

Trabalho 3

Grupo 24

Pedro Faria - A72640

Hugo Costeira - A87976

a

Como é dado no enunciado existe 5 modos, nomeadamente:

START FREE STOPPING BLOCKED STOPPED

Enquanto a velocidade do carro for diferente de 0 ($v_{stopped}$), o carro irá estar num loop entre FREE, STOPPING e BLOCKED.

A mudança de modo FREE e STOPPING é determinada pelos invariantes: $tm < 0.3$ e $V-v < 0.5$ ($tmax_free$ e d_free).

O carro ficará com as rodas bloqueadas quando a diferença de velocidades for menor que 1 m/s (d_stop)

Case as rodas estejam bloqueadas mais que $0.2s$ ($tmax_blocked$) o ABS deixará de travar (passando do modo ***BLOCKED*** para o modo ***FREE***)

O estado inicial do FOTS é derivado facilmente a partir da definição do autómato híbrido. Por exemplo, no caso do ABS temos:

$$m = START \wedge t = 0 \wedge tm = 0 \wedge v = v_0 \wedge V = v_0$$

As transições do FOTS incluem os dois tipos de transição que podem ocorrer num autómato híbrido:

- Transições *timed* descrevem os *flows* associados a cada modo (a evolução das variáveis contínuas)
- Transições *untimed* descrevem os *switches* entre modos

As transições *untimed* podem ser obtidas através de uma codificação muito directa das guardas e efeitos especificadas nos *switches*, com a restrição que o tempo não evolui nestas transições, nem as variáveis contínuas se modificam a não ser que lhes seja explicitamente atribuído um novo valor no efeito do *switch*. Por exemplo, no caso do ABS temos 5 transições deste tipo:

$$\begin{aligned}
m = START \wedge m' = FREE \wedge t' = t \wedge tm' = tn \\
\vee \\
m = FREE \wedge m' = STOPPING \wedge (tm \geq tmax_{free} \vee V - v \geq d_{free}) \wedge tm' = tn \\
\vee \\
m = STOPPING \wedge m' = BLOCKED \wedge t = t' \wedge tm = tm' \wedge v = v' \\
\vee \\
m = BLOCKED \wedge m' = Free \wedge tm \geq tmax_{blocked} \wedge tm = 0 \wedge t
\end{aligned}$$

Nas transições *timed* o modo permanece constante, mas o resto das variáveis evoluem de acordo com as restrições indicadas. Os *flows* são especificados indicando qual a derivada em relação ao tempo de cada variável contínua. Para codificar os *flows* no FOTS é necessário fazer a sua *discretização*, ou seja, indicar qual a variação ocorrida num intervalo de tempo $t' - t$.

$$m = FREE \wedge m' = FREE \wedge tm < tmax_{free} \wedge V - v < d_{free} \wedge t' = t + dt \wedge tm' = tn$$

$$m = STOPPING \wedge m' = STOPPING \wedge V - v \geq tmax_{blocked} \wedge tm = tm' \wedge t' = t + dt$$

$$m = BLOCKED \wedge m' = BLOCKED \wedge tm < tmax_{blocked} \wedge tm +$$

C

```
In [ ]: !pip install z3-solver
!pip install matplotlib
```

```
In [ ]: from z3 import *
import matplotlib.pyplot as plt
Mode, (START,FREE,STOPPING,BLOCKED,STOPPED) = z3.EnumSort ('Mode', ('START','FREE','STOPPING','BLOCKED','STOPPED'))
```

```
In [ ]: def declare(i):
    s = {}
    s['t'] = Real('t'+str(i))
    s['tm'] = Real('tm'+str(i)) #timer
    s['m'] = Const('m'+str(i),Mode)
    s['v'] = Real('v'+str(i))
    s['V'] = Real('V'+str(i))
    return s

def init(s,v):
    return And(s['t']==0,s['tm']==0,s['m']==START,s['v']==v,s['V']==v)
```

