Model-Based System Design
Project of Firefighter Robot Model development

Junxiong Wang, Wei Ma, Alexander Mylnikov

EPFL

Abstract. For many particular problems it is nontrivial to design a correct model architecture. The course project requests to model a firefighter robot, which is needed to be fully validated. On the other hand the course project pushes to make a connection between theoretical knowledge taught on lectures and practice experience of building model systems. Report covers stages of specification and requirements definition, system design explanation, code generation of BIP simulation, SMV translation and system verification.

Keywords: Model Checking, System Design, BIP, CTL, NuXmv

1 Requirements

Characteristics of model's world:

- Environment is flat 2D space with basic item of square cell.
- Environment surrounded by impermeable wall.
- Environment has finite borders.
- Each cell has temperature value.
- Environment is in firestorm.
- Firestorm never ends.
- Firestorm can change its location.
- Robot has a limit of temperature which causes it's death.
- There are victims to save for robot on environment.

Minimal safety rules for robot:

- Robot must not advance and rotate at the same time.
- Robot must not leave its predefined mission area.
- Robot must not crash into walls.
- Robot must remain in zones, where the heat intensity is below the robots failure level.
- When robot finds a victim it must transmit their location to the rescue mission control center.
- Robot operation must always be based on recent measurement information

According to the task statement we construct following requirement for model system in general:

- Component model with operational semantics should be presented.
- Verification and safety validation should be done.
- Executable code should be generated.
- Additional assumptions could be made.
- The model system shouldnt have any deadlocks.
- The model system should be modular.

Basic specification of robot system from side of physical abilities with respect to previous requirements:

- Robot is square with size of one cell.
- Robot has 4 sensors by each edge.
- All robots equipment lays inside the Robot.
- Robot could move only forward (by one edge).
- Robot could rotate one direction.

From the side of the modular structure of the robot should have following components:

- Engine
- Heat sensor
- Navigation
- Transmitter

2 Specification

To clarify and avoid any obscure points about the project tasks, we put following assumptions to the World:

- The cell where the robot stands cannot become in fire.
- No modules could fail.
- A sensor could measure any temperature and even temperature which is over the limit.
- The world has no time.

With them the World is ready for correct model design. To clarify it is said that all assumptions are close to reality and make needed level of abstraction.

3 Design

Robot system is made of six components:

1. Engine
2. Heat sensor
3. Navigation System
4. Transmitter
5. Buffer

6. Searching Algorithm (Searcher)

Engine is the core component and it cooperates with the other components. Four heat sensors are installed in the robot to measure temperature in four directions, up, down, right and left. Each sensor has it's own buffer. The engine gets temperature information from buffers. The engine synchronizes with four buffers to get the temperature in four directions (left, right, up, down). Navigation System detects if the robot faced a wall or the border of the region it is confined to and whether a victim has been found. A message is used to represent the output of the Navigation System, which is a command that the engine gets from it. Searching Algorithm(Searcher) is responsible for conducting the robot where to move and search victim position. It is a black box in our design with communication port. It has no details about structure. If the different components have the same name export port, it means they are the synchronized in action.

