachen, FRG),
Paul Kröger (CvOU Oldenburg, FRG)

[1] Dpt. of Computing Science · Carl von Ossietzky Universität · Oldenburg, Germany

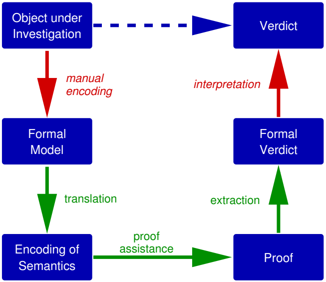
The traditional formal verification cycle



The traditional formal verification cycle



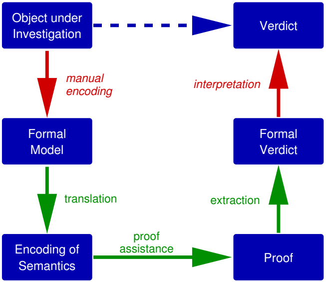
The traditional formal verification cycle



But what if

- faithful formal modeling is too complex to be feasible?
- object under investigation is an embedded system that learns part of its behavior only after deployment (and thus, after verification time)?
- object under investigation is an autonomous system which may eventually enter unknown (and thus, impossible to model a priori) environments & unpredictable system configurations?

The traditional formal verification cycle



But what if

- faithful formal modeling is too complex to be feasible?
- object under investigation is an embedded system that learns part of its behavior only after deployment (and thus, after verification time)?
- object under investigation is an autonomous system which may eventually enter unknown (and thus, impossible to model a priori) environments & unpredictable system configurations?

Such applications become increasingly relevant, *challenging our approaches to verification.*

Example: Safety-critical learning in situ

Predicting direction of driving requires

- detailed knowledge of factual tracks,
- which may not coincide with marked lanes,
- and which may change unexpectedly due to, e.g., construction works.

Example: Safety-critical learning in situ

Predicting direction of driving requires

- detailed knowledge of factual tracks,
- which may not coincide with marked lanes,
- and which may change unexpectedly due to, e.g., construction works.

Industry wants to counter these problems by

- use of *high-resolution digital maps*, plus
- *machine learning* for (temporarily) adapting the map in situ.

Example: Safety-critical learning in situ

Predicting direction of driving requires

- detailed knowledge of factual tracks,
- which may not coincide with marked lanes,
- and which may change unexpectedly due to, e.g., construction works.

Industry wants to counter these problems by

- use of *high-resolution digital maps*, plus
- *machine learning* for (temporarily) adapting the map in situ.

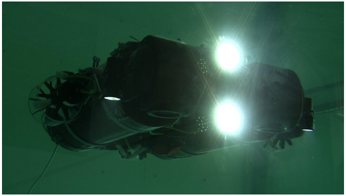
How to make sure that machine learning

- doesn't err in interpreting observations and in learning?
- actually learns relevant facts?
- invalidates them when no longer factual?

Example: Unpredictable system configurations

Future cyber-physical systems will be *long-term autonomous*:

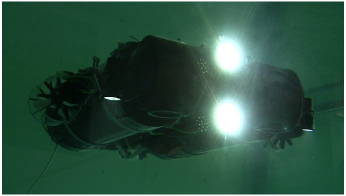
- sustain unattended operation for orders of magnitude longer duration than the typical inter-maintenance period of systems in the respective class,
- thereby have to be guaranteed to stay safe, reliable, operational, . . .



Example: Unpredictable system configurations

Future cyber-physical systems will be *long-term autonomous*:

- sustain unattended operation for orders of magnitude longer duration than the typical inter-maintenance period of systems in the respective class,
- thereby have to be guaranteed to stay safe, reliable, operational, . . .



which implies that they

- have to survive arbitrary combinations of multi-point failures, component degradations, component losses, . . . , as well as unpredicted environments
- employing behavioral adaptation (e.g., multi-objective parameter fitting), reconfiguration, function substitution, . . .

spanning a configuration space

- too large to be verified in advance,
- such that adaptation has to be safeguarded and guided by verification.

The mission:

Applications increasingly call for bridging the gap betw. AI techniques and FMs, e.g.:

Machine learning

Symbolic verification

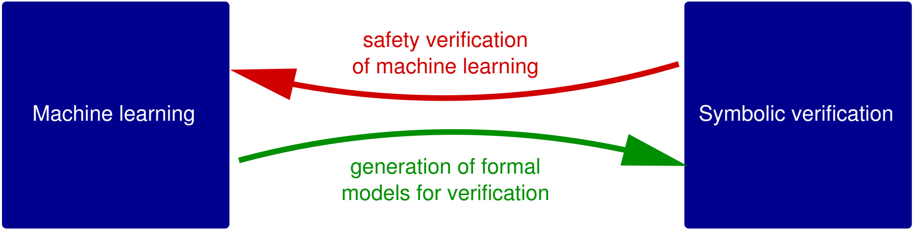
The mission:

Applications increasingly call for bridging the gap betw. AI techniques and FMs, e.g.:

- Need for mechanically supplying safety certificates for machine learning and similar AI techniques (statically and/or run-time verification)

The mission:

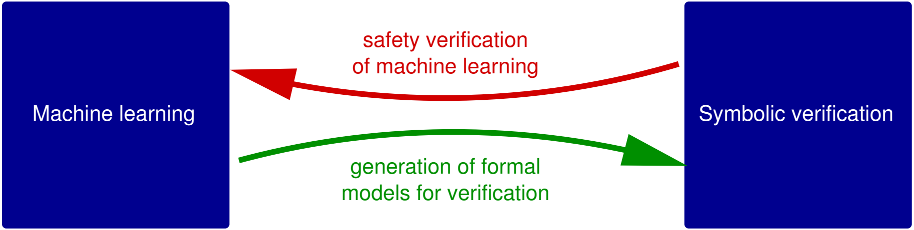
Applications increasingly call for bridging the gap betw. AI techniques and FMs, e.g.:



- Need for mechanically supplying safety certificates for machine learning and similar AI techniques (statically and/or run-time verification)
- May want to exploit AI techniques to bridge the modeling gap
  - when entering unknown / partially known environments, unpredicted system configuration, . . .
  - when faced with overly complex modeling task.

The mission: overall and today

Applications increasingly call for bridging the gap betw. AI techniques and FMs, e.g.:



- Need for mechanically supplying safety certificates for machine learning and similar AI techniques (statically and/or run-time verification)
- May want to exploit AI techniques to bridge the modeling gap
  - when entering unknown / partially known environments, unpredicted system configuration, ...
  - when faced with overly complex modeling task.

A bird's eye view of what we'll achieve today

Traditional symbolic analysis assumes a well-understood, closed-form symbolic representation facilitating constraint-based analysis:

Preoccupation to a fixed representation may prevent some fruitful applications:

- What happens, e.g., if the constraint representation is learnt from samples, thus blending machine learning with constraint solving?

A bird's eye view of what we'll achieve today

Traditional symbolic analysis assumes a well-understood, closed-form symbolic representation facilitating constraint-based analysis:

Preoccupation to a fixed representation may prevent some fruitful applications:

- What happens, e.g., if the constraint representation is learnt from samples, thus blending machine learning with constraint solving?
- Could we perhaps *automatically* generate/mine PAC formalizations?

