Evaluating Monte-Carlo Tree Search for Property Falsification on Hybrid Systems

Rémi Delmas, Thomas Loquen (ONERA)
Josep Boada-Bauxell, Mathieu Carton (Airbus Operations SAS)

ONERA – Airbus Operations SAS

20/06/2019
Outline

Context

Industrial Benchmark

Property Falsification as Optimization

Monte-Carlo Tree Search (MCTS)

Benchmark Application Results

Related Works

Conclusion et Perspectives
Outline

Context

Industrial Benchmark

Property Falsification as Optimization

Monte-Carlo Tree Search (MCTS)

Benchmark Application Results

Related Works

Conclusion et Perspectives
The IKKY-SEFA Project (2016–2019)

Intégration cocKpit & sYstèmes – Embedded Systems & Advanced Functions

- Funding: Délégation Générale de l’Aviation Civile (DGAC)
- Partners: ONERA, Airbus, Dassault, LAAS-CNRS
- Goals:
  - Improve the design processes for industrial embedded systems
  - Evaluate the SoA of hybrid systems verification...
    - Model-checking, SAT-modulo-semidefinite-programming, robustness analysis, reinforcement learning, ...
  - ...on industrial benchmarks:
    - An aircraft pitch control law (Airbus),
    - Reference model + altered models.

This presentation

Reinforcement learning techniques applied to property falsification.
Hybrid Systems Verification Challenges

In our particular case:

- Time-Discrete/Continuous hybrid closed-loop model,
- Modal control law: manual mode, autopilot mode, flight envelope protection,
- Non-linearity: polynomials, trig. functions, LUTs, vote, saturations, . . .
- matlab/Simulink: no formal semantics, numerical issues (ODEs), . . .
Outline

Context

Industrial Benchmark

Property Falsification as Optimization

Monte-Carlo Tree Search (MCTS)

Benchmark Application Results

Related Works

Conclusion et Perspectives
Evaluating Monte-Carlo Tree Search for Property Falsification on Hybrid Systems

Industrial Benchmark

Benchmark Overview

Composants

- Continuous time (ODE),
- Aircraft model: flight dynamics + wind,
- Actuator model: order allocation, dynamics, saturations,
- Sensor model: dynamics, filtering, delay.
Control Law

Components

- Discrete time, multi-rate,
- AC State estimation,
- Feedback control on $n_z$ (LPV),
- Manual and autopilot modes,
- Dynamic flight envelope protection.
Control Law

Controllable inputs

- **bapeng**: Boolean d’*autopilot engagement*,
- **selalt**: real d’*selected altitude* for autopilot,
- **nzcmanche**: real d’*pilot stick order* for direct mode,
- **wx, wz**: real wind speed on horiz. and vert. axes.
Control Law

Some Figures

- 112 continuous states, 27 switches, 4 latches, 28 2D-LUTs, 34 saturations.
- discrete multi-rate: $T_1$ et $T_2 = 1.5 T_1$
Autopilot Modes

- Transfer
- Reversion
- Capture
- Maintain

selalt

alt
Outline

Context

Industrial Benchmark

Property Falsification as Optimization

Monte-Carlo Tree Search (MCTS)

Benchmark Application Results

Related Works

Conclusion et Perspectives
Temporal Logics with Robust Semantics

The Signal Temporal Logic (STL) [4] with language, \((a, b) \in \mathbb{R}^2:\)

\[
\phi \quad ::= \quad \text{true} \mid x_i \geq 0 \mid \neg \phi \mid \phi \land \phi \mid \phi \mathcal{U}_{[a,b]} \phi
\]

\[
\mathcal{F}_{[a,b]} \phi \quad ::= \quad \text{true} \mathcal{U}_{[a,b]} \phi
\]

\[
\mathcal{G}_{[a,b]} \phi \quad ::= \quad \neg \mathcal{F}_{[a,b]} \neg \phi
\]

defines, in addition to the standard Boolean interpretation \(\models\), a \emph{robust} interpretation \(\mathcal{R}\) over timed traces \(w\) such that:

\[
\mathcal{R}(\phi, w, t) \geq 0 \text{ iff } (w, t) \models \phi
\]