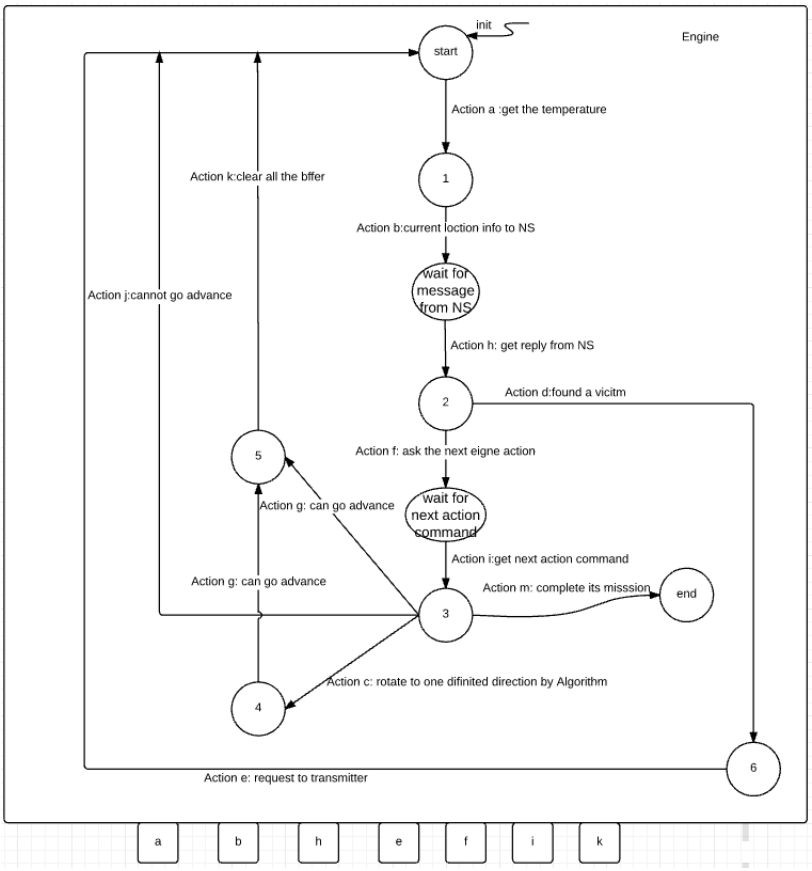

Fig. 1. The Engine

The figure 1 demonstrates details of the engine. It has nine states and thirty transitions. The list below contains it's transitions. The engine could go advance or rotate. The Navigation System and the Searching Algorithm (searcher) give information to the engine. The engine gets the temperature information from the buffers.

Action Engine.a Synchronized transition between the Engine and four buffers. The Engine gets temperature measurements after the transition.

Action Engine.b Synchronized transition between the Engine and the Navigation System. The Engine gives The Navigation System current situation around.

Action Engine.c Internal transition. After the Engine gets the next action command from the Searching Algorithm, if the command contains any rotate requirement, the robot executes rotating action.

Action Engine.d Internal transition. After the engine gets a victim presence message from the Navigation System, the action would be taken.

Action Engine.e Synchronized transition between the Engine and the Transmitter if there is a victim.

Action Engine.f Synchronized transition between the Engine and the Searching Algorithm. Temperature information and message from the Navigation System is input of the Searching Algorithm.

Action Engine.g Internal transition. If the Searching Algorithm tells the Engine that it could go in advance, the transition is been executed.

Action Engine.h Synchronized transition between the Engine and the Navigation System. The Navigation System gives a message to the Engine.

Action Engine.i Synchronized transition between the Engine and the Searching Algorithm. The Searching Algorithm gives a command to the Engine.

Action Engine.j Internal transition. If the Searching Algorithm tells the Engine that it cannot go in advance, the transition is been executed.

Action Engine.k Synchronized transition between the Engine and the Buffer. When the Engine arrives to a new position (cell), before it gets temperature measurements from buffers. Buffers cleans old data.

The figure 2 demonstrates the structure of the Navigation System (NS). The Navigation System checks the environment around the robot and gives feedback to the Engine.

Action NS.b Synchronized transition between the NS and the Engine. When the engine request the NS to give an guide, the action will be executed.

Action NS.h Synchronized transition between the NS and the engine. The NS gives an message to the Engine.

Action NS.x Internal transition. If the NS doesn't detect a victim, the action is been taken.

Action NS.s Internal transition. If the NS detects a victim, the action is been taken.

The figure 3 shows structure of the Transmitter. The Transmitter receives a message that victim was found and it sends a message to the mission center.