Example: Demand-Response Schemes in Smart Grids

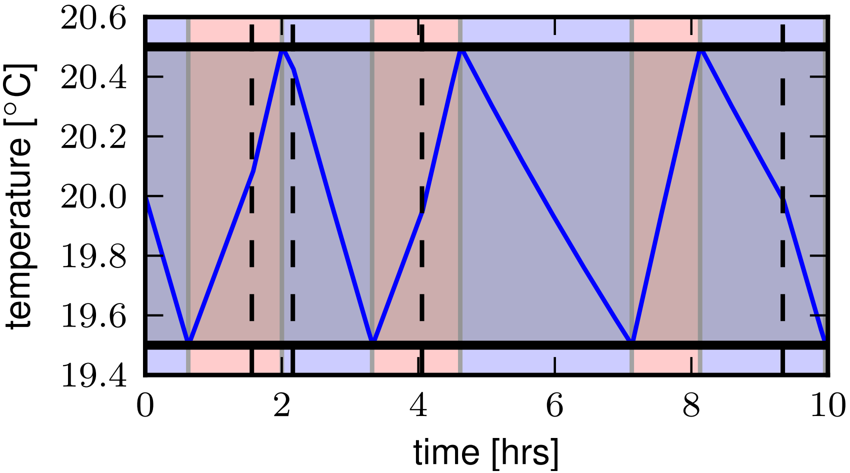
A Practical Problem Featuring Hybrid Dynamics

Demand Response: Supplying Reserve Power by Thermostatically Ctrl.ed Loads (TCLs) [Callaway 2009]

Idea: Control power demand by (marginally) modifying switching thresholds of AC systems.

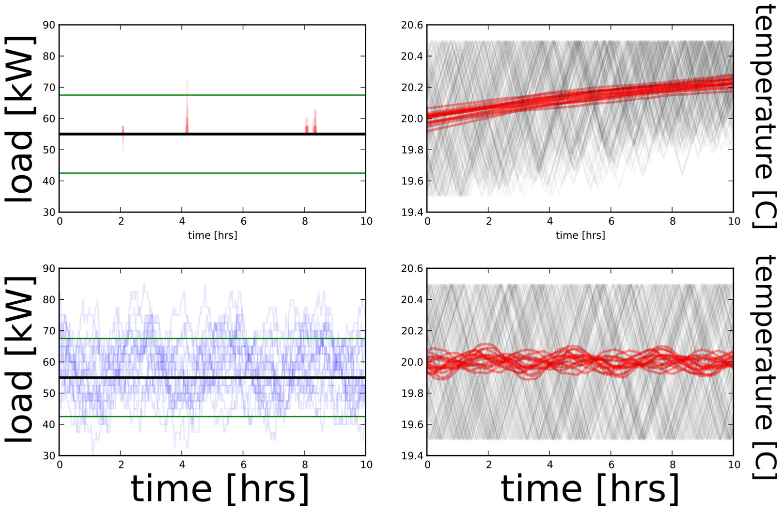
- On power shortage, provide reserve power by switching off early / switching on late.
- On excess power, consume reserve power by switching off late / switching on early.
- Unnoticeable to residents due to marginal adjustments to switching thresholds.

Dynamics of a Single Household — Simulation



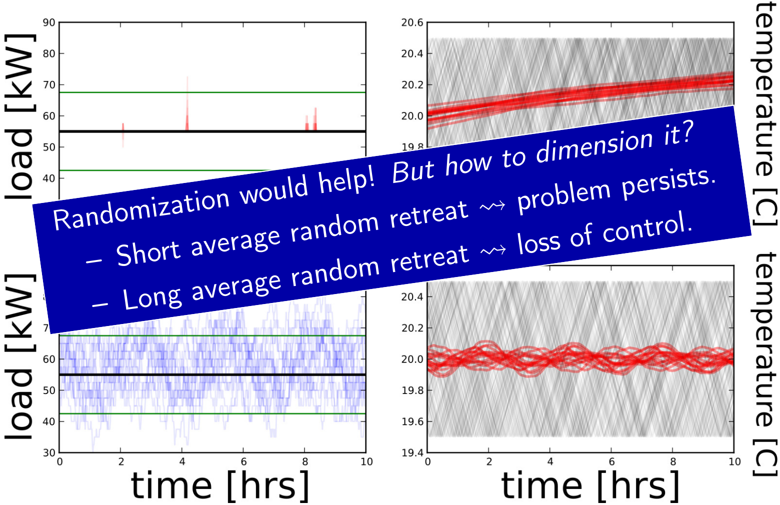
Dashed lines indicate window opening / closing events.

Multiple Similar TCLs ($N = 50$) — Simulation



Externally controlled (power target 55 kW) vs. uncontrolled ensemble.
Control strategy: switch off coldest households if power target exceeded.

Multiple Similar TCLs ($N = 50$) — Simulation



Randomization would help! *But how to dimension it?*

– Short average random retreat ⇝ problem persists.

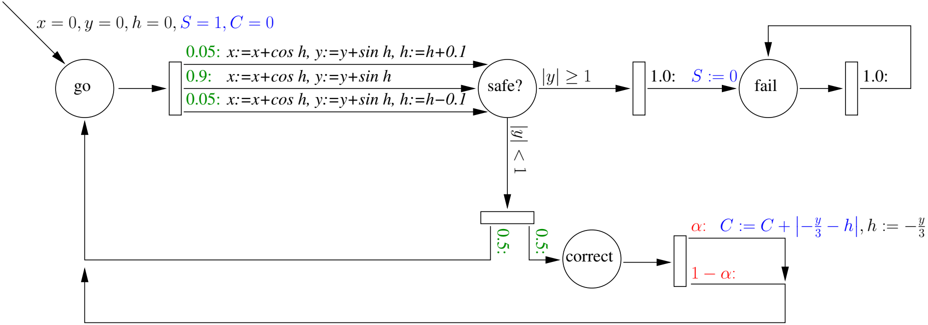
– Long average random retreat ⇝ loss of control.

Externally controlled (power target 55 kW) vs. uncontrolled ensemble.
Control strategy: switch off coldest households if power target exceeded.

The Formal Model

Parametric Probabilistic HA

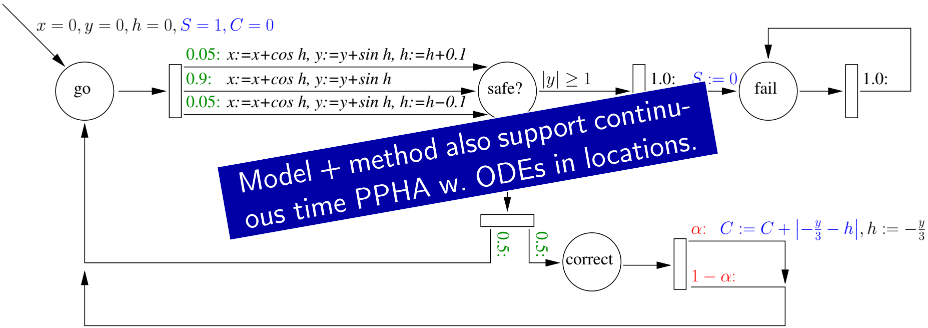
A (discrete time) Parametric Probabilistic HA



Car maneuvre: Keep lane while driving along a road.

