From Reasoning with Constraints to Mining Constraints:

Multi-Objective Parameter Fitting in Parametric Probabilistic Hybrid Automata

Martin Fränzle[1]

joint work (in progress) 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

Why this Talk in a WS on Constraint Solving?

- Traditional symbolic verification assumes that the analysis problem features a well-understood, closed-form symbolic representation, facilitating constraint-based analysis:

Why this Talk in a WS on Constraint Solving?

- Traditional symbolic verification assumes that the analysis problem features a well-understood, closed-form symbolic representation, facilitating constraint-based analysis:

- To me, this preoccupation to classical symbolic methods seems to prevent some fruitful applications of constraint-based analysis.
  - What happens, e.g., if the constraint representation is learnt from samples, thus blending machine learning with constraint solving?
- This talk is intended as a motivating example.

Why this Talk in a WS on Constraint Solving?

- Traditional symbolic verification assumes that the analysis problem features a well-understood, closed-form symbolic representation, facilitating constraint-based analysis:

- To me, this preoccupation to classical symbolic methods seems to prevent some fruitful applications of constraint-based analysis.
  - What happens, e.g., if the constraint representation is learnt from samples, thus blending machine learning with constraint solving?
- This talk is intended as a motivating example.

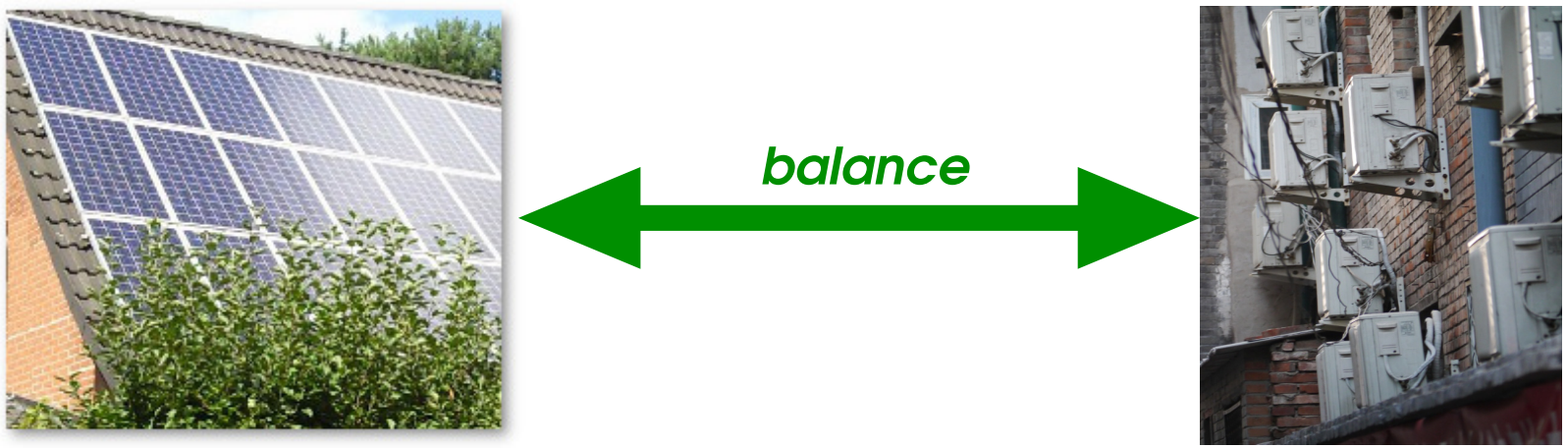
$\rightsquigarrow$ Today, I will thus not talk about

- SMT solving for arithmetic constraints involving ODE (iSAT-ODE),
- SMT solving for stochastic arithmetic constraint systems (SiSAT)
- and just briefly about SMT solving for arithmetic constraints beyond the polynomial fragment (iSAT).

Example: Demand-Response Schemes in Smart Grids

A Practical Problem Featuring Hybrid Dynamics

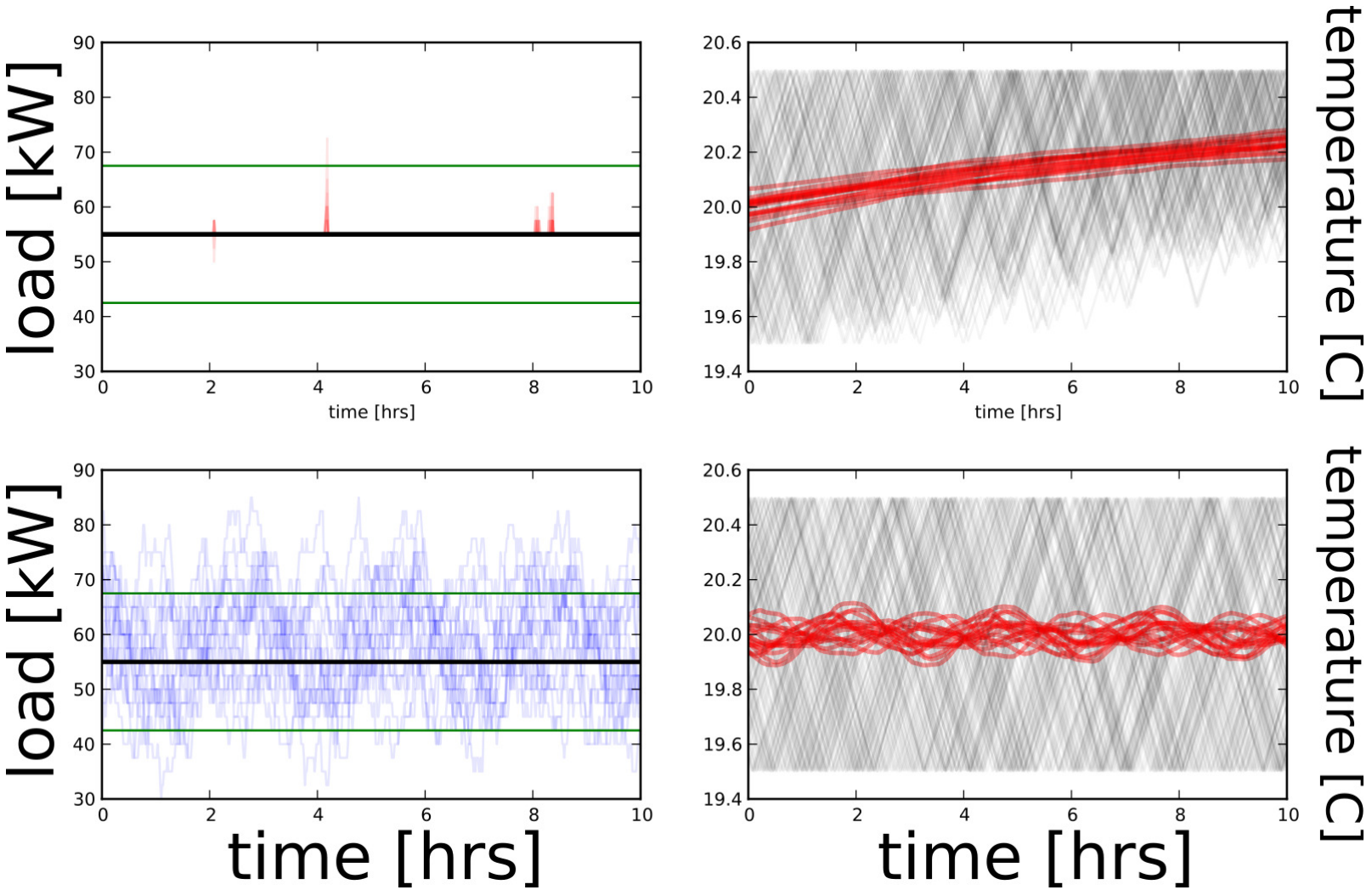
Demand Response: Supplying Reserve Power by Thermostatically Ctrl.ed Loads (TCLs) [Callaway 2009]