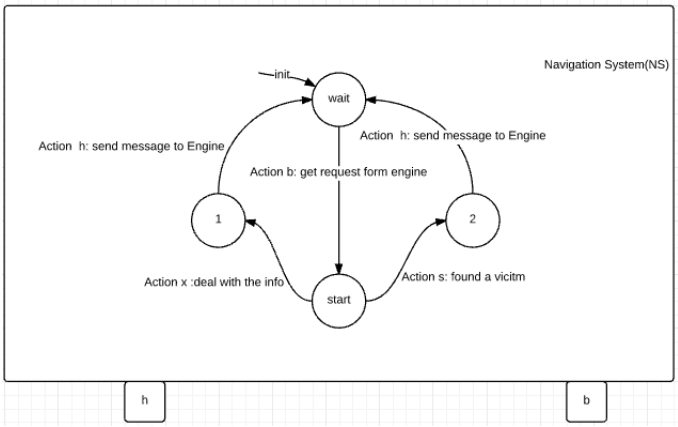

Fig. 2. The Navigation System

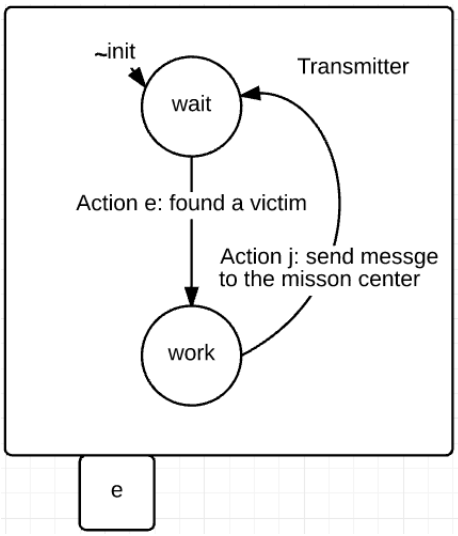

Fig. 3. The Transmitter

Action Transmitter.e Synchronized transition between the Transmitter and the Engine. If the Engine tells the Transmitter that there is a victim, the Transmitter executes action.

Action Transmitter.j Internal transition. The Transmitter sends a message that contains the location of the victim to the mission center.

The figure 4 shows the structure of the Sensor. The Sensor measures temperature periodically (in particular computation moments) and then sends new measurement data to it's own buffer.

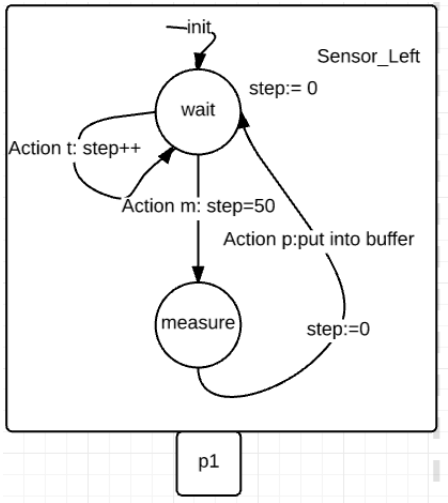

Fig. 4. The Sensor

Action Sensor.m Internal transition. If the computational steps gets over the threshold, the Sensor measures temperature.

Action Sensor.t Internal transition. Computational steps are increased by one step.

Action Sensor.p Synchronized transition between the Sensor and the Buffer. The Sensor puts new temperature measurement to the Buffer.

The figure 5 shows the structure of the Buffer. The Buffer receives new measurement data from the Sensor and in any state to make the Engine could read the latest measurement data.

Action Buffer.a Synchronized transition between the Buffer and the Engine. The engine gets last temperature measurement from the Buffer.

Action Buffer.k Synchronized transition. If the Buffer receives request from the Engine to clear it's internal buffer the action is been executed.

Action Buffer.p Synchronized transition between the Sensor and the buffer. The Sensor puts new temperature measurement to the Buffer.

The figure 6 shows structure of the Searching Algorithm component. It accepts the input from the engine and gives back the output to the engine.

Action SA.f Synchronized transition between the Searching Algorithm and the Engine. The search algorithm accept the input from the engine.

Action SA.i Synchronized transition between the Searching Algorithm and the Engine. the Searching Algorithm gives the feedback to the Engine.