Where:

\[
\begin{align*}
\mathcal{R}(\text{true}, w, t) &= +\infty \\
\mathcal{R}(x_i \geq 0, w, t) &= x_i^w(t) \\
\mathcal{R}(\neg \phi, w, t) &= -\mathcal{R}(\phi, w, t) \\
\mathcal{R}(\phi_1 \land \phi_2) &= \min(\mathcal{R}(\phi_1, w, t), \mathcal{R}(\phi_2, w, t)) \\
\mathcal{R}(\phi_1 \mathcal{U}_{[a,b]} \phi_2) &= \max_{t' \in t+[a,b]}(\min(\mathcal{R}(\phi_2, w, t'), \min_{t'' \in [t, t']}(\mathcal{R}(\phi_1, w, t''))))
\end{align*}
\]
From Verification to Optimization

Given:

- \( H = \langle S_H, A_H, T_H \rangle \) a hybrid model with:
  - \( S_H \): state space,
  - \( A_H \): controllable input space,
  - \( T_H \): hybrid transition relation,
- \( T \in \mathbb{R} \) a finite *horizon*,
- \( d \in \mathbb{R} \) a constant *action duration* \( d < T \),
- \( \Phi = G_{[0,T]} \phi \): a safety property on \( H \) with \( \phi \) modality-free,
- \( \text{sim}(T_H, s, a, d) \) the trajectory of \( H \) from state \( s \) duration \( d \) with constant control input \( a \).
We define a finite-action MDP $M = \langle S, A, T, R, \alpha \rangle$ where:

- $S \subseteq S_H$,
- $A \subseteq A_H$ is finite and user-specified,
- $(s, a, s') \in T$ iff $s'$ is the final state of $\text{sim}(T_H, s, a, d)$
- $R(s, a, s') = -R(\phi, \text{sim}(T_H, s, a, d), d)$

The falsification problem of $\Phi$ on $H$ from state $s_0$ is under-approximated as an optimal planning problem on $M$ from $s_0$ over finite horizon $T$, where the goal is to generate a finite action sequence producing a trace $w$ such that $\mathbb{E}[R(\Phi, w, t)]$ is the minimal for all $t$. 
Outline

Context

Industrial Benchmark

Property Falsification as Optimization

Monte-Carlo Tree Search (MCTS)

Benchmark Application Results

Related Works

Conclusion et Perspectives
Monte-Carlo Tree Search (MCTS)

*Monte-Carlo Tree Search (MCTS)*[3], [6] is a generic algorithm for finite-horizon planning of discrete-action MDPs which builds a search tree over $A$ from the initial state, and estimates $\mathbb{E}[R_{t+i}|a_t = a_0, \ldots, a_{t+i} = a_i]$ in each node of depth $i$ using:

- **Rollout** for fringe nodes: the cumulative reward is sampled to the horizon using a stochastic policy,
- **Backup** for internal nodes: backpropagates estimates from subtrees up to the root,
- **Multi-Armed Bandit** policies to select which branch to grow during search.
Multi-Armed Bandits

A $K$-Bandit problem is defined by:

- a set of sequences of random variables $R_{i,n}$ pour $i \in [1, K]$, $n \in \mathbb{N}^+$, i.i.d with unknow mean $\mu_i$ and finite variance $\sigma_i^2$,
- $R_{i,\star}$ and $R_{j,\star}$ are independant for all $i, j$.

A each game step $n$, the learner choses an arm $i$ and gets a reward $r_{i,n} \sim R_{i,n}$. 
Multi-Armed Bandits: Exemple

Exemple:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$r_{i,1}$</th>
<th>$r_{i,2}$</th>
<th>$r_{i,3}$</th>
<th>$r_{i,4}$</th>
<th>$r_{i,5}$</th>
<th>$r_{i,6}$</th>
<th>$r_{i,7}$</th>
<th>$r_{i,8}$</th>
<th>$\bar{R}_{i,[1-8]}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15.84</td>
<td>11.92</td>
<td>15.09</td>
<td>14.40</td>
<td>14.78</td>
<td>12.66</td>
<td>13.08</td>
<td>17.87</td>
<td>14.46</td>
</tr>
<tr>
<td>3</td>
<td>11.60</td>
<td>10.79</td>
<td>9.11</td>
<td>10.94</td>
<td>11.29</td>
<td>11.89</td>
<td>11.58</td>
<td>9.67</td>
<td>10.86</td>
</tr>
</tbody>
</table>