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

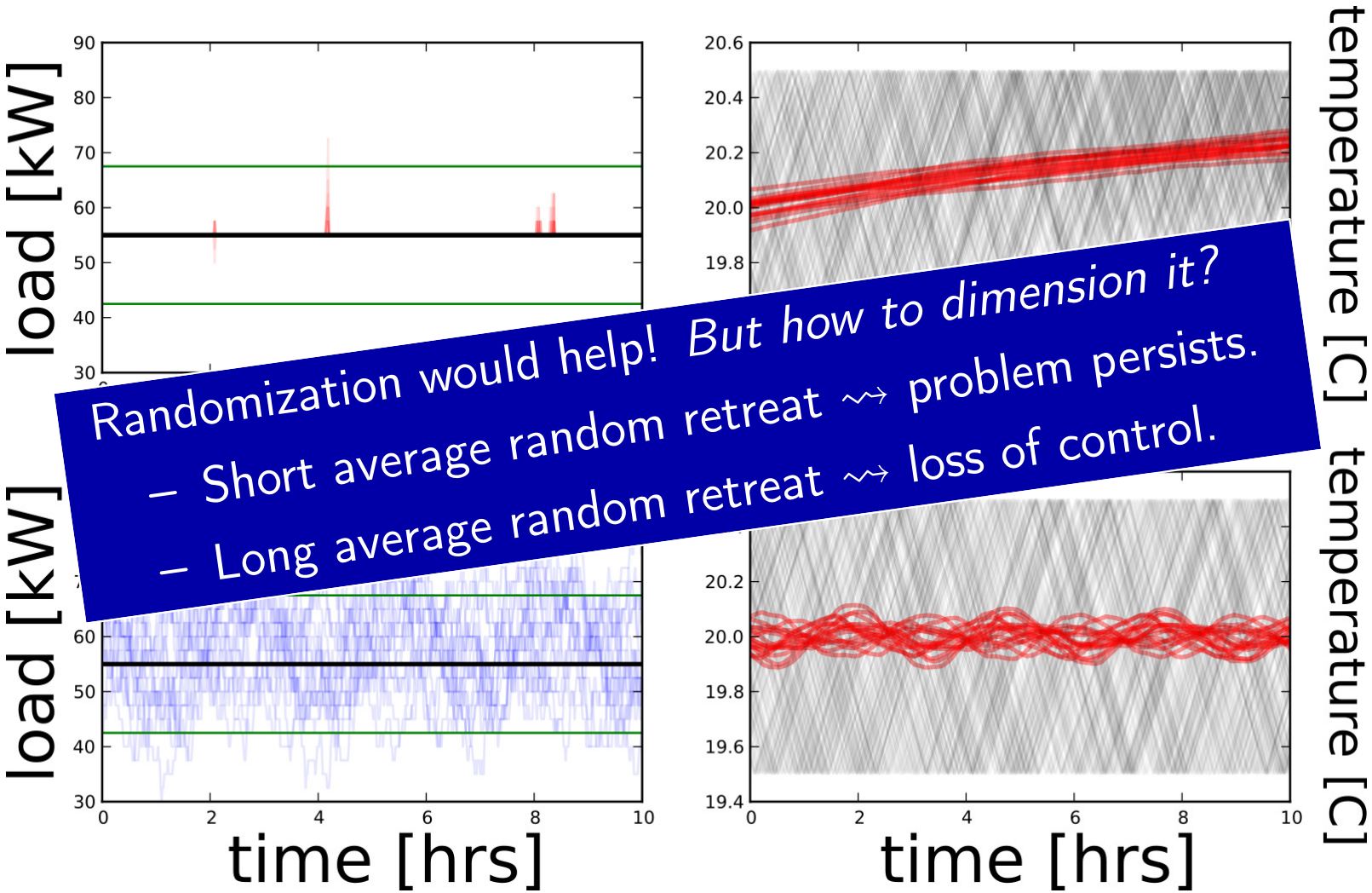
- On power shortage, provide reserve power by switching off early / switching on late.
- On excess power, consume reserve power by switching off late / switching on early.
- Unnoticeable to residents due to marginal adjustments to switching thresholds.

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

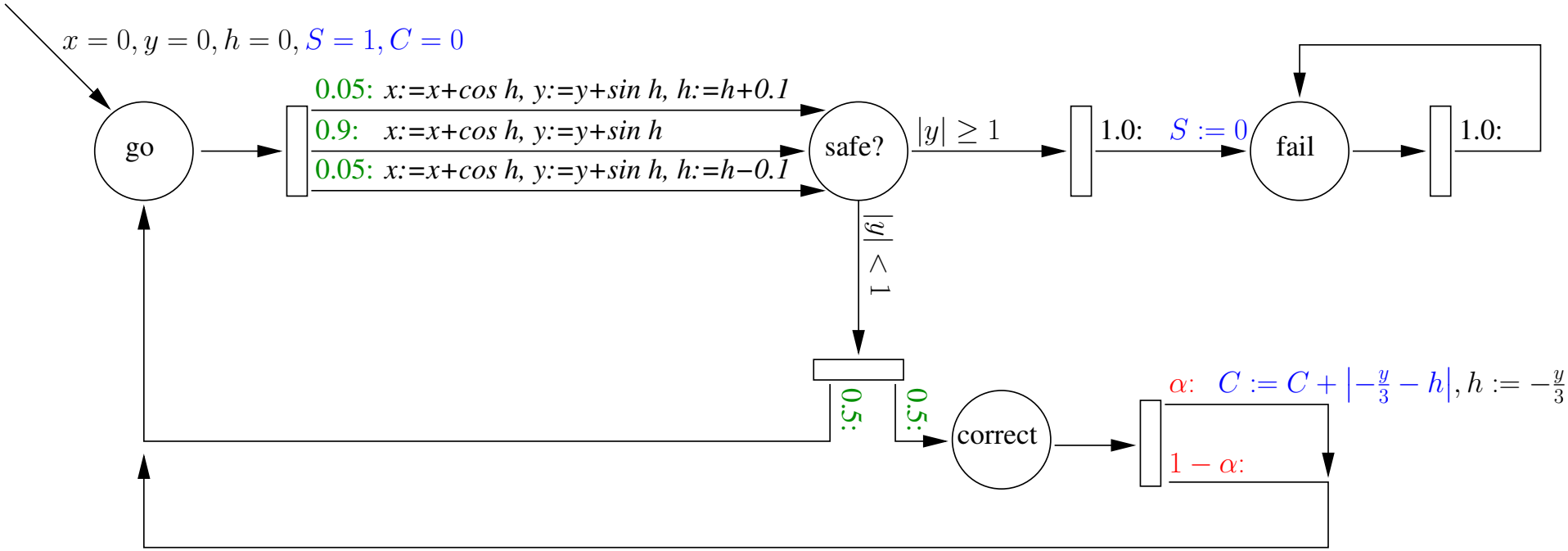


Externally controlled (power target 55 kW) vs. uncontrolled ensemble.
Control strategy: switch off coldest households if power target exceeded.

The Formal Model

Parametric Probabilistic HA

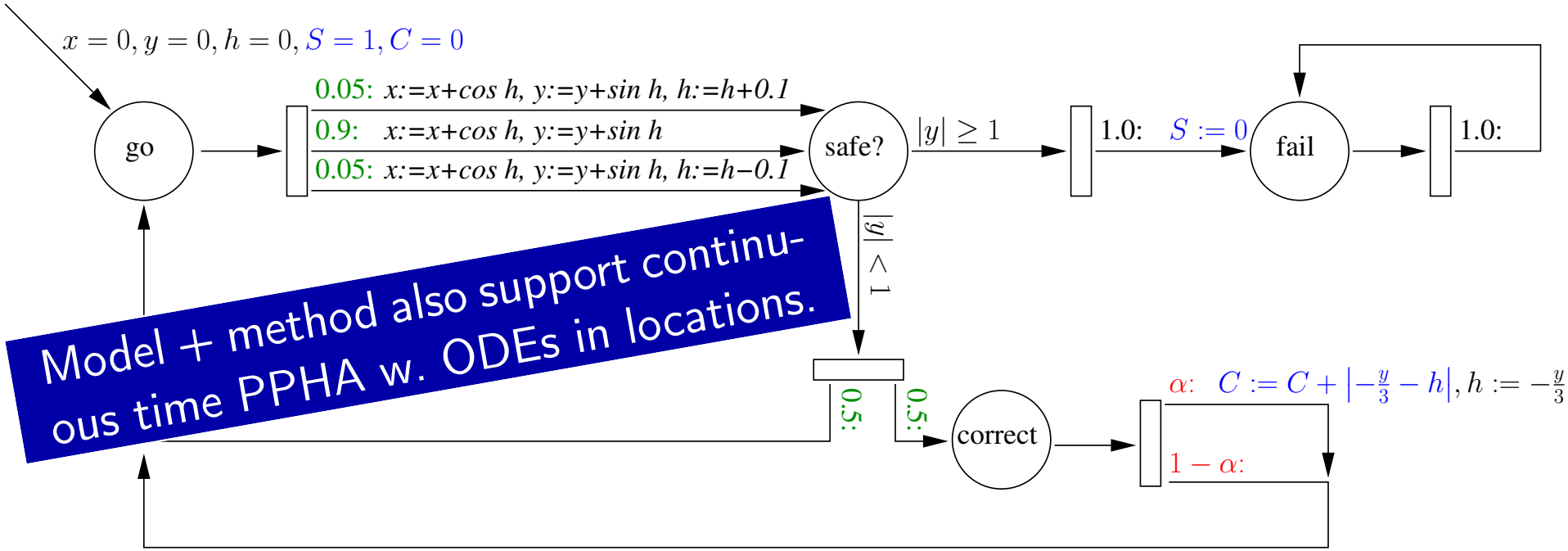
A (discrete time) Parametric Probabilistic HA



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 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 corrective actions.
- Two reward / cost variables:
  - C records accumulated cost of corrective steering actions,
  - S records successful stay in lane.