- Measurement of position in lane fails with probability 0.5.
- Upon success, do occasional (due to cost associated) corrections of heading angle h by proportional control.
  - Parameter α controls frequency of these corrective actions.
- Two reward / cost variables:
  - C records accumulated cost of corrective steering actions,
  - S records successful stay in lane.

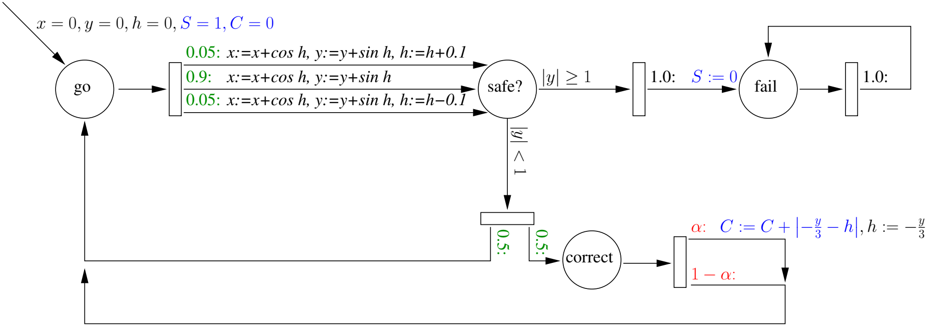
A (discrete time) Parametric Probabilistic HA



Car maneuvre: Keep lane while driving along a road.

- Measurement of position in lane fails with probability 0.5.
- Upon success, do occasional (due to cost associated) corrections of heading angle h by proportional control.
  - Parameter α controls frequency of these corrective actions.
- Two reward / cost variables:
  - C records accumulated cost of corrective steering actions,
  - S records successful stay in lane.

A multi-objective design problem



Find parameterization α^* such that

- the system is sufficiently safe: $P(\text{safe}) = \mathcal{E}(\text{S}, \alpha^*) \geq \theta_1$, where θ_1 is the safety target;
- at acceptable cost: $\mathcal{E}(C, \alpha^*) \leq \theta_2$, where θ_2 is a cost bound.

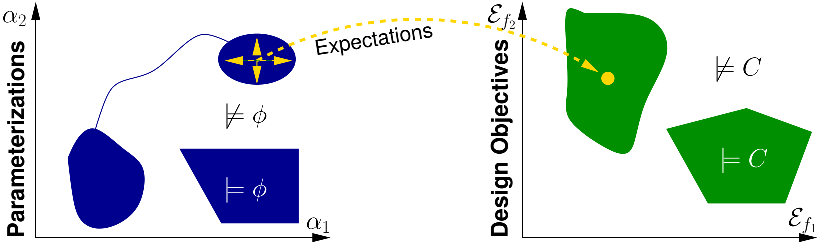
The design problem, abstractly

Given

1. a PPHA A, featuring
   - a vector $\vec{\alpha} = (\alpha_1, \ldots, \alpha_k)$ of parameters,
   - a vector $\vec{f} = (f_1, \ldots, f_n)$ of reward (or cost) functions,
2. a constraint ϕ over $\vec{\alpha}$ specifying the possible parameter instances, and
3. a constraint C over $\mathcal{E}_{\vec{f}}$ specifying the (multi-objective) design goal,

find (or prove non-existence of) a parameter instance $\vec{\alpha}^* \in \mathbb{R}^k$ that

1. satisfies ϕ and
2. yields expected *time-bounded rewards* $\mathcal{E}[\vec{f}, \vec{\alpha}^*]$ satisfying C.



Approach [F., Gerwinn, Kröger, Abate, Katoen, FORMATS '15]

1. Substitution of parametric probabilities in the system model by fixed substitute probabilities;
2. Introduction of counters into the model counting how frequently such substitutes have been chosen along a simulation run;
3. Statistical model checking of the modified model, yielding estimates of the expected costs/rewards in the non-parametric substitute model;
4. Exploitation of the re-normalization equations of importance sampling for obtaining a symbolic expression of the (estimated) parameter dependency of the costs/rewards;
5. Simplification of that expression by means of merging terms;
6. Use of SMT solving over, a.o., higher-order polynomials for determining suitable parameters.

Estimating (Parametric) Expectations by Random Sampling

Sampling as in traditional SMC [Younes, Simmons 2002–]

$p(\cdot;\alpha)$ be the parameter-dependent distribution of random variable $x \in X$;
let $\alpha^* \models \phi$ be a fixed parameter instance;
let $f : X \to [a, b]$ be a bounded reward function.

Expectation of f depending on α:

$$\mathcal{E}[f;\alpha] = \sum_{x \in X} f(x)p(x;\alpha) \tag{1}$$

Sampling as in traditional SMC [Younes, Simmons 2002–]

$p(\cdot;\alpha)$ be the parameter-dependent distribution of random variable $x \in X$;
let $\alpha^* \models \phi$ be a fixed parameter instance;
let $f : X \to [a, b]$ be a bounded reward function.

Expectation of f depending on α:

$$\mathcal{E}[f;\alpha] = \sum_{x \in X} f(x)p(x;\alpha) \tag{1}$$

Estimated expectation of f in α^*:

1. Use randomized simulation faithfully representing $p(\cdot,\alpha^*)$ to generate n samples $x_1, \ldots, x_m \in X$.
2. Compute the empirical mean

$$\tilde{\mathcal{E}}[f;\alpha^*] = \frac{1}{N}\sum_{i=1}^{N} f(x_i) \tag{2}$$

of the sampled f values.

Quality of the estimate

For large numbers of samples N, grossly outlying estimates are unlikely.

Quality of the estimate

For large numbers of samples N, grossly outlying estimates are unlikely.

Hoeffding's inequality [Hoeffding, 1963] yields

$$P\left(\tilde{\mathcal{E}}[f;\alpha^*] - \mathcal{E}[f;\alpha^*] \geq +\varepsilon\right) \leq \exp\left(-2\frac{\varepsilon^2 N}{(b_f - a_f)^2}\right) \quad , \tag{3a}$$

$$P\left(\tilde{\mathcal{E}}[f;\alpha^*] - \mathcal{E}[f;\alpha^*] \leq -\varepsilon\right) \leq \exp\left(-2\frac{\varepsilon^2 N}{(b_f - a_f)^2}\right) \quad . \tag{3b}$$

Quality of the estimate

For large numbers of samples N, grossly outlying estimates are unlikely.

Hoeffding's inequality [Hoeffding, 1963] yields

$$P\left(\tilde{\mathcal{E}}[f;\alpha^*] - \mathcal{E}[f;\alpha^*] \geq +\varepsilon\right) \leq \exp\left(-2\frac{\varepsilon^2 N}{(b_f - a_f)^2}\right) \quad , \tag{3a}$$

$$P\left(\tilde{\mathcal{E}}[f;\alpha^*] - \mathcal{E}[f;\alpha^*] \leq -\varepsilon\right) \leq \exp\left(-2\frac{\varepsilon^2 N}{(b_f - a_f)^2}\right) \quad . \tag{3b}$$

- Thus, SMC can be used for determining (with confidence) whether an instance of a PPHA, i.e., a PHA, satisfies design objective C.
  - Build a formula determining whether *all* the ε neighbourhood of the empirical mean satisfies C; check by SMT solving. E.g.,

    $$\texttt{unsat?} \quad \mathcal{E}_f \in B_\varepsilon(\mathcal{E}[\tilde{f}, \alpha^*]) \wedge \neg C$$
- The multi-objective parameter fitting problem can then in principle be solved by sampling the parameter space.