Questions:

- Is it possible to design a policy maximizing the cumulative reward expectation?
Multi-Armed Bandits: Exemple

Exemple:

<table>
<thead>
<tr>
<th>i</th>
<th>$r_{i,1}$</th>
<th>$r_{i,2}$</th>
<th>$r_{i,4}$</th>
<th>$r_{i,5}$</th>
<th>$r_{i,6}$</th>
<th>$r_{i,7}$</th>
<th>$r_{i,8}$</th>
<th>...</th>
<th>$\bar{R}_{i,[1-10000]}$</th>
<th>Dis.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15.84</td>
<td>11.92</td>
<td>15.09</td>
<td>14.40</td>
<td>14.78</td>
<td>12.66</td>
<td>13.08</td>
<td>17.87</td>
<td>...</td>
<td>13.98</td>
</tr>
<tr>
<td>2</td>
<td>12.27</td>
<td>14.12</td>
<td>13.93</td>
<td>13.25</td>
<td>14.28</td>
<td>13.79</td>
<td>12.69</td>
<td>15.79</td>
<td>...</td>
<td>14.01</td>
</tr>
<tr>
<td>3</td>
<td>11.60</td>
<td>10.79</td>
<td>9.11</td>
<td>10.94</td>
<td>11.29</td>
<td>11.89</td>
<td>11.58</td>
<td>9.67</td>
<td>...</td>
<td>10.50</td>
</tr>
</tbody>
</table>

Questions:

- Is it possible to design a policy maximizing the cumulative reward expectation?
Exploration vs. Exploitation

The problem:

- One must use an arm to estimate its mean & variance,
- **Exploring non-optimal** arms instead of **exploiting** the **optimal** arm generates a **regret**,
- The goal is to build a **regret-minimizing policy** $\pi$ using only past information:
  - $\bar{R}_i(n)$: empirical mean reward of arm $i$ at step $n$,
  - $\mathbb{V}(\bar{R}_i)(n)$: empirical variance of the empirical mean reward for machine $i$ at step $n$. 
Cumulative Regret

\[ regret(n) = n\mu^* - \mu_j \sum_{1 \leq j \leq K} \mathbb{E}(t_j(n)) \]

With:

- \( \mu^* = \max_i(\mu_i) \): optimal average reward,
- \( t_j(n) \): number of times arm \( j \) was played over the \( n \) first game steps.
The Upper Confidence Bound Policy (UCB1)[2]
At each game step $n$ select arm $i$ maximizing an over-approximation of the mean reward:

$$UCB1_i(n) = \bar{R}_i(n) + c \times \sqrt{\frac{\ln(n)}{t_i(n)}}$$

Where:

- $c > 0$: exploration/exploitation tradeoff parameter,

This policy is such that:

- $\text{regret}(n) \approx O(\ln(n))$ when $n \to \infty$,
- the probability of using a sub optimal arm goes 0 when $n \to \infty$. 
UCB1: Intuition

- The *exploration term* term $c \times \sqrt{\frac{\ln(n)}{t_i(n)}}$:
  - decreases when $i$ is played,
  - increases if an arm $j \neq i$ is played,
  - approaches 0 for all arms when $n \to \infty$,
- initially, fair exploration of arms $i, j$ if $\bar{R}_i \simeq \bar{R}_j$,
- long term, exploitation of the arm with best mean reward.
Monte-Carlo Tree Search (MCTS)

The Upper Confidence bound applied to Trees (UCT) algorithm:

- Each node stores $UCB1$ statistics $(\bar{R}_t, n)$,
- Only requires a black box simulator for $R_t$. 

