Verification mini-project

Crossing the River

Jacob Karsten Wortmann

Sam Sepstrup Olesen

Nicklas Andersen

sw805f14

May 14, 2014

Introduction

The purpose of the exercise is to model a game, Crossing the River, in UPPAAL. The goal of the game is to get a group of people from one riverside to the other. The group consists of two daughters, two sons, a mom, a dad, a police officer, and a thief. The game has a list of rules that must be followed at all times to complete the game:

- Max 2 persons on the boat,

- Mom not alone with boys,

- Dad not alone with girls,

- Thief not alone with family,

- Only police officer, dad and mom can handle the boat.

Boat

Figure 1 shows the automata of the boat. The rules of the game defines that an adult needs to be on the boat for it to sail. To conform to this rule our automata require an adult to embark to boat before a second person is allowed to embark. When people on the boat disembark, we check whether this new state conforms to the rules. If it is an allowed state we update the location of the boat. If it is not a valid state the automata goes to a `gameover` location, which will cause a deadlock.
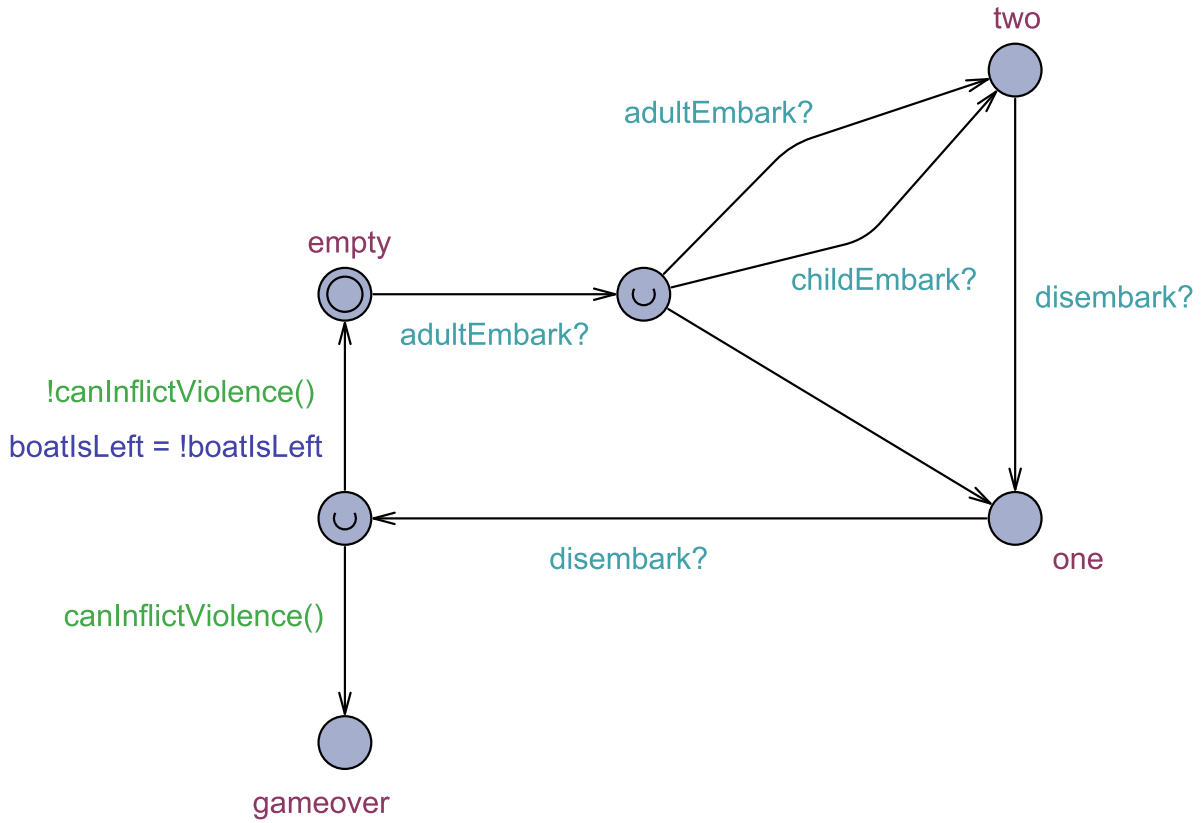
Figure 1: Boat automata.