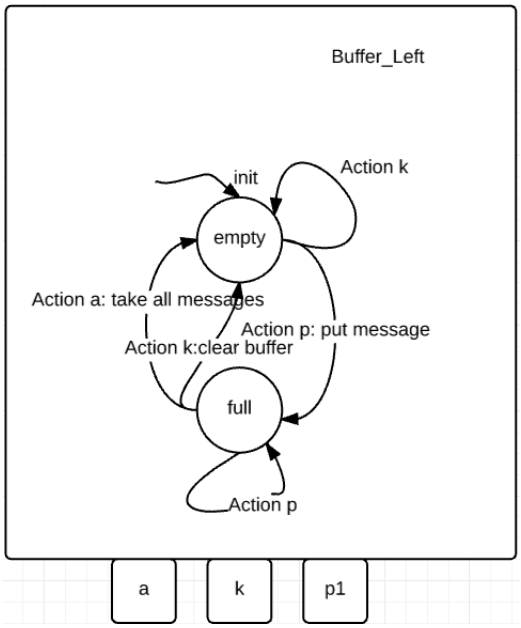

Fig. 5. The Buffer

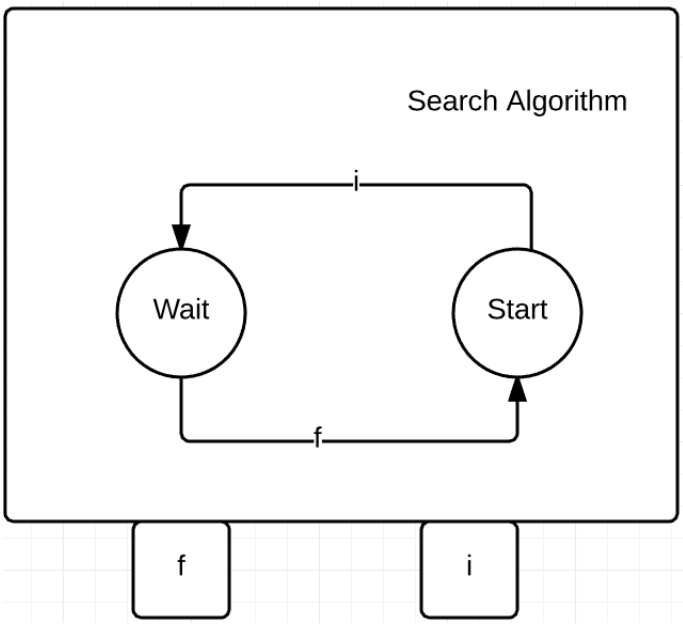

Fig. 6. The Search Algorithm(SA)

4 Code Generation

We implemented two BIP-code versions of simulation, the old BIP syntax version and the new BIP one. We used the new BIP version to simulate our model and the old one to verify the model. In the new BIP version, rand (random) function is used to simulate input from the environment such as temperature data and the map. We used the BIP translator to convert the old BIP simulation code to SMV verification file.

5 Verification

Because of large number of states, it is needed vast amount of time to verify our model. It is used to be a NP-hard problem. Therefore, we analyze the trace file. All sates could be reached. Robot physically could die, however in case of correct Search Algorithm and other component behavior this state is unreachable. It is crucial for verification to check whether our implementation satisfies all requirements. The process of our verification introduced below.

5.1 Translate BIP to SMV

The provided translator tool doesn't support the new version BIP syntax, it was needed to modify our BIP file to respect old version syntax. The generated SMV file was used for verification with nuXmv tool. Pic

5.2 Simulation and Checking Deadlock-free

The simulation commands.

```
nuXmv -int output.smv
go
pick_state -r
simulate -r -k 4000
show_trace -o log.txt
check_fsm
computer_reachable
```

The simulation result is shown on 7 8

5.3 Verification of specifications

1. The robot can't go in advance if temperature measurement is over the certain limit.

$$AG((engine.frontTemperature \geq threshold) \rightarrow AX(\neg enable(GoAhead))$$