In []:

```
def trans(s,p):
    #s= estado atual
    #p= posição

    dt=0.1
    a=0.01 #constante atrito ar
    b=0.7 # atrito corpo
    P=1000 #peso

    #tempo maximo em que está no modo free
    tmax_free=0.3
    #tempo maximo em que está no modo blocked
    tmax_blocked=0.2
    #diferença velocidade free free_stopping
    d_free=0.5
    #diferença velocidade stopping_blocked stopping
    d_stop=1
    #velocidade em que o carro é considerado parado
    v_stopped=0
    #constante porporcionalidade para modo free
    c_free=0.5
    #constante proporcionalidad para stopping
    c_stopping=5

    # Untimed
    #Start -> Free
    start_free= And(s['m']==START,
                    p['m']==FREE,
                    p['t']==s['t'],
                    p['tm']==s['tm'],
                    p['v']==s['v'],
                    p['V']==s['V'])
    )

    #Free -> Stopping
    free_stopping= And(s['m']==FREE,
                        p['m']==STOPPING,
                        Or(s['tm']>=tmax_free,s['v']-s['v']>=d_free),
                        p['tm']==0,
                        s['t']==p['t'],
                        s['v']==p['v'],
                        s['V']==p['V'],
                        s['v']>v_stopped)
    )

    #Stopping -> Blocked
    stopping_blocked= And(s['m']==STOPPING,
                           p['m']==BLOCKED,
                           s['t']==p['t'],
                           s['tm']==p['tm'],
                           s['v']==p['v'],
                           s['V']==p['V'],
                           s['v']-s['v']<d_stop,
                           s['v']>v_stopped)
    )

    #Blocked -> Free
    blocked_free= And(s['m']==BLOCKED,
```

```

        )
#Verifica se esta parado
checkStop= And(Or(s['m']==BLOCKED,s['m']==STOPPING,s['m']==FREE,s['m']==STOPPED,
                  p['m']==STOPPED,
                  s['tm']==p['tm'],
                  s['t']==p['t'],
                  s['v']<=v_stopped,
                  p['v']==0,p['v']==0)

# Timed
free= And(s['m']==FREE,p['m']==FREE,s['tm']<tmax_free,s['v']-s['v']<d_free,
           p['t']==s['t']+dt,p['tm']==s['tm']+dt,
           p['v']==s['v']+(-a*P+c_free*(s['v']-s['v']))*dt,p['v']==s['v'],
           s['v']>v_stopped
         )

stopping= And(s['m']==STOPPING,p['m']==STOPPING,
              s['V']-s['v']>=d_stop,
              s['tm']==p['tm'],p['t']==s['t']+dt,
              p['v']==s['v']+(-a*P+c_stopping*(s['v']-s['v']))*dt,p['v']==p['v'],
              s['v']>v_stopped
            )

blocked= And( s['m']==BLOCKED,p['m']==BLOCKED,
              s['tm']<tmax_blocked,
              s['tm']+dt==p['tm'],s['t']+dt==p['t'],
              p['V']==s['V']+(-a*P-b)*dt,p['v']==p['v'],
              s['v']>v_stopped
            )

return Or(start_free,free_stopping,stopping_blocked,blocked_free,checkStop,:)

```

```

In [ ]: def gera_traco(declare,init,trans,k,v):
    rodas=[]
    time=[]
    carro=[]
    s = Solver()
    traco = [declare(i) for i in range(k)]
    s.add(init(traco[0],v))
    for i in range(k-1):
        s.add(trans(traco[i],traco[i+1]))

    if s.check()==sat :
        m = s.model()
        for i in range(k):
            print("Estado: ",i)

            rodas.append(float(m[traco[i]['v']].numerator_as_long())/float(m[traco[i]['v']].denominator_as_long()))
            carro.append(float(m[traco[i]['V']].numerator_as_long())/float(m[traco[i]['V']].denominator_as_long()))