Quality of the estimate

For large numbers of samples N, grossly outlying estimates are unlikely.

Hoeffding's inequality [Hoeffding, 1963] yields

$$P\left(\tilde{\mathcal{E}}[f;\alpha^*] - \mathcal{E}[f;\alpha^*] \geq +\varepsilon\right) \leq \exp\left(-2\frac{\varepsilon^2 N}{(b_f - a_f)^2}\right) \quad , \qquad \text{(3a)}$$

$$P\left(\tilde{\mathcal{E}}[f;\alpha^*] - \mathcal{E}[f;\alpha^*] \leq -\varepsilon\right) \leq \exp\left(-2\frac{\varepsilon^2 N}{(b_f - a_f)^2}\right) \quad . \qquad \text{(3b)}$$

- Thus, SMC can be used for determining (with confidence) whether an instance of a PPHA, i.e., a PHA, satisfies design objective C.
  - Build a formula determining whether *all* the ε neighbourhood of the empirical mean satisfies C; check by SMT solving. E.g.,

    `unsat?` $\mathcal{E}_f \in B_\varepsilon(\mathcal{E}[\tilde{f}, \alpha^*]) \wedge \neg C$
- The multi-objective parameter fitting problem can then in principle be solved by sampling the parameter space.
- But this approach is plagued by the curse of dimensionality; instead need a constructive form of generalizing from samples.

Importance Sampling

The classical, non-symbolic version

Importance sampling [Hammersley, 1954]

An estimate for the expectation of f wrt. distribution $p(\cdot, \alpha)$ can be obtained by sampling X wrt. a different ("proposal") distribution q:

$$\begin{aligned}
\mathcal{E}[f;\alpha] &= \sum_{x \in X} f(x) p(x;\alpha) \\
&= \sum_{x \in X} \underbrace{f(x) \frac{p(x;\alpha)}{q(x)}}_{g(x,\alpha)} q(x) \\
&\approx \frac{1}{N} \sum_{i=1}^{N} \overbrace{f(x_i) \frac{p(x_i;\alpha)}{q(x_i)}} \qquad \text{where } x_i \sim q \qquad (4a) \\
&=: \hat{\mathcal{E}}[f;\alpha] \qquad (4b)
\end{aligned}$$

Importance sampling [Hammersley, 1954]

An estimate for the expectation of f wrt. distribution $p(\cdot, \alpha)$ can be obtained by sampling X wrt. a different ("proposal") distribution q:

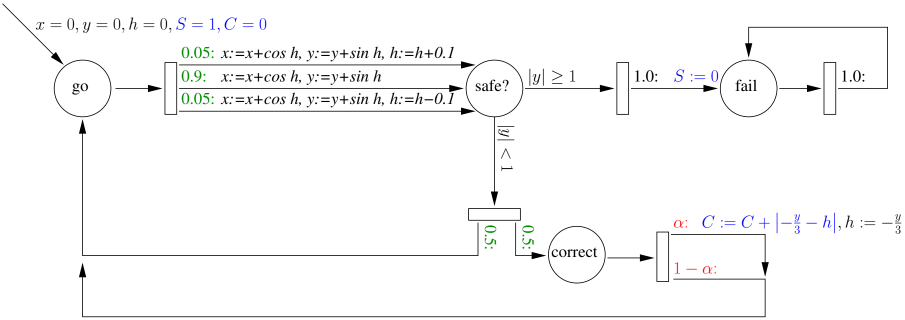
$$\begin{aligned}
\mathcal{E}[f;\alpha] &= \sum_{x \in X} f(x) p(x;\alpha) \\
&= \sum_{x \in X} \underbrace{f(x) \frac{p(x;\alpha)}{q(x)}}_{g(x,\alpha)} q(x) \\
&\approx \frac{1}{N} \sum_{i=1}^{N} \overbrace{f(x_i) \frac{p(x_i;\alpha)}{q(x_i)}} \qquad \text{where } x_i \sim q \qquad \text{(4a)} \\
&=: \hat{\mathcal{E}}[f;\alpha] \qquad \text{(4b)}
\end{aligned}$$

Note that samples $\{x_1, \ldots, x_N\}$ are drawn according to the substitute distribution q; nevertheless, (4a–4b) permits to compute estimates $\hat{\mathcal{E}}[f;\alpha]$ for arbitrary values of α.

Symbolic Importance Sampling

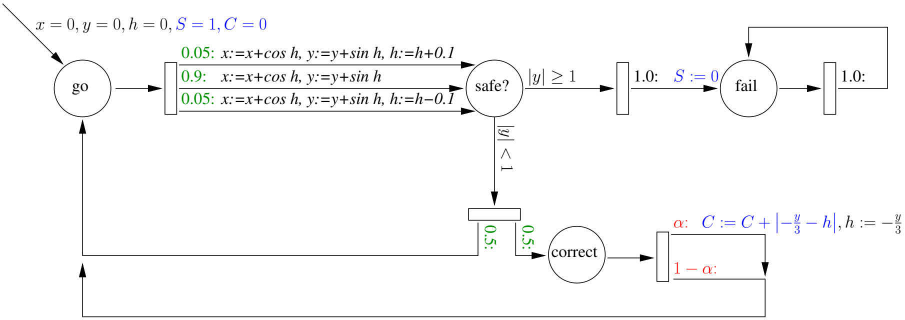
Mining (not yet PAC) Formal Models

Importance sampling in a PPHA



Pursue a simulation with a *concrete* substitute probability q replacing α.

Importance sampling in a PPHA



Pursue a simulation with a *concrete* substitute probability q replacing α.

Assume simulation yields a run taking the α branch n times and the $(1-\alpha)$ branch m times. Then

- the probability of this run is $c \cdot q^n \cdot (1-q)^m$ in the simulation,
- the probability of this run is $c \cdot \alpha^n \cdot (1-\alpha)^m$ in the PPHA, *for arbitrary* α.

Here, c denotes the accumulated probability of all other choices along the run.

Symbolic importance sampling

$t_1, \ldots, t_l$ be the parameter-dependent probability terms in the PPHA A.
Let $\#_i t_j$ denote the number of times the t_j branch was taken in run x_i when simulating A with the substitute parameterization q.

Symbolic importance sampling

$t_1, \ldots, t_l$ be the parameter-dependent probability terms in the PPHA A. Let $\#_i t_j$ denote the number of times the t_j branch was taken in run x_i when simulating A with the substitute parameterization q.

