Formal Methods
Fall 2017

S03

Professor : Ultes-Nitsche Ulrich
Assistant : Christophe Stammet

Submitted by Sylvain Julmy

Exercise 1

$$
\begin{aligned}
F \equiv &(C \vee \neg A) \wedge D \wedge (D \vee A) \wedge (E \vee B) \wedge \\
&(\neg C \vee A \vee B) \wedge E \wedge (\neg D \vee \neg A) \wedge \\
&(C \vee \neg B) \wedge (\neg E \vee \neg B) \wedge C
\end{aligned}
$$

Applying the resolution

Using A

$$
\begin{gathered}
(C \vee \neg A) \wedge (D \vee A) \wedge (\neg C \vee A \vee B) \wedge (\neg D \vee \neg A) \wedge \cdots = \\
(C \vee D) \wedge \underbrace{(\neg D \vee D)}_{\top} \wedge \underbrace{(C \vee \neg C \vee B)}_{\top} \wedge (\neg C \vee \neg D \vee B) \wedge \cdots = \\
(C \vee D) \wedge (\neg C \vee \neg D \vee B) \wedge (E \vee B) \wedge (C \vee \neg B) \wedge (\neg E \vee \neg B) \wedge D \wedge E \wedge C
\end{gathered}
$$

Using C

$$
\begin{gathered}
(C \vee D) \wedge (\neg C \vee \neg D \vee B) \wedge (C \vee \neg B) \wedge C \wedge \cdots = \\
\underbrace{(D \vee \neg D \vee B)}_{\top} \wedge \underbrace{(\neg B \vee B \vee \neg D)}_{\top} \wedge (\neg D \vee B) \wedge D \wedge E \wedge (E \vee B) \wedge (\neg E \vee \neg B) = \\
(\neg D \vee B) \wedge D \wedge E \wedge (E \vee B) \wedge (\neg E \vee \neg B)
\end{gathered}
$$

Using E

$$
\begin{gathered}
(\neg D \vee B) \wedge D \wedge E \wedge (E \vee B) \wedge (\neg E \vee \neg B) = \\
\underbrace{(B \vee \neg B)}_{\top} \wedge \neg B \wedge D \wedge (\neg D \vee B) = \\
\neg B \wedge D \wedge (\neg D \vee B)
\end{gathered}
$$

Using B

$$
\begin{gathered}
\neg B \wedge D \wedge (\neg D \vee B) = \\
D \wedge \neg D = \bot
\end{gathered}
$$

because $F \equiv \bot$, the formula is not satisfiable.

Exercice 2

We suppose that we have a function $tester(P)$ which test if the program P will stop or not :

- If P stops, tester return $true$
- Else, it return $false$.

Now, we create the following program :

```
tester2(P) =
  if (tester(P)
    then loop forever
    else true
```

Now we call $tester2$ with $tester2$: `tester2(tester2)`. And we have the contradiction : $tester2$ loop forever if and only if $tester$ accept $tester2$ and if and only if $tester2$ end, which is impossible because $tester$ has to accept $tester2$. So it is a proof that the program $tester$ can't exist.

Exercice 3

Formal description of M_1

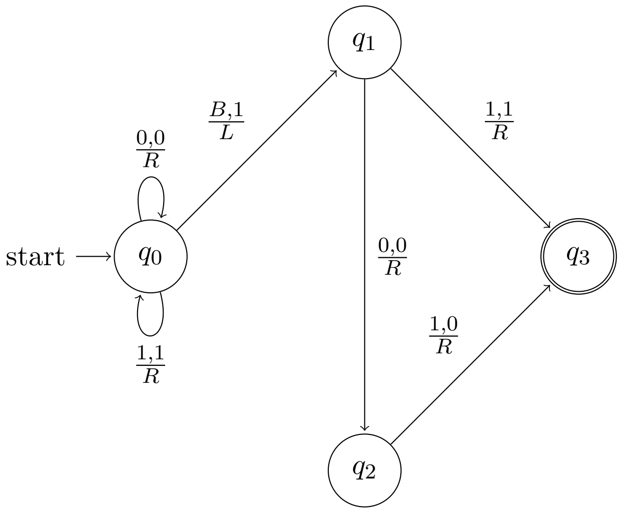
$$M_1 : (\{q_0, q_1, q_2, q_3\}, \{0, 1\}, \{0, 1, B\}, \delta, q_0, B, \{q_3\})$$

where

$$\begin{aligned}
\delta(q_0, 0) &= (q_1, B, R)\\
\delta(q_0, 1) &= (q_2, B, R)\\
\delta(q_1, 0) &= (q_1, 0, R)\\
\delta(q_1, 1) &= (q_2, 0, R)\\
\delta(q_1, B) &= (q_3, 0, R)\\
\delta(q_2, B) &= (q_3, 1, R)\\
\delta(q_2, 1) &= (q_2, 1, R)\\
\delta(q_2, 0) &= (q_1, 1, R)
\end{aligned}$$

This Turing machine implements a right shift on binary number, it multiply a binary number by 2 (or divide it by 2, depending on which direction we represent the number...)

Finite state represantation of M_2



This Turing machine add the last (right most) digit of a number to his end , for example 0101 give 01011 and 0100 give 01000.