Global declarations

```
1  clock time;
2
3  chan adultEmbark, childEmbark, disembark;
4
5  bool boatIsLeft = true;
6
7  bool policeIsLeft = true;
8  bool thiefIsLeft  = true;
9  bool dadIsLeft     = true;
10 bool momIsLeft     = true;
11 bool boy1IsLeft    = true;
12 bool boy2IsLeft    = true;
13 bool girl1IsLeft   = true;
14 bool girl2IsLeft   = true;
15
16
17 bool canInflictViolence() {
18     if ((((momIsLeft == boy1IsLeft) || (momIsLeft == boy2IsLeft))
```

```
19          && (momIsLeft != dadIsLeft))
20            return true;
21
22      if (((dadIsLeft == girl1IsLeft) || (dadIsLeft == girl2IsLeft))
23        && (dadIsLeft != momIsLeft))
24            return true;
25
26      if (((thiefIsLeft == boy1IsLeft) || (thiefIsLeft == boy2IsLeft)
27        || (thiefIsLeft == girl1IsLeft) || (thiefIsLeft == girl2IsLeft)
28        || (thiefIsLeft == dadIsLeft) || (thiefIsLeft == momIsLeft))
29        && (thiefIsLeft != policeIsLeft))
30            return true;
31
32      return false;
33 }
```

Listing 1: Global declaration.

In the the global declarations we define a clock `time` which can be used to see how fast a given trace is. `time` is not used by the system.

The global declaration also declares the channels and defines the location of the boat and all the people. `canInflictViolence` is a function that can be used to check whether the current state is invalid.
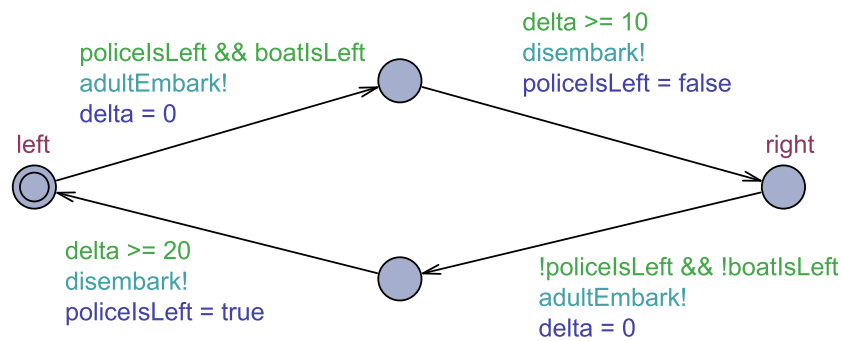
Police & Thief

Figure 2: Police automata.

Parents

```
1 clock delta;
2
```
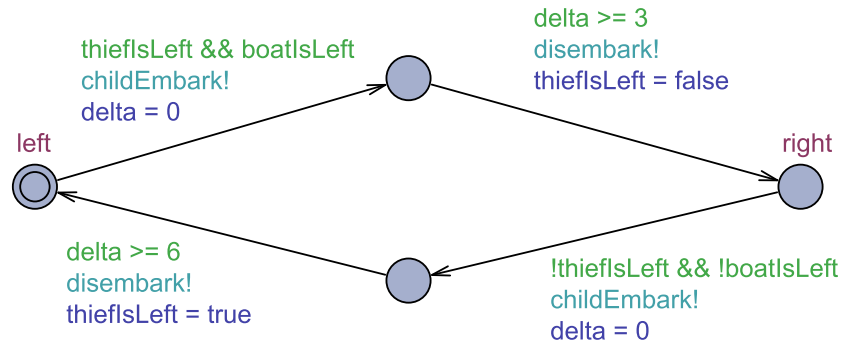
Figure 3: Thief automata.
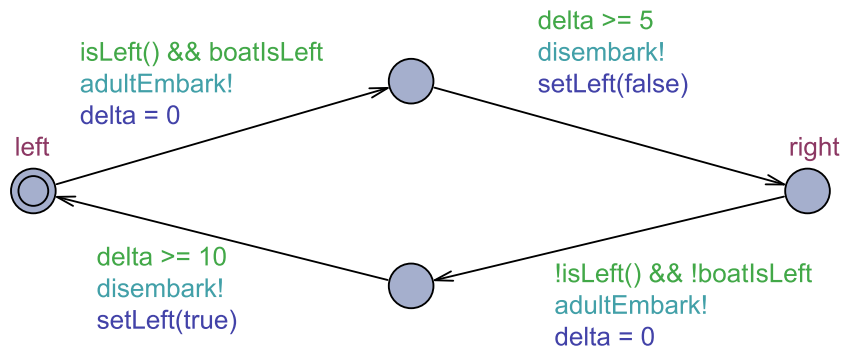