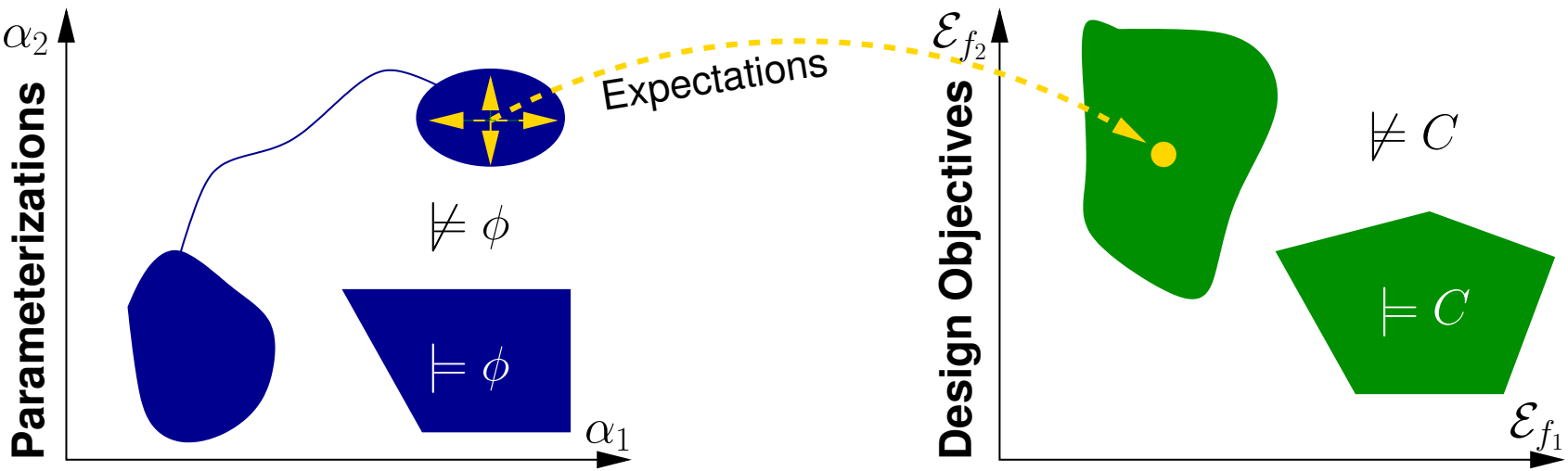
The problem

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 $\theta \in \mathbb{R}^k$ that

1. satisfies ϕ and
2. yields expected rewards $\mathcal{E}[\vec{f}, \theta]$ satisfying C.



Parameter synthesis problem: formally

Let $f_1, \ldots, f_n : \Sigma^* \to \mathbb{R}$ be a vector of rewards in a Markov chain M and let C be a design goal in the form of a constraint on the expected rewards, i.e. an arithmetic predicate containing $\mathcal{E}_{f_1}, \ldots, \mathcal{E}_{f_n}$ as free variables. A parameter instance $\theta : \vec{\alpha} \to \mathbb{R}$ is feasible (wrt. M and C) iff

$$\theta \models \phi \quad \text{and} \quad [\mathcal{E}_{f_1} \mapsto \mathcal{E}_{M,k}(f_1;\theta), \ldots, \mathcal{E}_{f_n} \mapsto \mathcal{E}_{M,k}(f_n;\theta)] \models C.$$

The multi-objective parameter synthesis problem is to find a feasible parameter instance θ, if such exists, or to prove absence thereof otherwise.

Approach

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 Expectations by Sampling

Classical sampling

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

Expectation of f depending on θ:

$$\mathcal{E}[f;\theta] = \int_X f(x)p(x;\theta)\,\mathrm{d}x \tag{1}$$

Classical sampling

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

Expectation of f depending on θ:

$$\mathcal{E}[f;\theta] = \int_X f(x)p(x;\theta)\,\mathrm{d}x \tag{1}$$

Estimated expectation of f in θ^*:

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

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

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

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

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

Quality of the estimate

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

Hoeffding's inequality [Hoeffding 1963] yields

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

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

- Thus, sampling can be used for determining (with confidence) whether a parameterized instance of a PPHA, i.e., a PHA, satisfies the design objective C.
  - Build a formula which determines whether *all* the ε neighbourhood of the empirical means satisfies C; check by SMT solving.
- 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

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

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

- Thus, sampling can be used for determining (with confidence) whether a parameterized instance of a PPHA, i.e., a PHA, satisfies the design objective C.
  - Build a formula which determines whether *all* the ε neighbourhood of the empirical means satisfies C; check by SMT solving.
- 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

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

$$\begin{aligned}
\mathcal{E}[f;\theta] &= \int_{\mathcal{X}} f(x)p(x;\theta)\,\mathrm{d}x \\
&= \int_{X} \frac{f(x)p(x;\theta)}{q(x)} q(x)\,\mathrm{d}x \\
&\approx \frac{1}{N}\sum_{i=1}^{N} \frac{f(x_i)p(x_i;\theta)}{q(x_i)} \qquad \text{where } x_i \sim q \qquad (4a) \\
&=: \hat{\mathcal{E}}[f;\theta] \qquad (4b)
\end{aligned}$$

Importance sampling

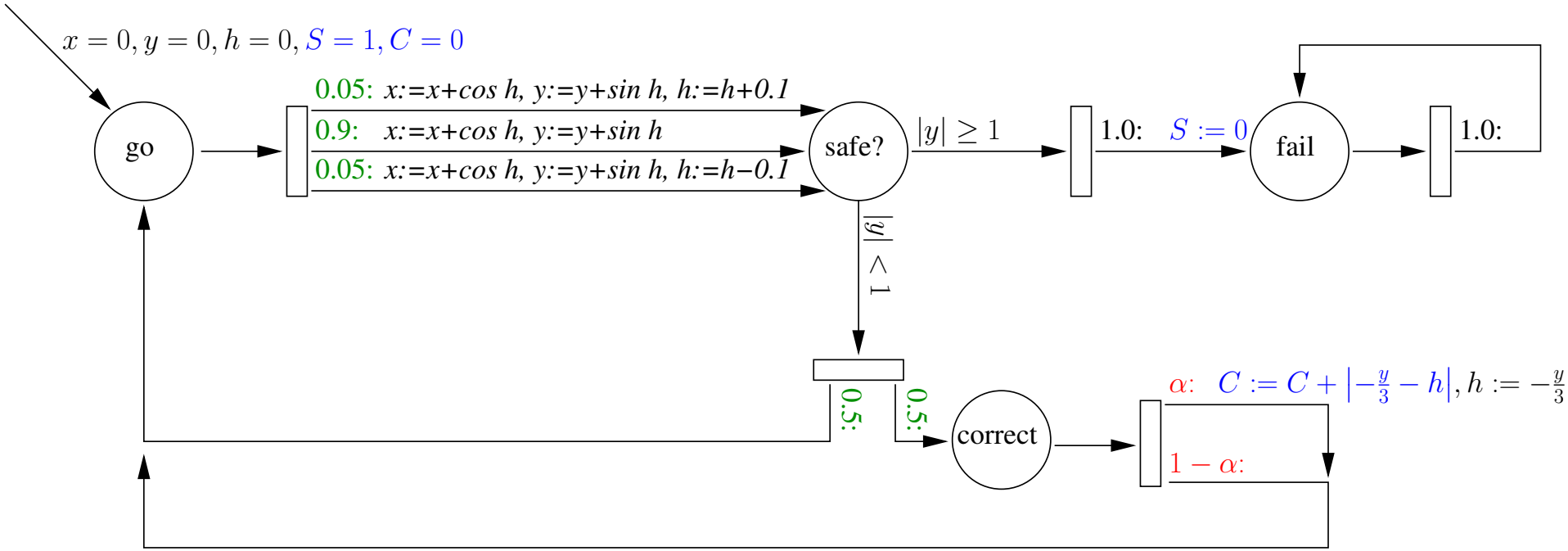
An estimate for the expectation of f wrt. distribution $p(\cdot, \theta)$ can be obtained by sampling X wrt. a different ("proposal") distribution q:

$$\begin{aligned}
\mathcal{E}[f;\theta] &= \int_{\mathcal{X}} f(x)p(x;\theta)\,\mathrm{d}x \\
&= \int_{X} \frac{f(x)p(x;\theta)}{q(x)} q(x)\,\mathrm{d}x \\
&\approx \frac{1}{N}\sum_{i=1}^{N} \frac{f(x_i)p(x_i;\theta)}{q(x_i)} \qquad \text{where } x_i \sim q \qquad \text{(4a)} \\
&=: \hat{\mathcal{E}}[f;\theta] \qquad \text{(4b)}
\end{aligned}$$

