Formal Verification of Autonomous Robotic Assistants

Abstract of *Toward Reliable Autonomous Robotic Assistants Through Formal Verification: A Case Study* by Matt Webster et al.[1]

Balázs Ludmány (ZG85W7)

November 26, 2018

1 Topic

The original article focuses on robotic assistants. These machines replace human labor in many areas ranging from personal healthcare to manufacturing jobs. This requires a high degree of safety and trustworthiness because they operate in close proximity of their human operators. The aim of the Trustworthy Robotic Assistants (TRA) project is to develop techniques that verify and validate robotic assistants against these requirements. The paper presented a case study on the use of formal verification to an autonomous robotic assistant called Care-O-bot deployed at the University of Hertfordshire's Robot House.

2 Tools

Model checking is used as a form of formal verification. First a nondeterministic model is created based on the system observed. Then a program called a model checker exhaustively analyzes all possible executions of the model, determining whether a certain property holds. These properties are derived from requirements.

The researchers used a model checker called SPIN, which stands for Simple PROMELA Interpreter. PROMELA in turn is short for Process Meta-Language, a programming language that lets you describe the system to be verified. Instead of using this language directly they utilized an agent modeling language and simulation environment called Brahms, and the BrahmsToPromela translator program.

Brahms is a very feature-rich language for specifying the behavior of multi-agent systems. It supports advanced techniques like inheritance, message passing and probabilities. The basic building blocks of the language are workframes, which closely resemble typical `IF-THEN` rules.

3 Parts of the model

The University of Hertfordshire's *Robot House* is a suburban three-bedroom house near Hatfield, U.K. It is fully furnished but also contains more than 50 sensors that provide real-time information on the sate of the house and its residents. These range across electrical (e.g., refrigerator door open/closed sensor), furniture (e.g., cupboard drawers open), services (e.g., detect when toilet flush is being used), and pressure (e.g., chair sensors to detect when someone is seated) devices.

The building also houses various autonomous robots to experiment with human-robot interaction. One of these is the *Care-O-bot*, a mobile robotic assistant developed to support people in domestic environments. It has an arm with 7-degree-of-freedom with a 3-finger gripper at the end, stereo cameras serving as "eyes", LED lights and a tray. The robot is aware of its location, as well as the state of the arm and the tray. Text-to-speech synthesis allows it to output a given text as audio.

The robot is controlled by a *script server* through *commands* like "raise tray" or "move to location". These high-level commands are then interpreted by the robot's own software. The decision making is governed by *rules*, these consist of a *guard* statement and a list of the aforementioned commands. A guard is a sequence of propositional statements linked by Boolean AND and OR operations. The list of commands is performed only if the guard statement holds true.

The guards are implemented as a sequence of SQL queries. The table `Sensors` stores the values of the sensors with unique `sensorId`-s. In the rule bellow the robot checks whether variable 500 (trayRaised) and 504 (trayEmpty) are set to 1 (true). If so, then the Care-O-bot will change the light color to yellow (indicating that the robot is in motion), set the tray to the lowered position, wait for completion, set the light to white (indicating that the robot has stopped moving), and wait for completion. Finally, the variables 500 (trayRaised) and 501 (trayLowered) are set to "false" and "true" respectively.

```
SELECT * FROM Sensors WHERE sensorId=500 AND value = 1
 &
SELECT * FROM Sensors WHERE sensorId=504 AND value = 1
==
light,0,yellow
tray,0,down
wait
light,0,white
wait
cond,0,500,0
cond,0,501,1
```

The Care-O-bot's rule database is composed of a variety of rules for determining a wide range of autonomous behaviors, like answering the door or reminding a person to take medication. The Robot House database consists of 31 default rules.

4 Definition of the model

4.1 Modeling the Care-O-bot

The 31 default rules were implemented in Brahms. The main construct in the language that allows this is the "workframe", which specifies a sequence of things to be executed when a given condition holds. The rule given in the previous example can be translated to the following workframe:

```
workframe wf_lowerTray {
repeat: true;
priority: 10;
when(knownval(current.trayIsRaised = true)
  and knownval(current.trayIsEmpty = true))
do{
  conclude((current.lightColour = current.colourYellow));
  lowerTray();
  wait();
  conclude((current.lightColour = current.colourWhite));
  wait();
  conclude((current.trayIsRaised = false));
  conclude((current.trayIsLowered = true));
}
```

The workframe is allowed to "repeat" meaning that it can be used multiple times by the agent. If multiple workframes are eligible for execution, the one with the highest priority is executed. The `when a do {b}` construct states what actions should be taken if the conditions are true. `conclude` can be used to set a certain belief (variable). `wait` and `lowerTray` are primitive actions that take a certain amount of time to finish.

User input is modeled with a separate "person" agent. The Care-O-bot would for example approach the user to ask whether she would like to watch television. The possible answers would be presented on a touch screen. In the model a special activity is used to query the person, then it responds by simulating simple human behavior, always choosing to watch TV for instance.

4.2 Modeling the Robot House

Besides the Care-O-bot and the person being assisted, a *scenario* also contains a model of the Robot House. This determines the boundaries of the model and the range of possibilities. The layout of the building is encoded using a *geography* in Brahms, providing a hierarchy of places an agent can be at. For example:

```
areadef House extends BaseAreaDef { }
areadef Room extends House { }
areadef areaOfInterest extends Room { }
area LivingRoom instanceof Room partof House { }
area LivingRoomTV instanceof areaOfInterest partof LivingRoom { }
```

The type `BaseAreaDef` is built into Brahms, and is the base of other area types. The types are then instantiated and placed in the hierarchy.