Figure: Algo. MCTS – extrait de [5]
Monte-Carlo Tree Search (MCTS)

(a) Selection: From the root, traverse down following best UCB1 nodes, stop on first incomplete leaf node.

Figure: Algo. MCTS – extrait de [5]
Monte-Carlo Tree Search (MCTS)

(a) Selection  (b) Expansion  (c) Simulation  (d) Backpropagation

Figure: Algo. MCTS – extrait de [5]

(b) Expansion Randomly select a not-yet-explored action and add child node.
Monte-Carlo Tree Search (MCTS)

(a) Selection  (b) Expansion  (c) Simulation  (d) Backpropagation

**Figure:** Algo. MCTS – extrait de [5]

(c) **Simulation** Simulate $R_t$ to the finite horizon using a uniform random policy over $A$. 
Monte-Carlo Tree Search (MCTS)

(a) Selection  (b) Expansion  (c) Simulation  (d) Backpropagation

**Figure**: Algo. MCTS – extrait de [5]

**d) Backpropagation** Update UCB1 statistics \( \bar{R}_t, n \) for each node of the current branch, then goto selection \( \text{(a) Selection} \).
Outline

Context

Industrial Benchmark

Property Falsification as Optimization

Monte-Carlo Tree Search (MCTS)

Benchmark Application Results

Related Works

Conclusion et Perspectives
Experimentation Approach

Implementation

- A matlab implementation of UCT,
- Simulink for cumulative reward sampling,
- Properties:
  - threshold overshoots, frequential behaviour, event-based properties,
  - modeled as synchronous Simulink observers.
- UCT is run on the reference benchmark and altered benchmarks,
- A human analysis of maximum reward traces is conducted: do they activate the expected defects?
Threshold Overshoot:  Spec + Reward

**Spec:** flight parameter $X$ exceeds its target value by some given *margin*, expressed as $X \geq X_{\text{target}} + \text{margin}$.

**Reward Function:**

![Diagram of reward function](image)

**Action Space:** $A = \{bapeng \in \{T, F\}, \text{selalt} = 10000\text{fts}, \text{nzcmanche} \in \{\text{neutral, half\_up, full\_up}\}, \text{wx} = \text{wz} = 0.0\text{ and an action duration 5s.}\}

**MCTS Parameters:** $\alpha = 0.9999$ and $p = 5$, Plan size to 30 with 30 MCTS iterations per plan step.
Threshold overshoot: Spec + Reward

Tree Search
Threshold overshoot: Spec + Reward

Best Reward Trace

- $\alpha$
- $\alpha_{\text{target}}$
- $\alpha_{\text{lim}}$
Frequential Prop. 1

**Spec**: the average amplitude of $n_z$ oscillations in a specific frequency band corresponding should be minimal.

**Reward Function**:

Low frequencies, corresponding to the expected response to low frequency pilot orders, and high frequencies corresponding to noise, are cut using an 11\textsuperscript{th} order Butterworth filter. The edge frequencies are defined according to flight control engineers knowledge. The absolute value of the filtered signal is then fed into an exponentially decaying moving average operator to obtain the final reward signal.
Frequential Prop. 1

Action Space:

- $bapeng \in \{ T, F \}$,
- $selalt \in \{25000\, fts, 28000\, fts\}$,
- $nzcmanche \in \{\text{half\_down}, \text{neutral}, \text{half\_up}\}$,
- $wx = wz = 0.0$
- action duration 5s.

MCTS Parameters: $\alpha = 0.9999$ and $p = 10$, plan size $= 25$, iterations $= 2880$. 
Frequential Prop. 1

Tree Search
Frequentional Prop. 1

Best Reward Trace
Frequential Prop. 2

**Spec:** \( n_z \) oscillations around the commanded \( n_{zc} \) should be minimal. Such oscillations are tolerated when they are low frequency, but can become problematic when they are high frequency and sustained over time, regardless of amplitude.

**Reward Function:**

Counts the number of sign inversions of \( n_z - n_{zc} \) in a sliding window of a few seconds,
Frequential Prop. 2

**Action Space**:  
- \( bapeng \in \{ T, F \} \),  
- \( selalt \in \{ 25000\text{fts}, 28000\text{fts} \} \),  
- \( nzcmanche \in \{ \text{half\_down}, \text{neutral}, \text{half\_up} \} \),  
- \( wx = wz = 0.0 \)  
- action duration 5s.