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

Symbolic Importance Sampling

Importance sampling in a PPHA



Pursue a simulation with a concrete substitute probability p replacing α.
If this simulation yields a run that has taken the α branch n times and the $1 - \alpha$ branch m times then, for arbitrary α,

- the probability of this run is $c \cdot p^n \cdot (1 - p)^m$ in the simulation with the substitute probability,
- the probability of this run is $c \cdot \alpha^n \cdot (1 - \alpha)^m$ in the PPHA.

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

Towards symbolic importance sampling

Let $t_1, \ldots, t_l$ be the parameter-dependent probability terms occurring in the PPHA A.
For each of the N samples $x_1, \ldots, x_N$ obtained by simulating A with the substitute parameterization θ^*, let $\#t_i$ denote the number of times the t_i branch was taken in run x_i.

Towards symbolic importance sampling

Let $t_1, \ldots, t_l$ be the parameter-dependent probability terms occurring in the PPHA A.
For each of the N samples $x_1, \ldots, x_N$ obtained by simulating A with the substitute parameterization θ^*, let $\#t_i$ denote the number of times the t_i branch was taken in run x_i.

A symbolic representation of the parameter dependency of $\hat{\mathcal{E}}[f;\theta]$ can now readily be obtained from (4a–4b) as follows:

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

Note that $f(x_i)$, $t_j[\theta^*/\theta]$ and $\#t_{j\,i}$ are constants s.t. the only free variables occurring in η_f are the parameters $\alpha_1, \ldots, \alpha_k$ within the terms $t_1, \ldots, t_l$.

Parameterization

- Term η_f in (5) is a large sum with multiple occurrences of parameters θ within different instances of sub-terms t_j.
- Let C be a constraint on the expected rewards for $\vec{f}$, i.e., C is a formula with free variable $\mathcal{E}_f$ formalizing the requirements on the expectation $\mathcal{E}[\vec{f};\theta]$.
- Let ϕ be the constraint on admissible parameterizations.

Parameterization

- Term η_f in (5) is a large sum with multiple occurrences of parameters θ within different instances of sub-terms t_j.
- Let C be a constraint on the expected rewards for $\vec{f}$, i.e., C is a formula with free variable $\mathcal{E}_f$ formalizing the requirements on the expectation $\mathcal{E}[\vec{f};\theta]$.
- Let ϕ be the constraint on admissible parameterizations.

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

$$(\mathcal{E}_{\vec{f}} = \eta_{\vec{f}}) \wedge \phi \wedge C \tag{6}$$

using an appropriate constraint solver.

Parameterization

- Term η_f in (5) is a large sum with multiple occurrences of parameters θ within different instances of sub-terms t_j.
- Let C be a constraint on the expected rewards for $\vec{f}$, i.e., C is a formula with free variable $\mathcal{E}_f$ formalizing the requirements on the expectation $\mathcal{E}[\vec{f};\theta]$.
- Let ϕ be the constraint on admissible parameterizations.

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

$$(\mathcal{E}_{\vec{f}} = \eta_{\vec{f}}) \wedge \phi \wedge C \tag{6}$$

using an appropriate constraint solver.

Remark: Existence of a parameter instance θ 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., $(\mathcal{E}_{\vec{f}} = \eta_{\vec{f}}) \wedge \phi \wedge C$, is an arithmetic constraint containing
  1. addition, multiplication, exponentiation by integer constants,
  2. the operations found in the terms $t_1, \ldots, t_l$ defining the parameter dependency $p(\theta)$ 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 is an algorithms integrating interval constraint propagation and SAT modulo theory for solving constraint systems over $\mathbb{R}, +, *, \sin, \exp, \ldots$ Implementations called HySAT II, iSAT, and iSAT-3 have been made available by the AVACS consortium from 2008 onward; more recently Sicun Gao et al. at CMU have provided an independent implementation under the name dReal.

[2] You ought to override 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 aver
  X236 = -28493.9 * alpha**6 * (1-alpha
-- X235 represents 12 sample(s) of aver
  X235 = -21845.3 * alpha**6 * (1-alpha
-- X234 represents 35 sample(s) of aver
  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 + ... +

-- 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 non-linear or even non-polynomial arithmetic.

Expectations and parameters may be

- multi-dimensional,
- subject to arbitrary Boolean combinations of constraints,
- subject to non-polynomial arithmetic constraints.

How iSAT works (here: iSAT 2)

$$
\begin{aligned}
c_1: &\quad (\neg a \vee \neg c \vee d) \\
c_2: &\wedge (\neg a \vee \neg b \vee c) \\
c_3: &\wedge (\neg c \vee \neg d) \\
c_4: &\wedge (b \vee x \geq -2) \\
c_5: &\wedge (x \geq 4 \vee y \leq 0 \vee h_3 \geq 6.2) \\
\\
c_6: &\wedge h_1 = x^2 \\
c_7: &\wedge h_2 = -2 \cdot y \\
c_8: &\wedge h_3 = h_1 + h_2
\end{aligned}
$$

- 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:

$$
\begin{aligned}
b &\equiv b \geq 1 \\
\neg b &\equiv b \leq 0
\end{aligned}
$$

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

How iSAT works (here: iSAT 2)

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

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

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

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

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

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

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

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

DL 1: $a \geq 1$

How iSAT works (here: iSAT 2)

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

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

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

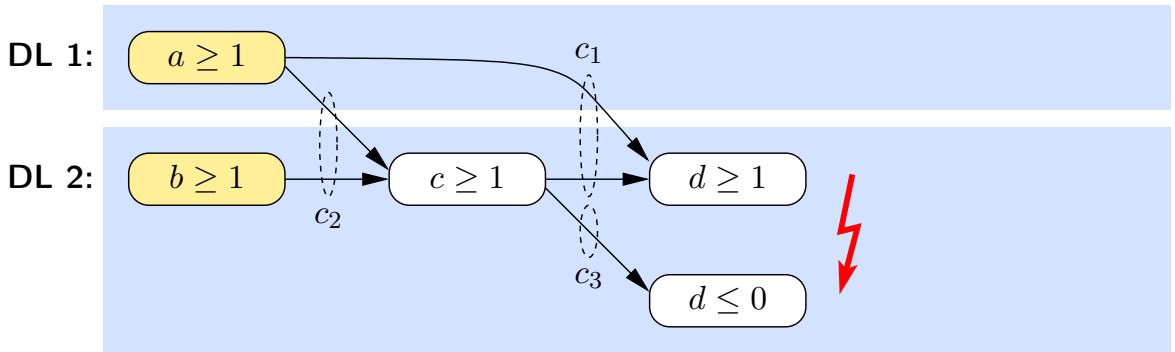
c_4 : $\wedge\ (b \vee x \geq -2)$

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

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

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

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



How iSAT works (here: iSAT 2)