A symbolic representation of the parameter dependency of $\hat{\mathcal{E}}[f;\alpha]$ can be obtained from importance sampling (4a–4b):

$$\hat{\mathcal{E}}[f;\alpha] = \underbrace{\frac{1}{N}\sum_{i=1}^{N} f(x_i) \prod_{j=1}^{l} \left(\frac{t_j}{t_j[q/\alpha]}\right)^{\#_i t_j}}_{\eta_f} \tag{5}$$

Note that $f(x_i)$, $t_j[q/\alpha]$ and $\#_i t_j$ are constants s.t. the only free variables occurring in η_f are the parameters $\alpha_1, \ldots, \alpha_k$ within terms $t_1, \ldots, t_l$.

Parameterization

- Term η_f in (5) is a large sum of products with multiple occurrences of parameters α_i within different instances of sub-terms t_j.
- Let C be a constraint over $\mathcal{E}_{f_1}, \ldots, \mathcal{E}_{f_n}$ formalizing the design objective.
- Let ϕ be the constraint on admissible parameterizations α.

Parameterization

- Term η_f in (5) is a large sum of products with multiple occurrences of parameters α_i within different instances of sub-terms t_j.
- Let C be a constraint over $\mathcal{E}_{f_1}, \ldots, \mathcal{E}_{f_n}$ formalizing the design objective.
- Let ϕ be the constraint on admissible parameterizations α.

A parameter instance $\alpha \models \phi$ guaranteeing C can now in principle be found — or conversely, the infeasibility of C over ϕ be established — by solving the constraint system

$$\underbrace{\phi}_{\text{parameter range}} \wedge \underbrace{\left(\bigwedge_{i=1}^{n} \mathcal{E}_{f_i} = \eta_{f_i} \right)}_{\text{parameter dependency of expectations}} \wedge \underbrace{C}_{\text{design objective}} \qquad (6)$$

using an appropriate constraint solver.

Parameterization

- Term η_f in (5) is a large sum of products with multiple occurrences of parameters α_i within different instances of sub-terms t_j.
- Let C be a constraint over $\mathcal{E}_{f_1}, \ldots, \mathcal{E}_{f_n}$ formalizing the design objective.
- Let ϕ be the constraint on admissible parameterizations α.

A parameter instance $\alpha \models \phi$ guaranteeing C can now in principle be found — or conversely, the infeasibility of C over ϕ be established — by solving the constraint system

$$\underbrace{\phi}_{\text{parameter range}} \wedge \underbrace{\left(\bigwedge_{i=1}^{n} \mathcal{E}_{f_i} \in B_{\varepsilon(\|\alpha - q\|, N)}(\eta_{f_i}) \right)}_{\text{parameter dependency of expectations}} \wedge \underbrace{C}_{\text{design objective}} \tag{6}$$

using an appropriate constraint solver.

Parameterization

- Term η_f in (5) is a large sum of products with multiple occurrences of parameters α_i within different instances of sub-terms t_j.
- Let C be a constraint over $\mathcal{E}_{f_1}, \ldots, \mathcal{E}_{f_n}$ formalizing the design objective.
- Let ϕ be the constraint on admissible parameterizations α.

A parameter instance $\alpha \models \phi$ guaranteeing C can now in principle be found — or conversely, the infeasibility of C over ϕ be established — by solving the constraint system

$$\underbrace{\phi}_{\text{parameter range}} \wedge \underbrace{\left(\bigwedge_{i=1}^{n} \mathcal{E}_{f_i} \in B_{\varepsilon(\|\alpha - q\|, N)}(\eta_{f_i}) \right)}_{\text{parameter dependency of expectations}} \wedge \underbrace{C}_{\text{design objective}} \tag{6}$$

using an appropriate constraint solver.

Caveat: Existence of α satisfying (6) is a necessary, though not sufficient condition for it satisfying the design goal with confidence.

(Will deal with that issue later.)

Finding Feasible Parameter Instances

Polynomial constraint solving of very high order

The shape of the constraint formulae

- Constraint (6), i.e., $\phi \wedge \left(\bigwedge_{i=1}^{n} \mathcal{E}_{f_i} \in B_{\varepsilon(\|\alpha - q\|, N)}(\eta_{f_i})\right) \wedge C$, is an arithmetic constraint involving
  1. addition, multiplication, exponentiation by (large!) integer constants,
  2. the operations found in the terms $t_1, \dots, t_l$ defining the parameter dependency $p(\alpha)$ of the Markov chain,
  3. the operations occurring in the parameter domain constraint ϕ and in the design goal C,
- it can be solved by SMT solvers addressing the corresponding subset of arithmetic, e.g. iSAT.[1] [2]

[1] iSAT [F., Herde, Ratschan, Schubert, Teige, 2007–] is an algorithms integrating interval constraint propagation and SAT modulo theory for solving constraint systems over $\mathbb{R}, +, *, \sin, \exp, \dots$

[2] You ought to refine iSAT's standard settings for accuracy, though.

A simple instance of the constraint formulae

```
EXPR
  ...
-- X236 represents 23 sample(s) of average reward -0.434783
  X236 = -28493.9 * alpha**6 * (1-alpha)**10;
-- X235 represents 12 sample(s) of average reward -0.666667
  X235 = -21845.3 * alpha**6 * (1-alpha)**9;
-- X234 represents 35 sample(s) of average reward -0.2
  X234 = -13107.2 * alpha**9 * (1-alpha)**7;
-- X233 represents 39 sample(s) of average reward -0.0512821
  X233 = -13443.3 * alpha**7 * (1-alpha)**11;
  ...

-- Computing empirical expectation E.
  E = 0.00025 * (X1 + X2 + X3 + ... + X236 + X237 + X238 + X239);

-- Optimization target is
  (-0.01 <= E) and (E <= 0.0);

-- Parameter constraint is
  (alpha < 0.0125) or (alpha > 0.99);
```

A simple instance of the constraint formulae

```
EXPR
  ...
-- X236 represents 23 sample(s) of average reward -0.434783
  X236 = -28493.9 * alpha**6 * (1-alpha)
-- X235 represents 12 sample(s) of avera
  X235 = -21845.3 * alpha**6 * (1-alpha)
-- X234 represents 35 sample(s) of avera
  X234 = -13107.2 * alpha**9 * (1-alpha)
-- X233 represents 39 sample(s) of average reward
  X233 = -13443.3 * alpha**7 * (1-alpha)**11;
  ...

-- Computing empirical expectation E.
  E = 0.00025 * (X1 + X2 + X3 + ... + X2

-- Optimization target is
  (-0.01 <= E) and (E <= 0.0);

-- Parameter constraint is
  (alpha < 0.0125) or (alpha > 0.99);
```

Terms over parameters can
- involve multiple different parameters,
- involve linear, polynomial, and transcendental arithmetic.

Expectations and parameters may be
- multi-dimensional,
- subject to arbitrary Boolean combinations of constraints,
- subject to non-polynomial arithmetic constraints.