**MCTS Parameters**: \( \alpha = 0.9999 \) and \( p = 5 \), plan size to 25 with 120 MCTS iterations per plan step.
Frequential Prop. 2

Tree Search

[Diagrams showing tree search with various branches and node expansions]
Frequential Prop. 2

Best Reward Trace
Spurious AP Disconnection

**Spec:** no spurious auto-pilot disconnection in presence of wind perturbations.

**Reward Function:**

More precisely, we are searching for wind scenarios which cause the internal auto-pilot engagement signal `bapeng_int` to become false on a stabilized altitude in the absence of pilot intervention. We use the disconnection time of the auto-pilot as reward function.
Spurious AP Disconnection

MCTS Parameters

- **Weak Wind:**
  - $bapeng = T,$
  - $selalt = 25000fts,$
  - $nzcmanche = neutral,$
  - $(wx, wz) \in \{zero, low\}^2,$
  - action duration 5s,
  - $\alpha = 0.9999, p = 5,$
  - plan size 15, 20 MCTS iterations per plan step.

- **Strong Wind:**
  - $bapeng = T,$
  - $selalt = 25000fts,$
  - $nzcmanche = neutral,$
  - $(wx, wz) \in \{zero, low, medium, high, very\_high\}^2,$
  - action duration 5s,
  - $\alpha = 0.9999, p = 5,$ plan size 15, 25 MCTS iterations per plan step.
Evaluating Monte-Carlo Tree Search for Property Falsification on Hybrid Systems

Benchmark Application Results

Spurious AP Disconnection

Tree Search Weak Wind

[Graphs showing spurious AP disconnection with tree search and weak wind]
Spurious AP Disconnection

Tree Search Strong Wind
Outline

Context

Industrial Benchmark

Property Falsification as Optimization

Monte-Carlo Tree Search (MCTS)

Benchmark Application Results

Related Works

Conclusion et Perspectives
Related Works

State of the Art 2017–2018

Our work is directly inspired from the following papers:


State of the Art 2017–2018


Outline

Context

Industrial Benchmark

Property Falsification as Optimization

Monte-Carlo Tree Search (MCTS)

Benchmark Application Results

Related Works

Conclusion et Perspectives
Conclusion

Results

- Surprisingly good results,
- All benchmarks were successfully analyzed,
- Generated traces allowed to pinpoint unsafe behaviours.

Limitations

- Reward engineering,
- Discrete action space and duration selection.
Perspectives

- Algorithmic evolutions:
  - Stochastic tree search policy instead of UCB1,
  - Introduce Kernel-Regression estimators to:
    - handle continuous action spaces,
    - use as a similarity measure of reached states to share subtrees,
    - reduce the rollout budget by predicting rollout values using KR.

- Study compilation of hybrid dataflow models to HW accelerators (GPU, FPGA) to speed-up rollout.
Takumi Akazaki, Shuang Liu, Yoriyuki Yamagata, Yihai Duan, and Jianye Hao.
Falsification of cyber-physical systems using deep reinforcement learning.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer.

Rémi Coulom.
Efficient selectivity and backup operators in monte-carlo tree search.
Alexandre Donzé and Oded Maler.
Robust satisfaction of temporal logic over real-valued signals.

Steven James, George Konidaris, and Benjamin Rosman.
An analysis of monte carlo tree search.
In Satinder P. Singh and Shaul Markovitch, editors, Proceedings of the Thirty-First AAAI Conference on Artificial
Levente Kocsis and Csaba Szepesvári.
Bandit-based monte-carlo planning.

Zhenya Zhang, Gidon Ernst, Ichiro Hasuo, and Sean Sedwards.

Time-staging enhancement of hybrid system falsification.  

Zhenya Zhang, Gidon Ernst, Sean Sedwards, Paolo Arcaini, and Ichiro Hasuo.  
Two-layered falsification of hybrid systems guided by monte carlo tree search.  