Authoring Human Simulators via Probabilistic Functional Reactive Program Synthesis

Michael Jae-Yoon Chung and Maya Cakmak

HRI2022 Late-breaking report (Paper), 2022/03/08

The Problem

- Testing social robot programs is difficult because **it requires humans**
- Researchers proposed to use **“human simulators”** (e.g., Chao & Thomas 2012)
- But **building human simulators is difficult**…

The Approach

We propose **program synthesis** approach to building human simulators!

The two key ideas are:

1. representing human simulators as **probabilistic functional reactive programming programs (PFRP)**
2. using **probabilistic inference** for synthesizing human simulator programs

Example: “Speaking” Human Simulator as PFRP

```
var makeHuman = function(state) {
  return merge(
    of(state),
    of(state).pipe(
      // Sample durations at each occurrence
      var speakDuration = gaussian(2000, 1000);
      var silentDuration = gaussian(1000, 500);
      delay(state === "speak"
        ? speakDuration
        : silentDuration
      ),
      map(function (s) {
        // State transition function
        return makeHuman(s === "speak"
          ? "silent"
          : "speak"
        );
      }),
      switchAll()
    )
  );
};
```

```
var human = makeHuman("silent");

// human emits:
// "silent" at 0ms
// "speak" at a sampled milliseconds from
//   gaussian(1000, 500)
// "silent" at the previous event timestamp
//   plus a sampled milliseconds from
//   gaussian(2000, 1000)
// "speak" at the previous event timestamp
//   plus a sampled milliseconds from
//   gaussian(1000, 500)
// ...
```

The example uses the syntax of [RxJS] and [WebPPL]. For gentle introductions, check out [this reactive programming tutorial] by Andre Staltz and [this probabilistic programming tutorial] by Adrian Sampson.

Sketching: The Human Simulator PFRP with **Holes**

```
var makeHuman = function(state) {
  return merge(
    of(state),
    of(state).pipe(
      // Sample durations at each occurrence
      var speakDuration = gaussian(2000, 1000);
      var silentDuration = gaussian(1000, 500);
      delay(state === "speak"
        ? speakDuration
        : silentDuration
      ),
      map(function (s) {
        // State transition function
        return makeHuman(s === "speak"
          ? "silent"
          : "speak"
        );
      }),
      switchAll()
    )
  );
};
```

For an introduction to Sketching, checkout Program Synthesis is Possible.

Step 1. Define "hole" random variables

```
...
  // Sample durations at each occurrence
  var h1 = uniform(0, 10000);
  var h2 = uniform(0, 10000);
  var speakDuration = gaussian(h1, 1000);
  var silentDuration = gaussian(h2, 500);
...
    // State transition function
    h3 = flip(0.5);
    return makeHuman(h3
      ? // 1st transition function
        s === "speak"
        ? "silent"
        : "speak"
      : // 2nd transition function
        s === "speak"
        ? "hesitate"
        : s === "hesitate"
        ? "silent"
        : "speak"
    );
    // should define hesitateDuration
    //   for the 2nd transition function
...
```

Step 2. "Fill"-ing holes via probabilistic inference, e.g., MAP

Human Simulator and Robot Behavior Authoring Workflow

1. Define a target human-robot interaction and **create an initial robot program and a human simulator sketch**.
2. **Collect input and output traces** from human-robot or human-human interactions.
3. **Synthesize the human simulator program** with the collected traces.
4. Update the robot behavior.
5. Repeat 2.-4. until satisfied.

Future Work

- Other synthesis techniques
- Human simulator domain-specific language design
- More applications
- Different workflow

Thank you!

For more details, checkout this short paper (4pgs)!