How iSAT works [Herde, 2010]

c_1 : $(\neg a \lor \neg c \lor d)$

c_2 : $\land\ (\neg a \lor \neg b \lor c)$

c_3 : $\land\ (\neg c \lor \neg d)$

c_4 : $\land\ (b \lor x \geq -2)$

c_5 : $\land\ (x \geq 4 \lor y \leq 0 \lor h_3 \geq 6.2)$

c_6 : $\land\ h_1 = x^2$

c_7 : $\land\ h_2 = -2 \cdot y$

c_8 : $\land\ h_3 = h_1 + h_2$

- Use Tseitin-style (i.e. definitional) transformation to rewrite input formula into a conjunction of constraints:
  - ▷ n-ary disjunctions of bounds
  - ▷ arithmetic constraints having at most one operation symbol
- Boolean variables are regarded as 0-1 integer variables. Allows identification of literals with bounds on Booleans:

$$b \equiv b \geq 1$$
$$\neg b \equiv b \leq 0$$

- Float variables h_1, h_2, h_3 are used for decomposition of complex constraint $x^2 - 2y \geq 6.2$.

How iSAT works

[Herde, 2010]

c_1 : $(\neg a \lor \neg c \lor d)$

c_2 : $\land\ (\neg a \lor \neg b \lor c)$

c_3 : $\land\ (\neg c \lor \neg d)$

c_4 : $\land\ (b \lor x \geq -2)$

c_5 : $\land\ (x \geq 4 \lor y \leq 0 \lor h_3 \geq 6.2)$

c_6 : $\land\ h_1 = x^2$

c_7 : $\land\ h_2 = -2 \cdot y$

c_8 : $\land\ h_3 = h_1 + h_2$

DL 1: $a \geq 1$

How iSAT works [Herde, 2010]

$c_1: \quad (\neg a \vee \neg c \vee d)$

$c_2: \quad \wedge\ (\neg a \vee \neg b \vee c)$

$c_3: \quad \wedge\ (\neg c \vee \neg d)$

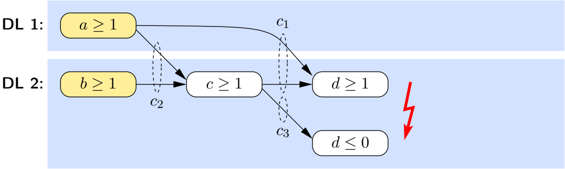
$c_4: \quad \wedge\ (b \vee x \geq -2)$

$c_5: \quad \wedge\ (x \geq 4 \vee y \leq 0 \vee h_3 \geq 6.2)$

$c_6: \quad \wedge\ h_1 = x^2$

$c_7: \quad \wedge\ h_2 = -2 \cdot y$

$c_8: \quad \wedge\ h_3 = h_1 + h_2$



How iSAT works

[Herde, 2010]

$c_1 :\quad (\neg a \vee \neg c \vee d)$

$c_2 :\quad \wedge\ (\neg a \vee \neg b \vee c)$

$c_3 :\quad \wedge\ (\neg c \vee \neg d)$

$c_4 :\quad \wedge\ (b \vee x \geq -2)$

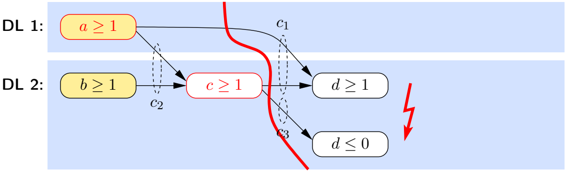
$c_5 :\quad \wedge\ (x \geq 4 \vee y \leq 0 \vee h_3 \geq 6.2)$

$c_6 :\quad \wedge\ h_1 = x^2$

$c_7 :\quad \wedge\ h_2 = -2 \cdot y$

$c_8 :\quad \wedge\ h_3 = h_1 + h_2$

$c_9 :\quad \wedge\ (\neg a \vee \neg c)$



How iSAT works

[Herde, 2010]

c_1 : $(\neg a \lor \neg c \lor d)$

c_2 : $\land\ (\neg a \lor \neg b \lor c)$

c_3 : $\land\ (\neg c \lor \neg d)$

c_4 : $\land\ (b \lor x \geq -2)$

c_5 : $\land\ (x \geq 4 \lor y \leq 0 \lor h_3 \geq 6.2)$

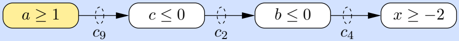
c_6 : $\land\ h_1 = x^2$

c_7 : $\land\ h_2 = -2 \cdot y$

c_8 : $\land\ h_3 = h_1 + h_2$

c_9 : $\land\ (\neg a \lor \neg c)$

DL 1:



How iSAT works [Herde, 2010]

$c_1 : \quad (\neg a \lor \neg c \lor d)$

$c_2 : \quad \land\ (\neg a \lor \neg b \lor c)$

$c_3 : \quad \land\ (\neg c \lor \neg d)$

$c_4 : \quad \land\ (b \lor x \geq -2)$

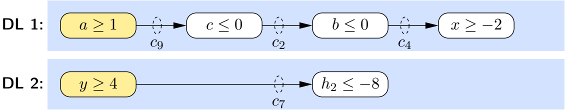
$c_5 : \quad \land\ (x \geq 4 \lor y \leq 0 \lor h_3 \geq 6.2)$

$c_6 : \quad \land\ h_1 = x^2$

$c_7 : \quad \land\ h_2 = -2 \cdot y$

$c_8 : \quad \land\ h_3 = h_1 + h_2$

$c_9 : \quad \land\ (\neg a \lor \neg c)$



How iSAT works [Herde, 2010]

$c_1: \quad (\neg a \vee \neg c \vee d)$

$c_2: \quad \wedge\ (\neg a \vee \neg b \vee c)$

$c_3: \quad \wedge\ (\neg c \vee \neg d)$

$c_4: \quad \wedge\ (b \vee x \geq -2)$

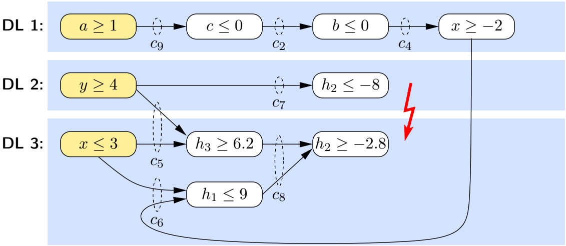
$c_5: \quad \wedge\ (x \geq 4 \vee y \leq 0 \vee h_3 \geq 6.2)$

$c_6: \quad \wedge\ h_1 = x^2$

$c_7: \quad \wedge\ h_2 = -2 \cdot y$

$c_8: \quad \wedge\ h_3 = h_1 + h_2$

$c_9: \quad \wedge\ (\neg a \vee \neg c)$



How iSAT works [Herde, 2010]

c_1 : $(\neg a \lor \neg c \lor d)$

c_2 : $\land\ (\neg a \lor \neg b \lor c)$

c_3 : $\land\ (\neg c \lor \neg d)$

c_4 : $\land\ (b \lor x \geq -2)$

c_5 : $\land\ (x \geq 4 \lor y \leq 0 \lor h_3 \geq 6.2)$

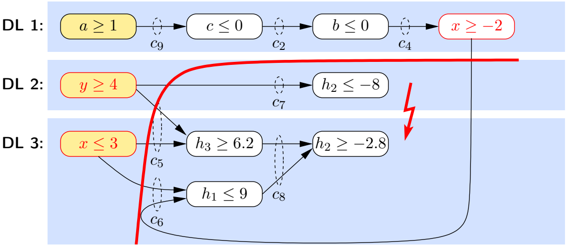
c_6 : $\land\ h_1 = x^2$

c_7 : $\land\ h_2 = -2 \cdot y$

c_8 : $\land\ h_3 = h_1 + h_2$

c_9 : $\land\ (\neg a \lor \neg c)$

c_{10} : $\land\ (x < -2 \lor y < 3 \lor x > 3)$



← conflict clause = symbolic description
of a rectangular region of the search space
which is excluded from future search