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

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

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

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

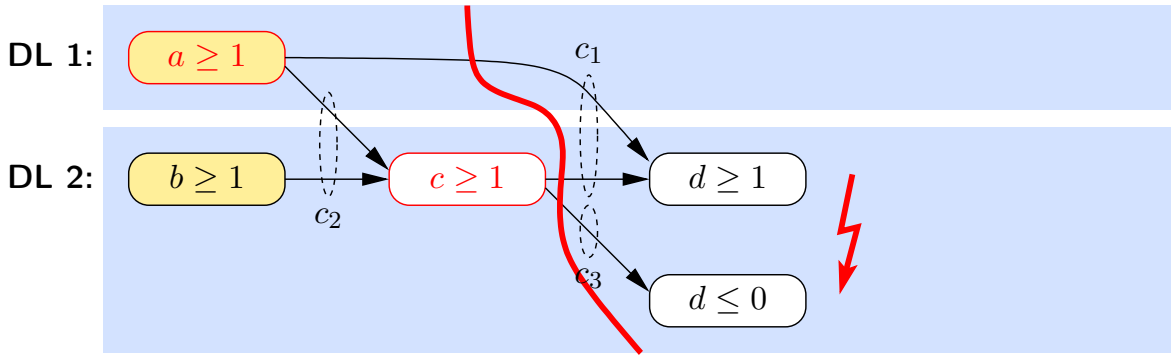
c_5 : $\wedge\ (x \geq 4 \vee y \leq 0 \vee h_3 \geq 6.2)$

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

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

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

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



How iSAT works (here: iSAT 2)

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

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

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

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

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

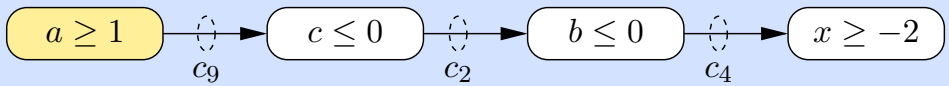
c_6 : $\wedge\ h_1 = x^2$

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

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

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

DL 1:



How iSAT works (here: iSAT 2)

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

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

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

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

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

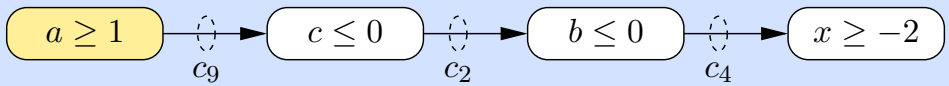
c_6 : $\wedge\ h_1 = x^2$

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

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

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

DL 1:



DL 2:

How iSAT works (here: iSAT 2)

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

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

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

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

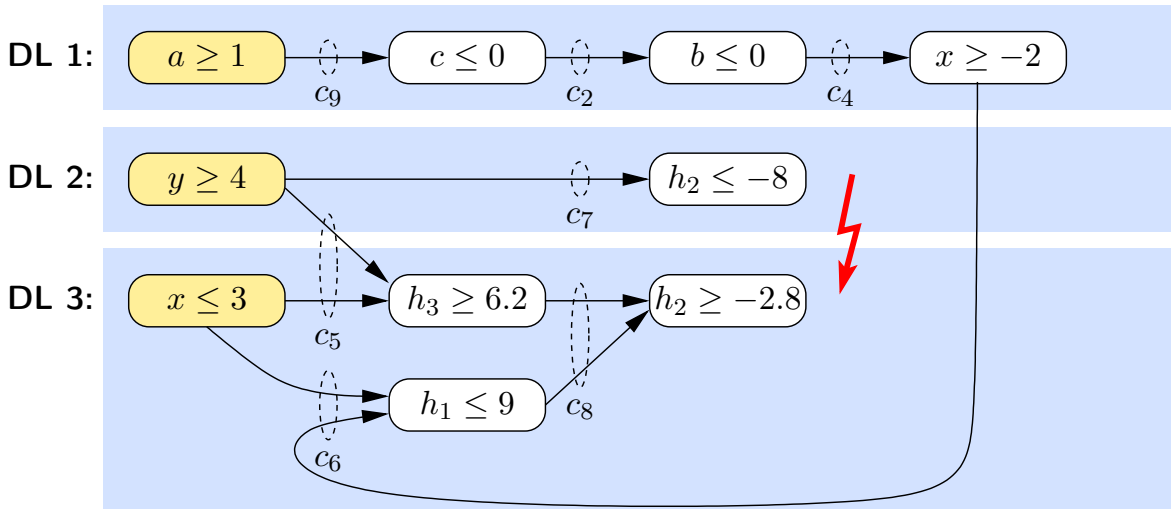
c_5 : $\wedge\ (x \geq 4 \vee y \leq 0 \vee h_3 \geq 6.2)$

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

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

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

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



How iSAT works (here: iSAT 2)

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

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

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

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

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

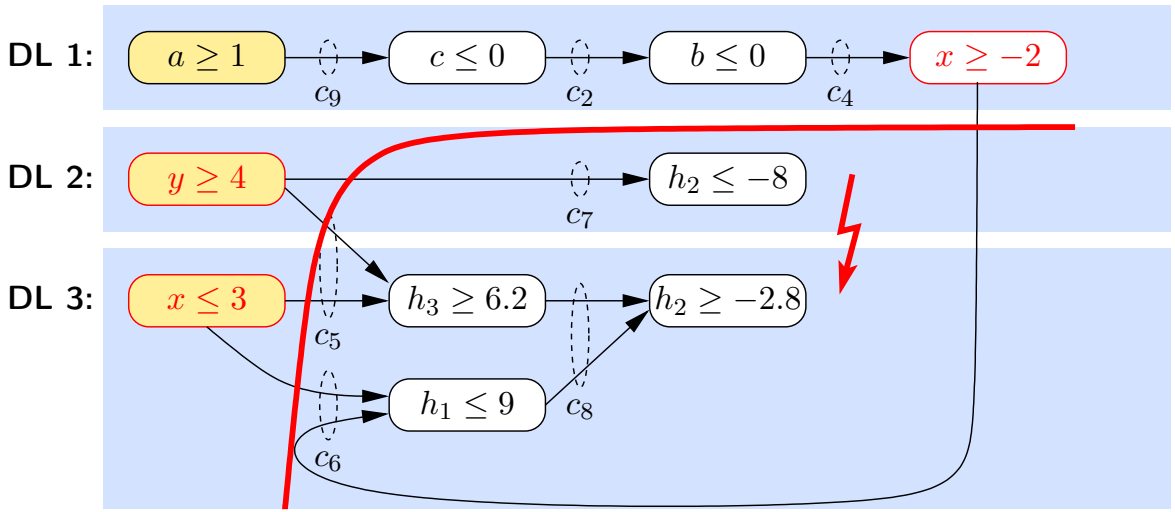
c_6 : $\wedge\ h_1 = x^2$

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

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

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

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



← conflict clause = symbolic description

of a rectangular region of the search space

which is excluded from future search

How iSAT works (here: iSAT 2)

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

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

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

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

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

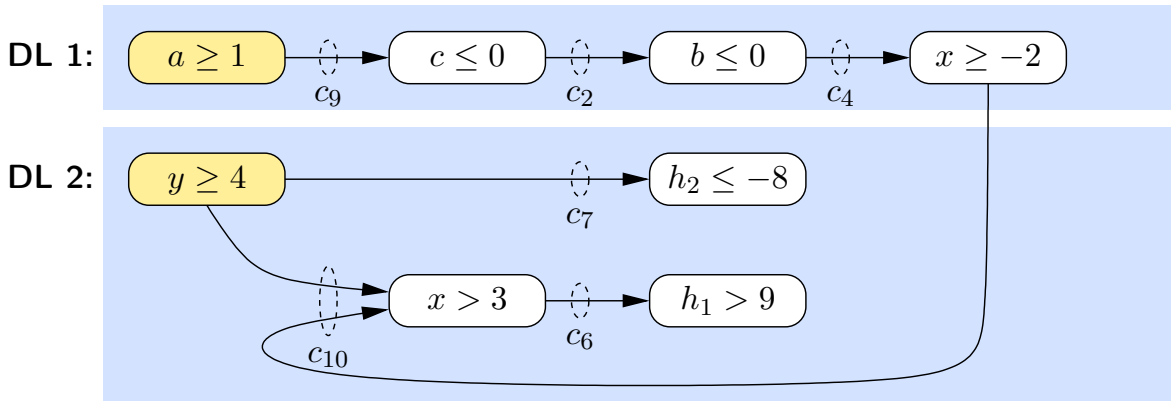
c_6 : $\wedge\ h_1 = x^2$

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

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

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

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



How iSAT works (here: iSAT 2)