```

```
In [ ]:         else:
    gera_traco(declare,init,trans,140,50)

    else:
        print("Não tem solução.")

    plt.plot(time,carro,time,rodas)
    plt.xlabel("Tempo (s)")
    plt.ylabel("Velocidade (m/s)")
    plt.legend(["Carro", "Rodas"], loc ="lower left")
    plt.grid(True)
tm = 0.0
m = FREE
v = 50.0
V = 50.0
Estado: 2
t = 0.1
tm = 0.1
m = FREE
v = 49.0
V = 49.93
Estado: 3
t = 0.1
tm = 0.0
m = STOPPING
v = 49.0
V = 49.93
Estado: 4
t = 0.1
tm = 0.0
m = BLOCKED
v = 49.0
V = 49.93
Estado: 5
t = 0.2
tm = 0.1
m = BLOCKED
v = 48.86
V = 48.86
Estado: 6
t = 0.3
tm = 0.2
m = BLOCKED
v = 47.79
V = 47.79
Estado: 7
t = 0.3
tm = 0.0
m = FREE
v = 47.79
V = 47.79
Estado: 8
t = 0.4
tm = 0.1
m = FREE
v = 46.79
V = 47.72
Estado: 9
```

t = 0.4
tm = 0.0
m = STOPPING
v = 46.79
V = 47.72
Estado: 10
t = 0.4
tm = 0.0
m = BLOCKED
v = 46.79
V = 47.72
Estado: 11
t = 0.5
tm = 0.1
m = BLOCKED
v = 46.65
V = 46.65
Estado: 12
t = 0.6
tm = 0.2
m = BLOCKED
v = 45.58
V = 45.58
Estado: 13
t = 0.6
tm = 0.0
m = FREE
v = 45.58
V = 45.58
Estado: 14
t = 0.7
tm = 0.1
m = FREE
v = 44.58
V = 45.51
Estado: 15
t = 0.7
tm = 0.0
m = STOPPING
v = 44.58
V = 45.51
Estado: 16
t = 0.7
tm = 0.0
m = BLOCKED
v = 44.58
V = 45.51
Estado: 17
t = 0.8
tm = 0.1
m = BLOCKED
v = 44.44
V = 44.44
Estado: 18
t = 0.9
tm = 0.2
m = BLOCKED
v = 43.37
V = 43.37

Estado: 19
t = 0.9
tm = 0.0
m = FREE
v = 43.37
V = 43.37
Estado: 20
t = 1.0
tm = 0.1
m = FREE
v = 42.37
V = 43.3
Estado: 21
t = 1.0
tm = 0.0
m = STOPPING
v = 42.37
V = 43.3
Estado: 22
t = 1.0
tm = 0.0
m = BLOCKED
v = 42.37
V = 43.3
Estado: 23
t = 1.1
tm = 0.1
m = BLOCKED
v = 42.23
V = 42.23
Estado: 24
t = 1.2
tm = 0.2
m = BLOCKED
v = 41.16
V = 41.16
Estado: 25
t = 1.2
tm = 0.0
m = FREE
v = 41.16
V = 41.16
Estado: 26
t = 1.3
tm = 0.1
m = FREE
v = 40.16
V = 41.09
Estado: 27
t = 1.3
tm = 0.0
m = STOPPING
v = 40.16
V = 41.09
Estado: 28
t = 1.3
tm = 0.0
m = BLOCKED
v = 40.16

v = 41.09
Estado: 29
t = 1.4
tm = 0.1
m = BLOCKED
v = 40.02
V = 40.02
Estado: 30
t = 1.5
tm = 0.2
m = BLOCKED
v = 38.95
V = 38.95
Estado: 31
t = 1.5
tm = 0.0
m = FREE
v = 38.95
V = 38.95
Estado: 32
t = 1.6
tm = 0.1
m = FREE
v = 37.95
V = 38.88
Estado: 33
t = 1.6
tm = 0.0
m = STOPPING
v = 37.95
V = 38.88
Estado: 34
t = 1.6
tm = 0.0
m = BLOCKED
v = 37.95
V = 38.88
Estado: 35
t = 1.7
tm = 0.1
m = BLOCKED
v = 37.81
V = 37.81
Estado: 36
t = 1.8
tm = 0.2
m = BLOCKED
v = 36.74
V = 36.74
Estado: 37
t = 1.8
tm = 0.0
m = FREE
v = 36.74
V = 36.74
Estado: 38
t = 1.9
tm = 0.1
m = FREE