How iSAT works

[Herde, 2010]

$c_1: \quad (\neg a \lor \neg c \lor d)$

$c_2: \quad \land\ (\neg a \lor \neg b \lor c)$

$c_3: \quad \land\ (\neg c \lor \neg d)$

$c_4: \quad \land\ (b \lor x \geq -2)$

$c_5: \quad \land\ (x \geq 4 \lor y \leq 0 \lor h_3 \geq 6.2)$

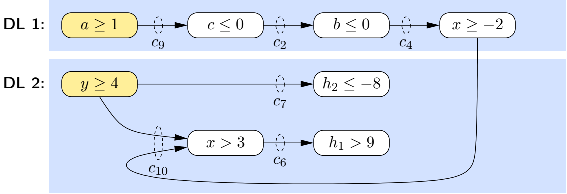
$c_6: \quad \land\ h_1 = x^2$

$c_7: \quad \land\ h_2 = -2 \cdot y$

$c_8: \quad \land\ h_3 = h_1 + h_2$

$c_9: \quad \land\ (\neg a \lor \neg c)$

$c_{10}: \quad \land\ (x < -2 \lor y < 3 \lor x > 3)$



How iSAT works [Herde, 2010]

$c_1 :\quad (\neg a \ \vee\ \neg c \ \vee\ d)$

$c_2 :\quad \wedge\ (\neg a \ \vee\ \neg b \ \vee\ c)$

$c_3 :\quad \wedge\ (\neg c \ \vee\ \neg d)$

$c_4 :\quad \wedge\ (b \ \vee\ x \geq -2)$

$c_5 :\quad \wedge\ (x \geq 4 \ \vee\ y \leq 0 \ \vee\ h_3 \geq 6.2)$

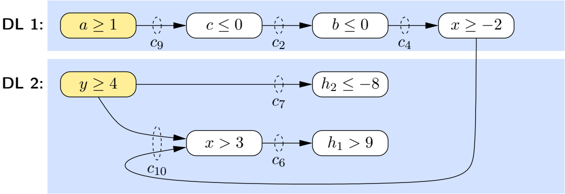
$c_6 :\quad \wedge\ h_1 = x^2$

$c_7 :\quad \wedge\ h_2 = -2 \cdot y$

$c_8 :\quad \wedge\ h_3 = h_1 + h_2$

$c_9 :\quad \wedge\ (\neg a \ \vee\ \neg c)$

$c_{10} :\quad \wedge\ (x < -2 \ \vee\ y < 3 \ \vee\ x > 3)$



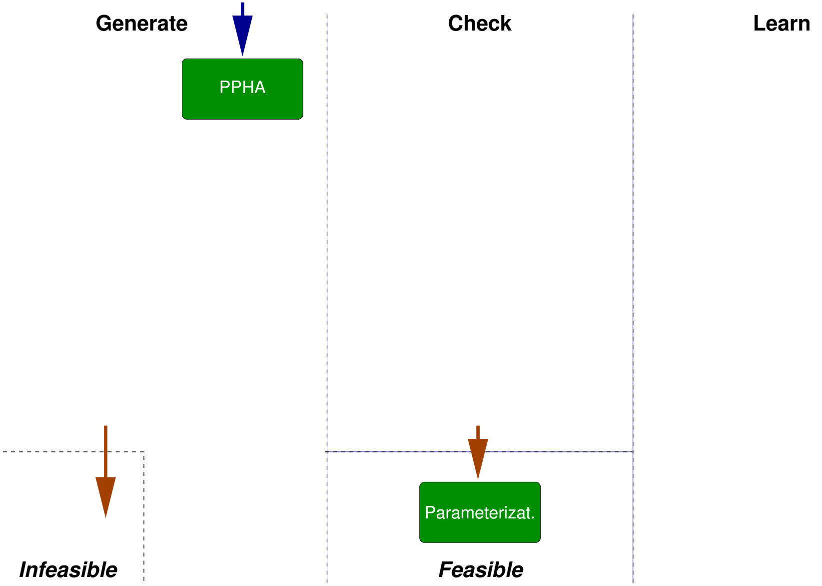
- Continue do split and deduce until either
  - ▷ formula turns out to be UNSAT (unresolvable conflict)
  - ▷ solver is left with 'sufficiently small' portion of the search space for which it cannot derive any contradiction

- Avoid infinite splitting and deduction:
  - ▷ minimal splitting width
  - ▷ discard a deduced bound if it yields small progress only

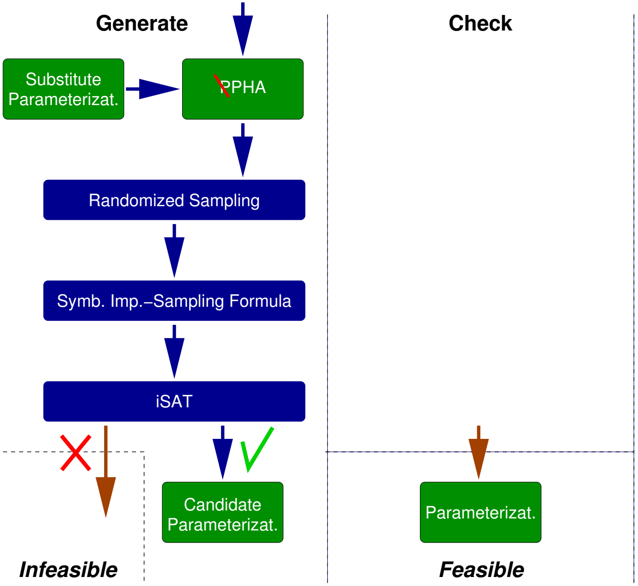
Becoming PAC: Iterative Refinement of the Encoding