Figure 4: Parent automata.

```
 3  bool isLeft() {
 4      if (isDad)
 5          return dadIsLeft;
 6      else
 7          return momIsLeft;
 8  }
 9
10  void setLeft(bool left) {
11      if (isDad)
12          dadIsLeft = left;
13      else
14          momIsLeft = left;
15  }
```

Listing 2: Parent declaration.

Children

```
 1  clock delta;
```
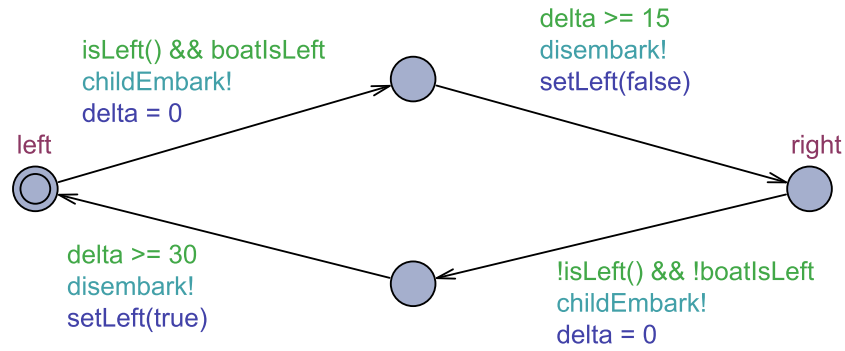
Figure 5: Child automata.

```
 2
 3  bool isLeft() {
 4      if (isBoy)
 5          if (isFirst)
 6              return boy1IsLeft;
 7          else
 8              return boy2IsLeft;
 9      else
10          if (isFirst)
11              return girl1IsLeft;
12          else
13              return girl2IsLeft;
14  }
15
16  void setLeft(bool left) {
17      if (isBoy)
18          if (isFirst)
19              boy1IsLeft = left;
20          else
21              boy2IsLeft = left;
22      else
23          if (isFirst)
24              girl1IsLeft = left;
25          else
26              girl2IsLeft = left;
27  }
```

Listing 3: Child declaration.

More children???

```
 1  clock time;
```

```
 2
 3  chan adultEmbark, childEmbark, disembark;
 4
 5  bool boatIsLeft = true;
 6
 7  bool policeIsLeft = true;
 8  bool thiefIsLeft  = true;
 9  bool dadIsLeft    = true;
10  bool momIsLeft    = true;
11  bool boy1IsLeft   = true;
12  bool boy2IsLeft   = true;
13  bool boy3IsLeft   = true;
14  bool girl1IsLeft  = true;
15  bool girl2IsLeft  = true;
16
17
18  bool canInflictViolence() {
19      if (((momIsLeft == boy1IsLeft) || (momIsLeft == boy2IsLeft)
20        || (momIsLeft == boy3IsLeft))
21       && (momIsLeft != dadIsLeft))
22          return true;
23
24      if (((dadIsLeft == girl1IsLeft) || (dadIsLeft == girl2IsLeft))
25        && (dadIsLeft != momIsLeft))
26          return true;
27
28      if (((thiefIsLeft == boy1IsLeft) || (thiefIsLeft == boy2IsLeft)
29      || (thiefIsLeft == boy3IsLeft)
30      || (thiefIsLeft == girl1IsLeft) || (thiefIsLeft == girl2IsLeft)
31      || (thiefIsLeft == dadIsLeft) || (thiefIsLeft == momIsLeft))
32      && (thiefIsLeft != policeIsLeft))
33          return true;
34
35      return false;
36  }
```

Listing 4: Global declarations with an extra boy.