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

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

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

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

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

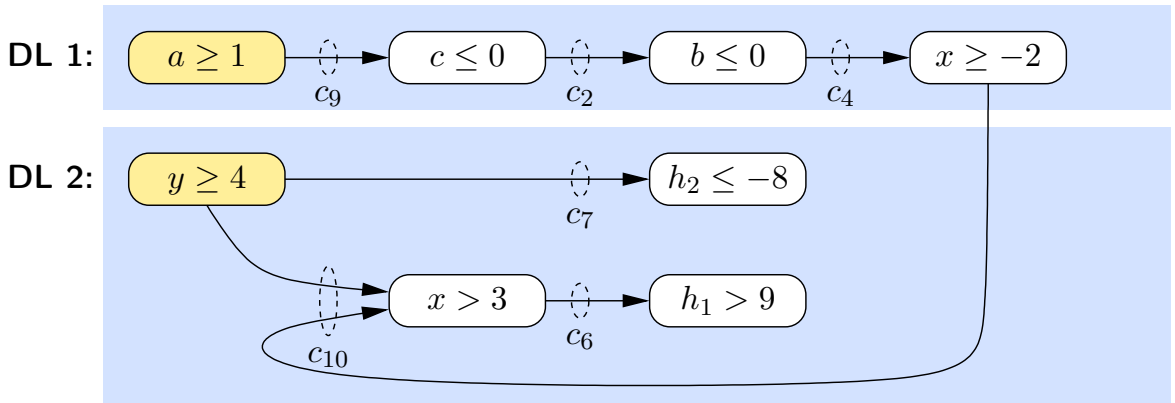
c_6 : $\wedge\ h_1 = x^2$

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

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

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

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



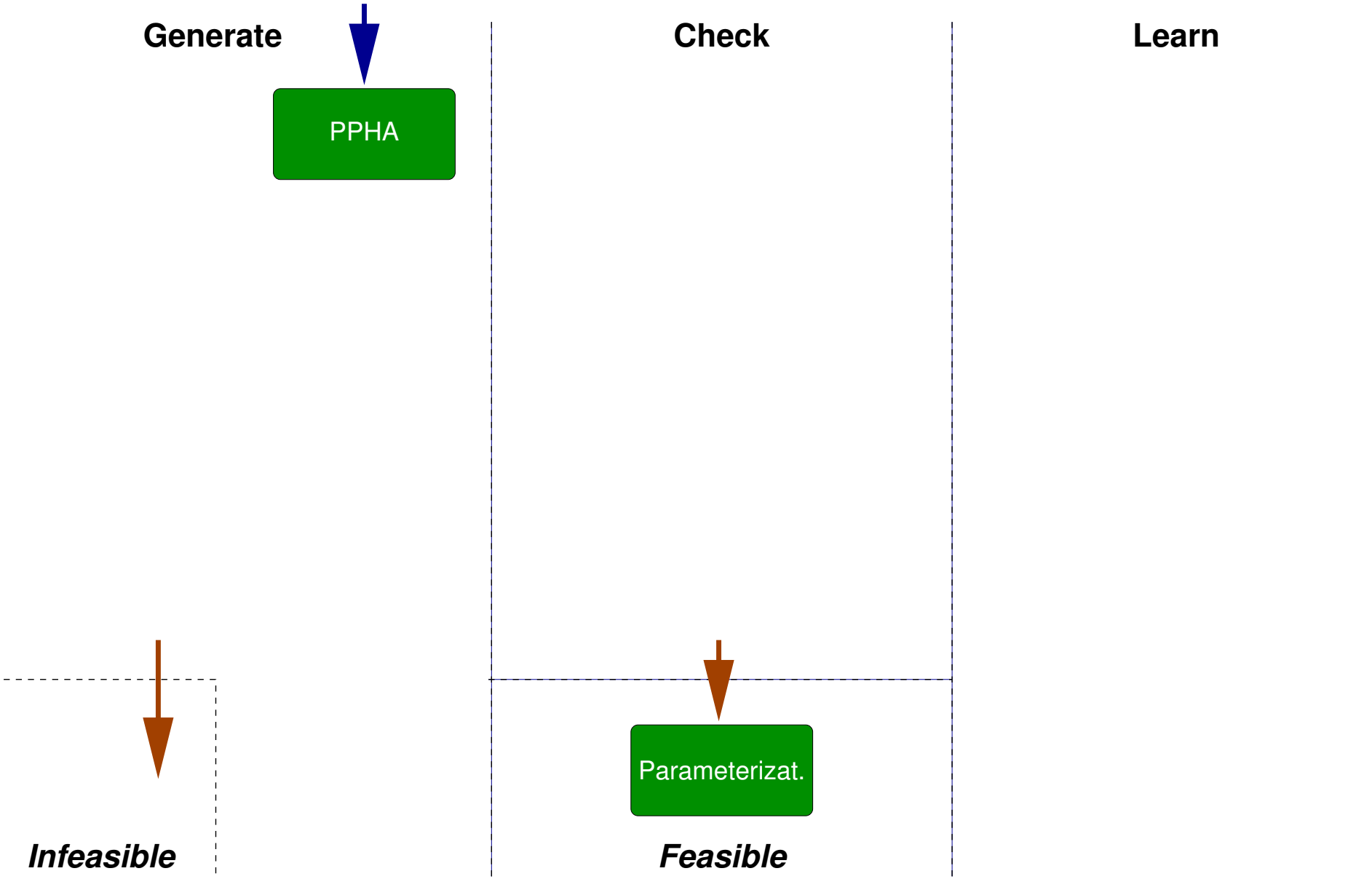
- Continue do split and deduce until either
  - ▷ formula turns out to be UNSAT (unresolvable conflict)
  - ▷ solver is left with 'sufficiently small' portion of the search space for which it cannot derive any contradiction

