Control Improvisation

Sanjit A. Seshia
Professor
EECS, UC Berkeley

Joint work with:
Daniel Fremont, Tommaso Dreossi, Rafael Valle, Sophie Libkind, Alexandre Donze, David Wessel

December 11, 2017
Diversity and Resilience in Control

• Diversity: How to *engineer* diversity of behaviors?
  – For system robustness, resilience, privacy, etc.

• Resilience: How to *adapt* to changing environment conditions?
  – Learn about environment via controlled random exploration

➢ Underlying computational problem: Control Improvisation
What does it mean to “improvise”?  

“To perform without preparation”  
– Merriam-Webster Dictionary

“Learn the rules and then break them in such a way as to exercise good taste.”  
– Sir George Shearing (Jazz musician)

“Only he who is well prepared has any opportunity to improvise.”  
– Ingmar Bergman (movie director, writer and producer)
Music Improvisation

• Generate a variant of existing tune
  – “improvise on this jazz melody”

• Follow conventions of the genre

• Don’t do the same thing too often

• Usually, be similar to the original

(examples at http://www.eecs.berkeley.edu/~donze/impro_page.html)
Robotic Patrolling

“Patrol an area in a way unpredictable to an adversary”

• Visit each location sufficiently often

• Don’t always take the same route

• Usually, take a route close to the best (e.g., shortest) one
Commonalities between these Applications

Generate (event/action) sequences subject to three kinds of constraints:

• Hard constraint
  – *Every* sequence satisfies some property P1

• Randomness requirement
  – No sequence is generated too frequently

• Soft constraint
  – “Most” sequences satisfy some other property P2

*Control Improvisation* is a precisely-defined theoretical problem capturing these requirements

[Fremont et al., FSTTCS 2015; Donze et al., ICMC’14; Valle et al., IoTDI’16]
Deterministic Planning from High-Level Task Specification

Generate sequences subject to:

• **Hard constraint**  
  – *Every* sequence satisfies some property P1

• **Randomness requirement**  
  – No sequence is generated too frequently

• **Soft constraint**  
  – *Most* sequences satisfy some other property P2

• **Encodeable as a Satisfiability Solving problem** (without adversarial environment)  
  – E.g. [Saha et al., IROS’14; Shoukry et al., CDC’16, HSCC’17, CDC’17]

• **Reactive Synthesis** (with adversarial environment)
Multi-Robot Motion Planning from Temporal Logic using Satisfiability Modulo Theories (SMT)

Declarative Task Specification (Temporal Logic)

SMT Solver

Component Library

Executable Software

MORE SCALABLE APPROACHES: ICCPS’16, CDC’16, HSCC’17, CDC’17

Video of Demonstration on Quadrotors
Probabilistic Planning from High-Level Task Specification

Generate ONE sequence subject to:

- **Hard constraint**
  - Sequence satisfies some property ("reach a goal state")

- **Randomness requirement**
  - Probability of success above some threshold

- **Soft constraint**
  - *Most* sequences satisfy some other property P2

- Randomness is an input to the problem
- Proved to be undecidable in the ‘90s
Control Improvisation

Generate (MANY) sequences subject to three kinds of constraints:

- **Hard constraint**
  - *Every* sequence satisfies some property P1

- **Randomness requirement**
  - No sequence is generated too frequently

- **Soft constraint**
  - “*Most*” sequences satisfy some other property P2

[Fremont et al., FSTTCS 2015; Donze et al., ICMC’14; Valle et al., IoTDI’16]
Structure of an Improviser

A generator of strings from a finite alphabet $\Sigma$ (say \{0,1\})
Example

- Variations of the string $s = \text{‘001’}$
  - Hard constraint: length 3, and no consecutive ‘1’s (easily encoded as a DFA $\mathcal{I}$)
  - Soft constraint: Hamming distance to $s$ is at most 1
Definition of Control Improviser (initial) [FSTTCS’15]

Given \( \mathcal{I} \) and \( A \), for \( 0 \leq \varepsilon \leq 1 \) and \( 0 < \rho \leq 1 \)
a distribution \( D : \Sigma^* \rightarrow [0,1] \) with support \( S \) is an
\((\varepsilon, \rho)\)-improvising distribution if:

- \( S \subseteq I \)  
  **Hard constraint**
- \( \forall w \in S, \ D(w) \leq \rho \)  
  **Randomness requirement**
- \( \Pr[w \in A \mid w \leftarrow D] \geq 1 - \varepsilon \)  
  **Soft Constraint**

The CI instance \( C = (\mathcal{I}, A, \varepsilon, \rho) \) is feasible if such a
distribution exists. An improviser is a probabilistic
algorithm generating strings whose output
distribution is an \((\varepsilon, \rho)\)-improvising distribution.
Existence of Improvisers

- Feasibility just requires $I$ and $A$ to be large enough
- All feasible instances have improvisers

**Theorem.** For any $C = (I, \alpha, \epsilon, \rho)$, the following are equivalent:

1. $C$ is feasible.
2. $|I| \geq 1/\rho$ and $|A| \geq (1 - \epsilon)/\rho$.
3. There is an improviser for $C$. 
Example

- With $\varepsilon = \rho = \frac{1}{4}$:
  - Uniformly sample from $A$ with probability $\frac{3}{4}$
  - Uniformly sample from $I \setminus A$ with probability $\frac{1}{4}$
Complexity Results

- Complexity grows as $I$ and $A$ are given by more complex automata

<table>
<thead>
<tr>
<th>$I$</th>
<th>$A$</th>
<th>DFA</th>
<th>NFA</th>
<th>PFA</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFA</td>
<td>$L(\mathcal{I}) = \infty$</td>
<td>poly-time</td>
<td>$L(\mathcal{A}) &lt; \infty$</td>
<td>$#P$-hard</td>
</tr>
<tr>
<td>NFA</td>
<td>$L(\mathcal{I}) = \infty$</td>
<td>poly-time</td>
<td>$L(\mathcal{A}) &lt; \infty$</td>
<td>$#P$-hard</td>
</tr>
<tr>
<td>PFA</td>
<td>$L(\mathcal{I}) &lt; \infty$</td>
<td>$#P$-hard</td>
<td>-</td>
<td>undecidable</td>
</tr>
</tbody>
</table>

- **Symbolic approach**: encode constraints in (propositional) logic, leverage progress in SAT solving, model counting
Recall: Robotic Patrolling

“Patrol an area in a way unpredictable to an adversary”

• Visit each location infinitely often
• Every sub-sequence of size $s$ of the plan is repeated with low probability
• Minimize mean time between visits to a location (over all locations)
Recent Control Improvisation Demo (Berkeley + Penn)

[joint with Vijay Kumar and George Pappas]
(see https://math.berkeley.edu/~dfremont/impro.html)

**Hard constraints:**
- Visit each location at least once
- Do not visit any location twice in a row
- No more than three locations without recharging

**Soft constraints:**
Visit each location exactly once with prob. > 0.8

**Randomness:**
No trajectory generated with prob. > 0.052
Summary and Future Work

• Control improvisation is a new class of problems involving random generation of event/action sequences with hard & soft constraints

• Many Ongoing/Future Directions:
  – Reactive setting: adversarial environments
  – Soft constraints encoding general quantitative requirements
  – Different types of randomness constraints encoding specific distributional properties
  – More applications

https://math.berkeley.edu/~dfremont/impro.html