**Problem**

The authors studied inconsistencies in interpreting the behavior of trigger-action programs and errors made in creating programs with desired behavior.

**Examples**

1. If every day at 12PM, then send me an email.
2. If exit a room, then turn of the lights.
3. If I am sleeping, then turn off the radio.

**Ambiguities**

There are ambiguities in the interpretation of when triggers occur, whether the conjunction of two triggers is meaningful to people, and whether sustain actions will revert automatically.

Example 1: If I am sleeping, then turn off the radio.  
There is ambiguity on this trigger occurring, because written this way, it can trigger at any time while the user is sleeping, versus the moment they fall asleep.

Example 2: If the doorbell rings and the sun is setting, then send me an email.  
There is ambiguity in this conjunction because the doorbell may never ring at the exact moment the sun is setting.

**Study I**

**Purpose**  
To begin understanding how users interpret different trigger and action types. The purpose of these questions was to evaluate whether users expected actions to occur according to the logical interpretations of the triggers.

**Size of study**  
60 people

**Methods**

Interpretation of trigger-action programs consisted of reading a text description of the program and then answering multiple-choice questions that assessed the understanding of the program. Throughout the study they tried to minimize variance between the questions by using a uniform set of triggers and actions. There were 9 such questions: 2 single event triggers, 2 single state triggers, 3 for combinations of one event trigger and one state trigger, 1 which combined the two event triggers and 1 which combined the two state triggers.

**Results**

The responders had different expectations for when the actions should be triggered depending on whether the triggers were an event trigger or a state trigger. 75.8% said the rule would start as soon as the event occurred, and for single state triggers only 36.7% said that the rule would start as soon as the state became true. Expectations varied widely for multiple state triggers and multiple event triggers were technically valid.

**Study II**

**Purpose**

The purpose of this study was to see how the rule creation process in context of a fully implemented interface could positively impact the user's mental model of how the program should behave. This interface resembled visual aspects of IFTTT, as well as the workflow for creating rules.

**Size of study**

42

**Methods**

This study did not ask users to synthesize rules themselves, instead they designed a TAP interface and conducted a second study. It was designed to feature multiple triggers with different triggers and action types which also resembled IFTTT. This study contained 5 program creation questions and 5 multiple choice questions about the participant's interpretation of the given rule. The participants were given a description of a desired behavior and were asked to create one or more rule for the smart home to achieve that behavior.

**Results**

It was found that multiple event triggers were used in practice, Users had a varied mental model for state triggers, users disagreed on sustained actions and forgot to undo them because the multiple-choice questions reveled that more people thought that sustained actions would be undone automatically when the trigger was a state compared to when the trigger was an event. The most common mistake was to not limit the time the rule was active.

**Solutions**

1. Automatically detect trigger-action conflicts and resolve.

2. Provide simulated (software) environment for testing rules.

3. Provide rule wizard, guide user through process to generate well-formed rules.

4. Ask for feedback from user when action is triggered, adjust if needed.

5. Human-in-the-loop: upload rules to human expert for review.

6. Constrain trigger-action vocabulary – i.e. eliminate ambiguous triggers such as “when I am sleeping”.

7. Conducting preliminary survey on usability, to better understand the functionalities.

8. Creating virtual environments or simulations