The scenario starts at noon and ends at 9 P.M. At 5 P.M. the person needs to take medication. The person may choose to watch TV, move into the living room, move into the kitchen, or send the Care-O-bot to the living room or the kitchen at any given time. This nondeterministic behavior is implemented with 5 different workframes within the person actor, each with different priorities. Every modification of a belief (`conclude` keyword) has a certain probability. If it is modified the Care-O-bot is notified, and the next workframe fires otherwise.

5 Formal verification of the model

The authors used the *BrahmsToPromela* translator to translate the previously defined Brahms model into PROMELA. This model was then verified against a set of formalized requirements using the SPIN model checker.

SPIN uses linear temporal logic to formalize requirements. The following operators are used in the article:

- $\Box$ now and at all points in the future
- $\Diamond$ now or at some point in the future
- $\bigcirc$ in the next state
- B belief operator. An extension by BrahmsToPromela. $\mathsf{B}_{\text{Care-O-bot}}x$ means that Care-O-bot believes x is true.

The article gives the following sample requirements and their translations:

1. It is always the case that if the Care-O-bot believes that the person has told it to move into the kitchen, then it will eventually move into the kitchen

$$\Box\left[\mathsf{B}_{\text{Care-O-bot}}\left(\mathsf{B}_{\text{Person}}\text{guiGoToKitchen}\right) \Rightarrow \Diamond \mathsf{B}_{\text{Care-O-bot}}\left(\mathsf{location} = \text{Kitchen}\right)\right]$$

2. It is always the case that if the Care-O-bot believes that the person has told it to move to the sofa in the living room, then it will eventually move there

$$\Box\left[\mathsf{B}_{\text{Care-O-bot}}\left(\mathsf{B}_{\text{Person}}\text{guiGoToSofa}\right) \Rightarrow \Diamond \mathsf{B}_{\text{Care-O-bot}}\left(\mathsf{location} = \text{LivingRoomSofa}\right)\right]$$

3. It is always the case that if the Robot House believes that the sofa seat has been occupied for at least 1 hour, and if the Robot House believes that the television power consumption is higher than 10 W, and if the Care-O-bot believes that it has not yet asked the person if he or she wants to watch television, then eventually, the Care-O-bot will move to the living room sofa and ask the person if he or she wants to watch the television

$$\begin{gathered}\Box\left[\mathsf{B}_{\text{RobotHouse}}\text{sofaOccupied1Hour} \wedge \mathsf{B}_{\text{RobotHouse}}\text{tvPower} > 10 \wedge \mathsf{B}_{\text{Care-O-bot}}\neg\text{askedToWatchTv}\right. \\ \Rightarrow \\ \left.\Diamond\left(\mathsf{B}_{\text{Care-O-bot}}\left(\mathsf{location} = \text{LivingRoomSofa}\right) \wedge \mathsf{B}_{\text{Care-O-bot}}\text{askedToWatchTV}\right)\right]\end{gathered}$$

4. It is always the case that if the time is 5 P.M., then the Care-O-bot will believe that the medicine is due

$$\Box\left[\mathsf{B}_{\text{Clock}}\left(\mathsf{time} = 5 \text{ P.M.}\right) \Rightarrow \Diamond \mathsf{B}_{\text{Care-O-bot}} \text{medicineDue}\right]$$

The article states that the requirements were successfully verified on a consumer grade laptop. It shows that this type of model checking is a viable option for verification as it does not require high performance computers.

6 Improvements on the environment models

In the previous sections the Care-O-bot was verified with simulated human behavior. The article describes three different methods of taking actual behavior into account. This richer environment model is based on data collected from the sensors in the actual Robot House. One person occupied the house for 4 days and 11 hours, and every sensor output was logged during this time.

6.1 Deterministic Environment Model

In this model every entry of the user activity log was directly converted into Brahms workframes. Every workframe had the same priority, and the time was a condition in the `when` clause. From a model-checking perspective this deterministic behavior can be seen as a single run of a nondeterministic model.

6.2 Extended Nondeterministic Environment Model

This one follows the logic of the original environment model. There are a list of possible actions a person can take, but the choice is nondeterministic. However, the list of actions was extended based on the data collected from the Robot House. The authors identified a total of 26 different activities based on the logs. These were implemented in Brahms and checked in SPIN successfully.

6.3 Nondeterministic Conjoined Activity Environment Model

Examination of the activity logs showed that a few activities overlap. The article gives the example of going into the bathroom and then using the toilet. These were joined into compound activities like `InBathroom_Toiletting`. A total of 133 conjoined activities were created this way. The requirements could still be checked, although with increased computational resources.

7 Conclusion

The authors demonstrated the application of model checking as a form of formal verification in the autonomous robotic assistant domain. They modeled different scenarios using the Brahms language, the model consisted of one of their existing robots and its environment. They found that this environment model can be greatly improved upon using sensor data from the Robot House hosting the real world equivalent of the assistant. Every iteration of the model was successfully checked against a set of requirements derived from real world requirements.

References

[1] M. Webster, C. Dixon, M. Fisher, M. Salem, J. Saunders, K. L. Koay, K. Dautenhahn, and J. Saez-Pons. Toward reliable autonomous robotic assistants through formal verification: A case study. *IEEE Transactions on Human-Machine Systems*, 46(2):186–196, April 2016.