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

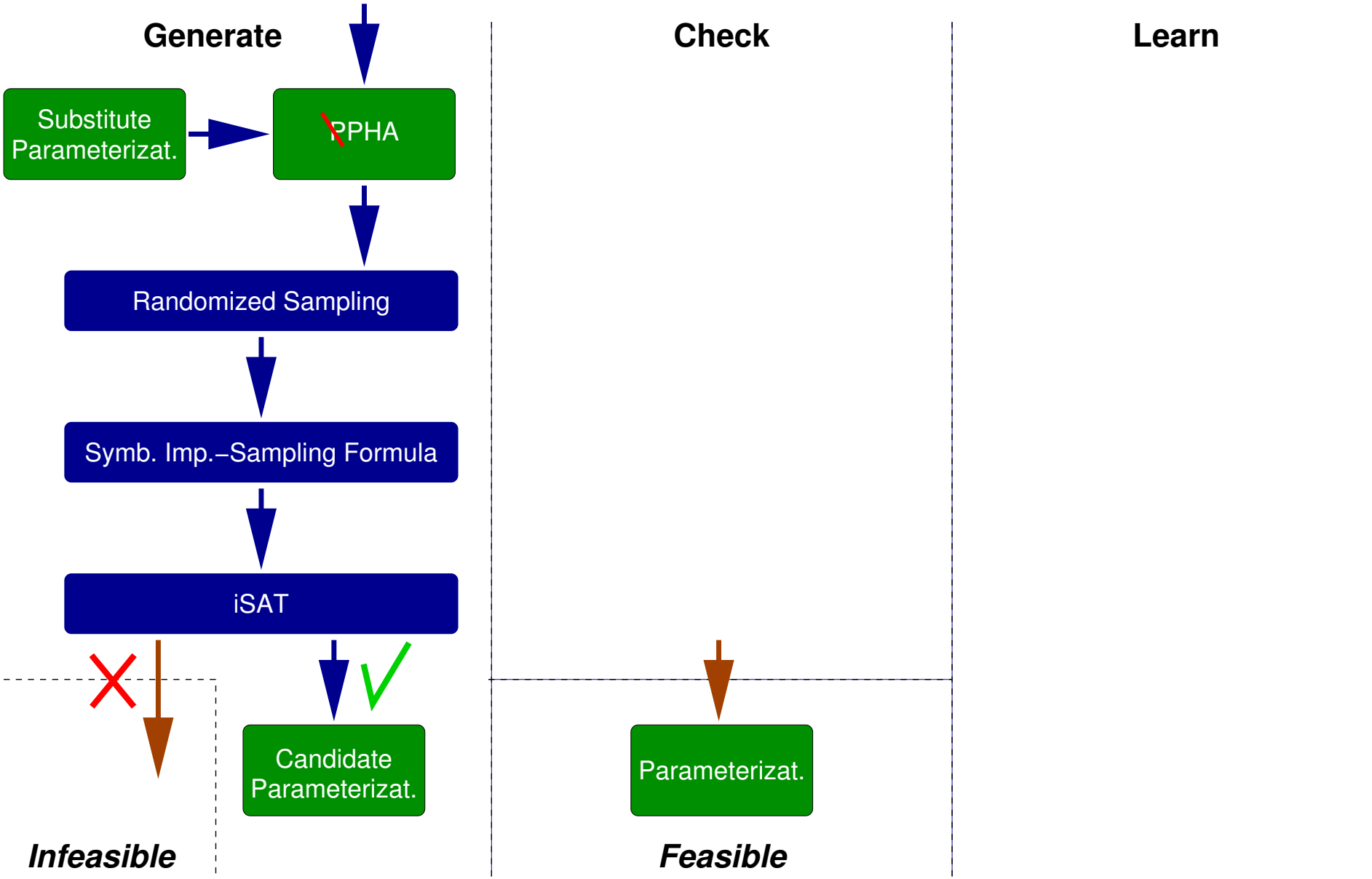
Iterative Refinement of the Encoding

Dealing with the approximation error incurred by importance sampling

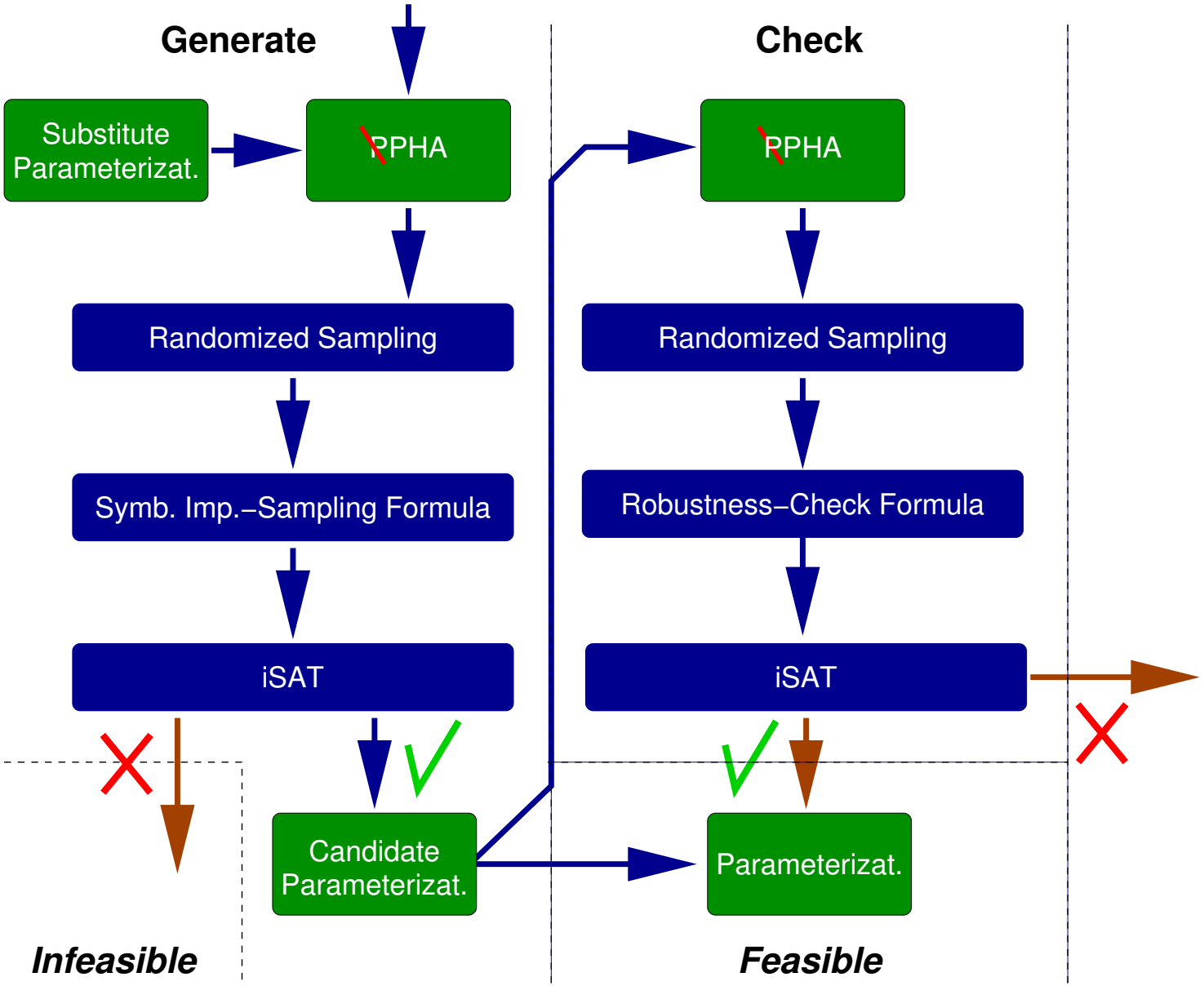
Learning from Counterexamples



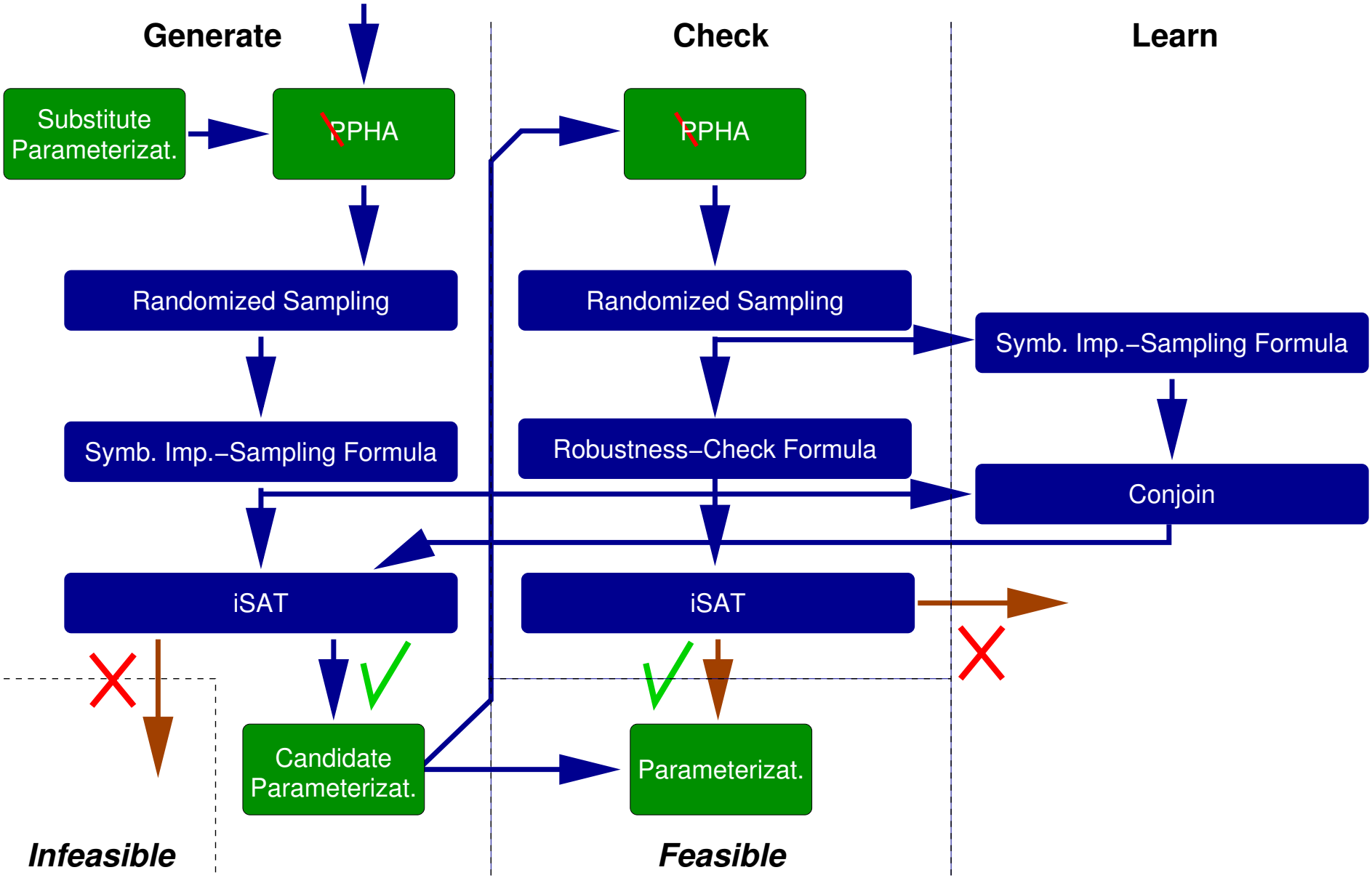
Learning from Counterexamples



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).