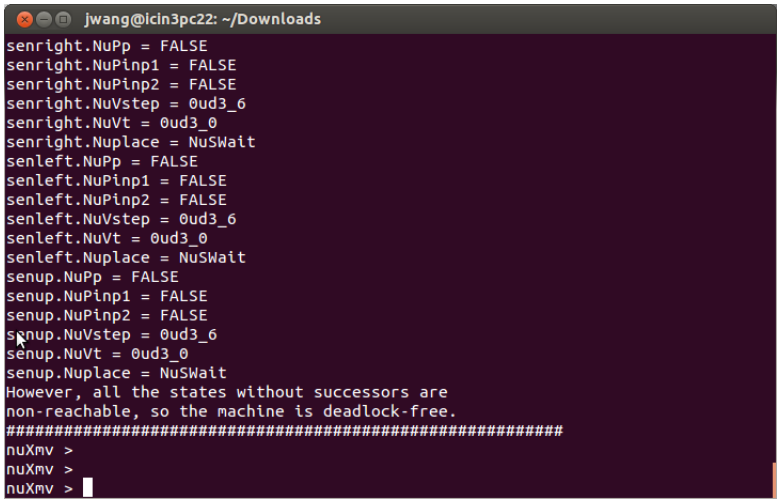

Fig. 7. Check deadlock

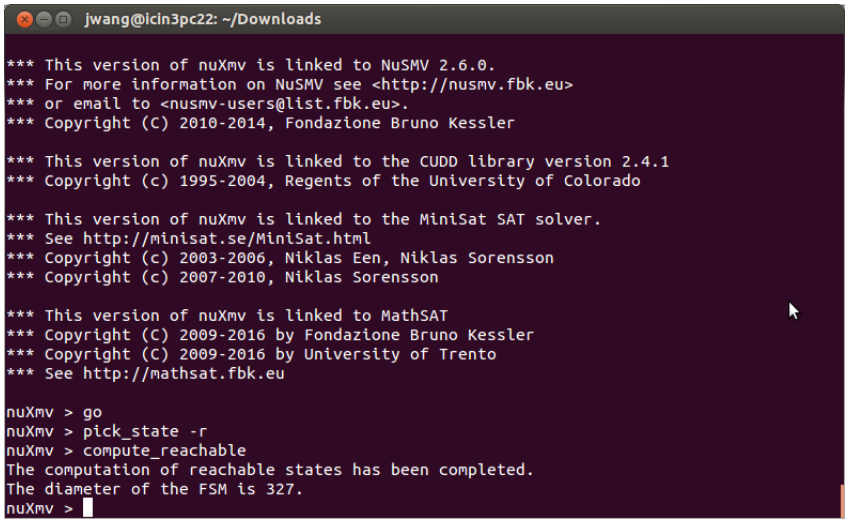

Fig. 8. Check Reachable State

2. The robot must not crash into walls.

   $AG((engine.navigationInfo == WallFront) \rightarrow AX(\neg enable(GoAhead)))$

3. Once the robot finds a victim, it must transmit the location to the mission control centre.

   $AG((engine.state = findVictim) \rightarrow AX(transmitter.state = sendMessage))$

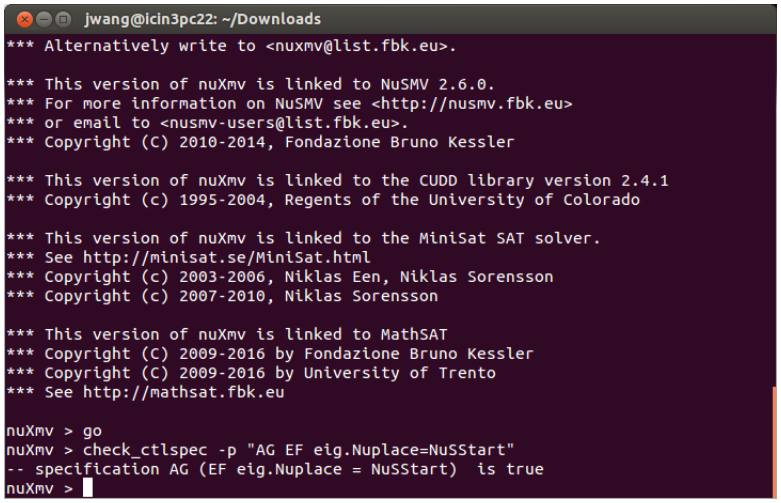

Fig. 9. Verify Requirement

4. Once the robot go in advance, it should measure temperature and send a command to navigation system again, as it shown on 9

$$AG(EF(engine.state = startMeasure))$$

5. Deadlock-freedom for action GoAhead

$$AG(EF(enable(a)))$$

6. No Starvation for action rotate

$$AG(AF(after(a)))$$

We are able to verify many other actions and states, however the CTL rules are similar for them. The list above contains main types of verification approaches. The verification was physically performed with our obtainable computational resources.

6 Conclusion

During the project, programmed and verified model-bases system of firefighter robot abstraction. The requirements of the project are respected. Some necessary assumptions are introduced. The main goal of robot is achieved. The system is proved to be deadlock-free. The system model is ready to perform rescue operation with The Search Algorithm module.
Our team achieve full connection between course's theoretical knowledge and practical experience of construction of the model-based system.
During the project process we used BIP simulation language, BIP execution tool, SMV verification tool.