Dealing with the approximation error incurred by importance sampling

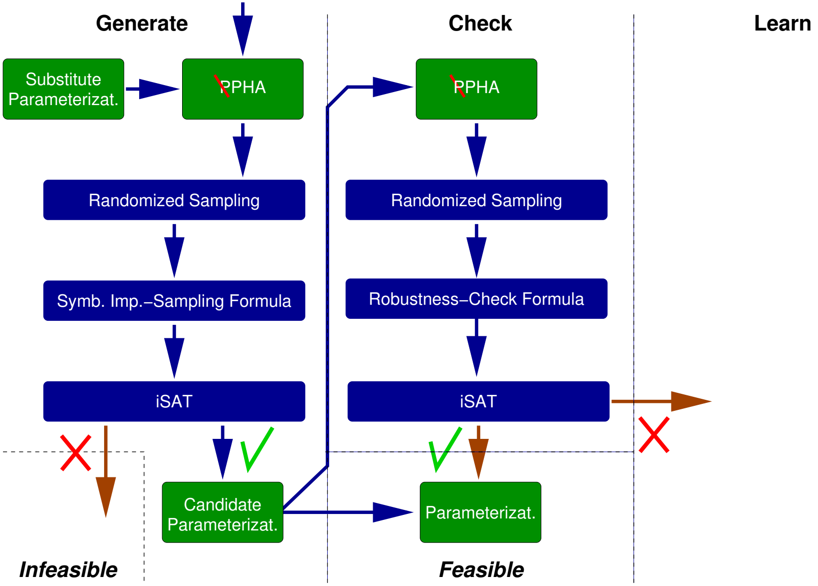
Learning from Counterexamples



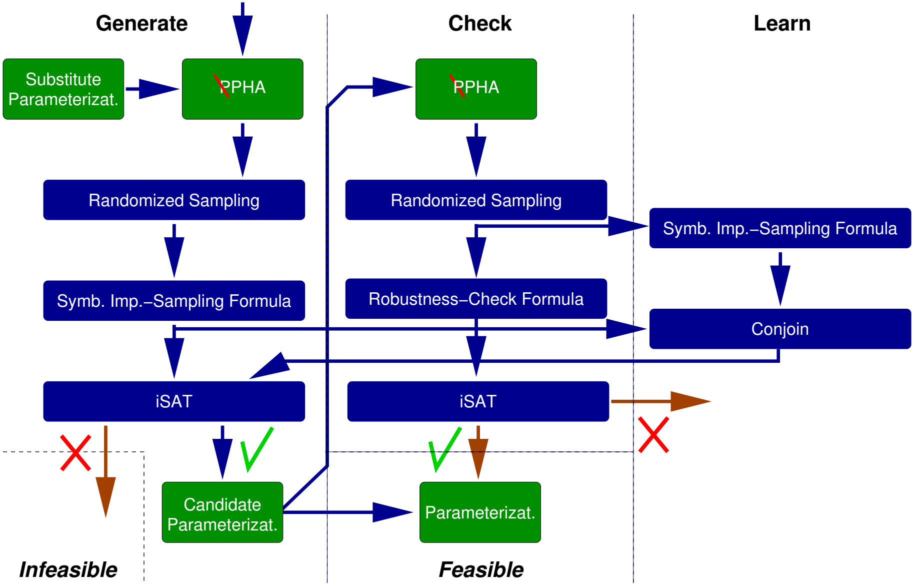
Learning from Counterexamples



Learning from Counterexamples



Learning from Counterexamples



Algorithm Properties

Let P be the user-required confidence and let the number N of samples drawn in each round be selected according to the Hoeffding bound (3).

Correctness

If the algorithm terminates, the following properties hold with confidence $\geq P$:

1. If it reports "Feasible" then the parameter instance provided yields expectations satisfying C.
2. If it reports "Infeasible" then for any parameter instance satisfying ϕ, the associated expectations violate C.

Discussion

What we did

Solved a complex design-space exploration problem by
(iterative) automated learning of a tractable, PAC formal model.

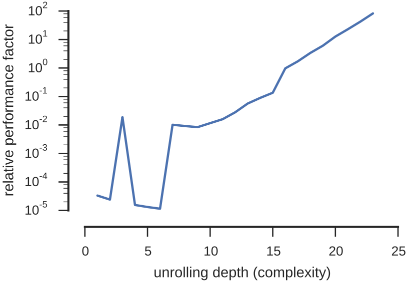
- Approach is based on an alternation of *sampling, generalization, constraint generation, SMT solving*

What we did

Solved a complex design-space exploration problem by
(iterative) automated learning of a tractable, PAC formal model.

- Approach is based on an alternation of *sampling, generalization, constraint generation, SMT solving*

- Closed-form representation based on SMT formulae well exists, but
  - exponentially sized formulae,
  - thus not scalable.

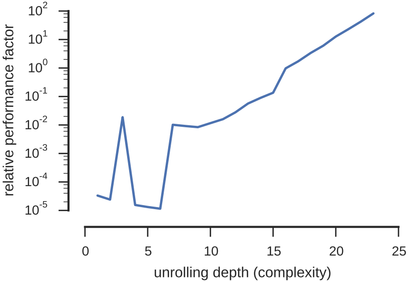


What we did

Solved a complex design-space exploration problem by
(iterative) automated learning of a tractable, PAC formal model.

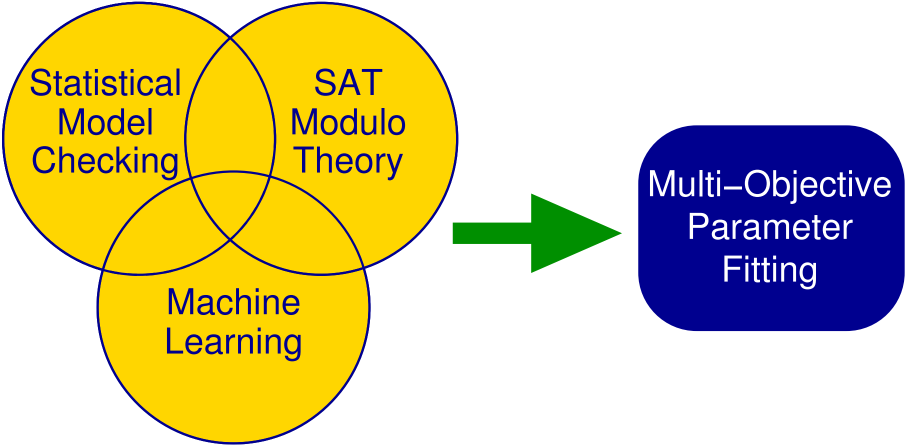
- Approach is based on an alternation of *sampling, generalization, constraint generation, SMT solving*

- Closed-form representation based on SMT formulae well exists, but
  - exponentially sized formulae,
  - thus not scalable.

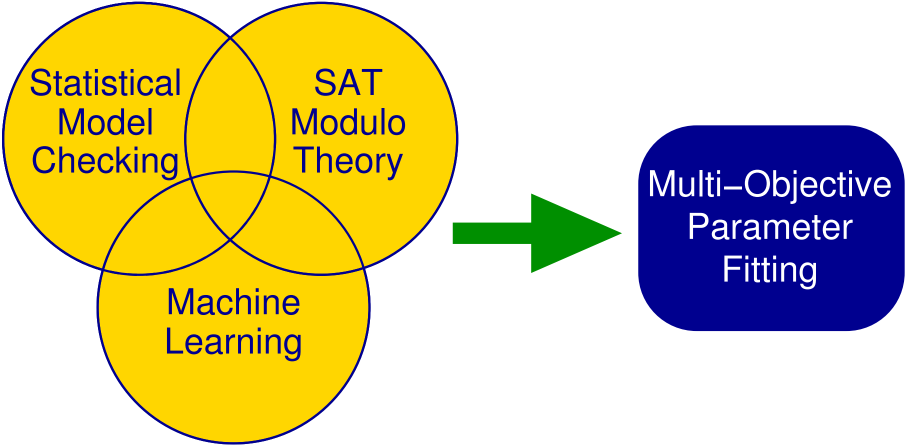


- A prototype implementation of our approach exists
  (result of an excellent BSc thesis — thank you, Paul).

The major ingredients



The major ingredients



Many more such combinations wait to be explored!

Let us go beyond...