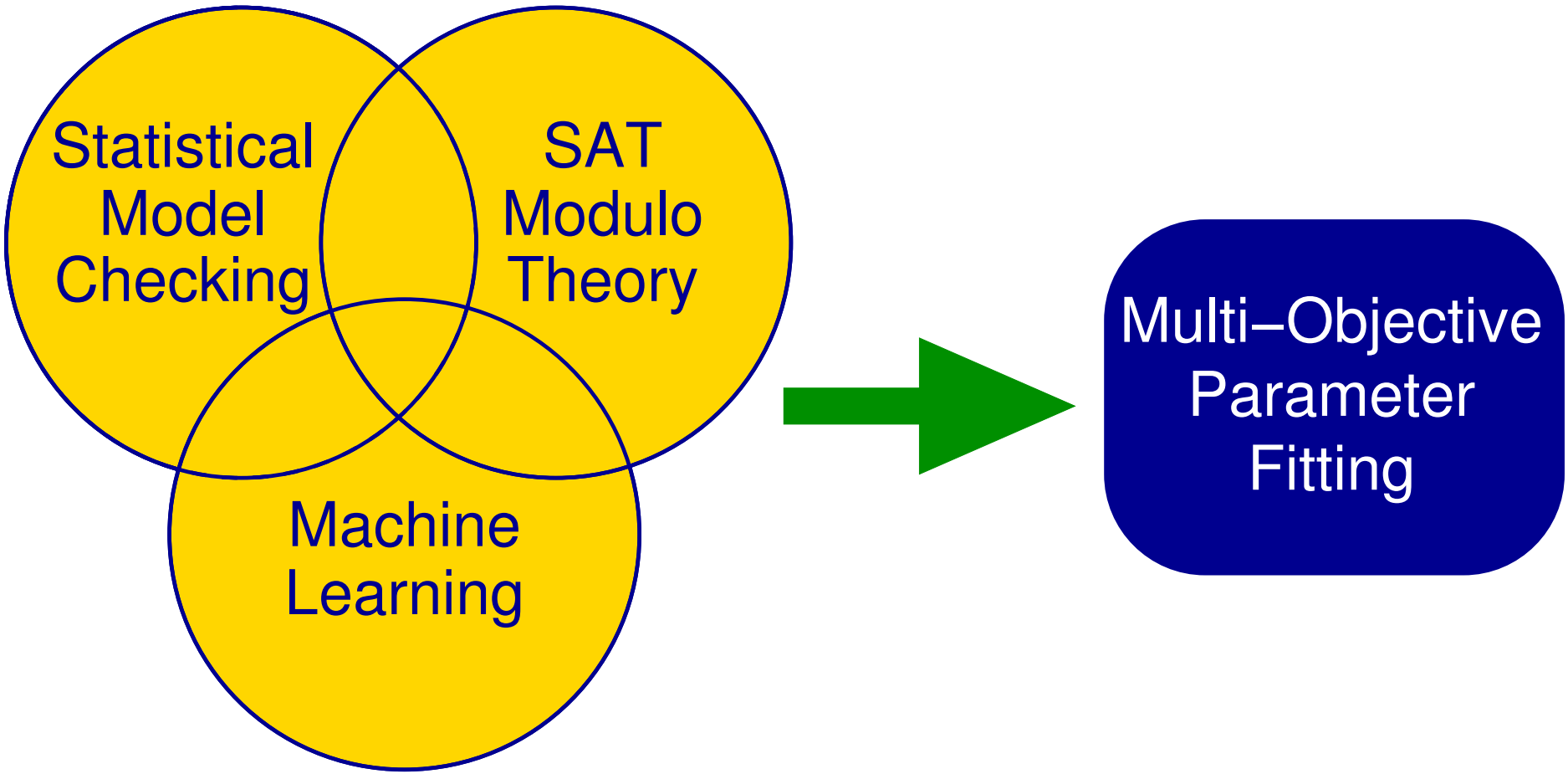
Correctness

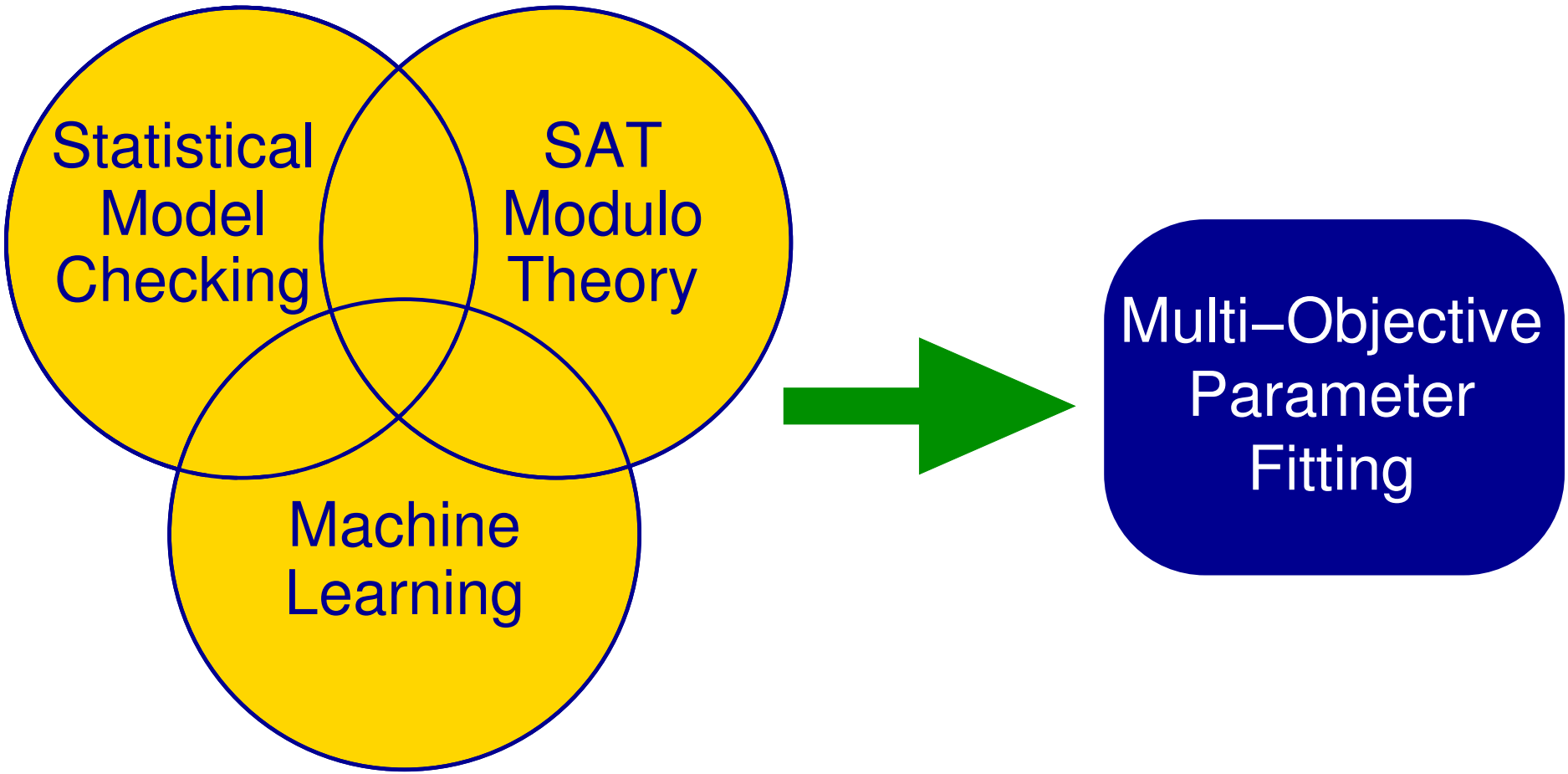
1. If the algorithm terminates with "Feasible" then the parameter instance provided yields expectations satisfying C with confidence $\geq P$.
2. If the algorithm terminates with "Infeasible" then for any parameter instance satisfying ϕ, the associated expectations violate C with confidence $\geq P$.

Discussion

In short: (the approach, once again)

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 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 rewards;
5. Simplification of that expression by means of merging terms;
6. Use of SMT solving for large Boolean combinations (multiple 100 conjuncts) of, a.o., higher-order polynomials (examples go to degree 400 and beyond) for determining suitable parameters;
   - iSAT tackled many instances of degrees below 100 in seconds — actually much faster than the simulation phase
7. Sampling, robustness check, and learning in case of failure.


Statistical
Model
Checking
SAT
Modulo
Theory
Machine
Learning
Multi–Objective
Parameter
Fitting



Many more such combinations wait to be explored!