v = 35.74
V = 36.67
Estado: 39
t = 1.9
tm = 0.0
m = STOPPING
v = 35.74
V = 36.67
Estado: 40
t = 1.9
tm = 0.0
m = BLOCKED
v = 35.74
V = 36.67
Estado: 41
t = 2.0
tm = 0.1
m = BLOCKED
v = 35.6
V = 35.6
Estado: 42
t = 2.1
tm = 0.2
m = BLOCKED
v = 34.53
V = 34.53
Estado: 43
t = 2.1
tm = 0.0
m = FREE
v = 34.53
V = 34.53
Estado: 44
t = 2.2
tm = 0.1
m = FREE
v = 33.53
V = 34.46
Estado: 45
t = 2.2
tm = 0.0
m = STOPPING
v = 33.53
V = 34.46
Estado: 46
t = 2.2
tm = 0.0
m = BLOCKED
v = 33.53
V = 34.46
Estado: 47
t = 2.3
tm = 0.1
m = BLOCKED
v = 33.39
V = 33.39
Estado: 48
t = 2.4
tm = 0.2

m = BLOCKED
v = 32.32
V = 32.32
Estado: 49
t = 2.4
tm = 0.0
m = FREE
v = 32.32
V = 32.32
Estado: 50
t = 2.5
tm = 0.1
m = FREE
v = 31.32
V = 32.25
Estado: 51
t = 2.5
tm = 0.0
m = STOPPING
v = 31.32
V = 32.25
Estado: 52
t = 2.5
tm = 0.0
m = BLOCKED
v = 31.32
V = 32.25
Estado: 53
t = 2.6
tm = 0.1
m = BLOCKED
v = 31.18
V = 31.18
Estado: 54
t = 2.7
tm = 0.2
m = BLOCKED
v = 30.11
V = 30.11
Estado: 55
t = 2.7
tm = 0.0
m = FREE
v = 30.11
V = 30.11
Estado: 56
t = 2.8
tm = 0.1
m = FREE
v = 29.11
V = 30.04
Estado: 57
t = 2.8
tm = 0.0
m = STOPPING
v = 29.11
V = 30.04
Estado: 58
t = 2.8

```
tm = 0.0
m = BLOCKED
v = 29.11
V = 30.04
Estado: 59
t = 2.9
tm = 0.1
m = BLOCKED
v = 28.97
V = 28.97
Estado: 60
t = 3.0
tm = 0.2
m = BLOCKED
v = 27.9
V = 27.9
Estado: 61
t = 3.0
tm = 0.0
m = FREE
v = 27.9
V = 27.9
Estado: 62
t = 3.1
tm = 0.1
m = FREE
v = 26.9
V = 27.83
Estado: 63
t = 3.1
tm = 0.0
m = STOPPING
v = 26.9
V = 27.83
Estado: 64
t = 3.1
tm = 0.0
m = BLOCKED
v = 26.9
V = 27.83
Estado: 65
t = 3.2
tm = 0.1
m = BLOCKED
v = 26.76
V = 26.76
Estado: 66
t = 3.3
tm = 0.2
m = BLOCKED
v = 25.69
V = 25.69
Estado: 67
t = 3.3
tm = 0.0
m = FREE
v = 25.69
V = 25.69
Estado: 68
```

t = 3.4
tm = 0.1
m = FREE
v = 24.69
V = 25.62
Estado: 69
t = 3.4
tm = 0.0
m = STOPPING
v = 24.69
V = 25.62
Estado: 70
t = 3.4
tm = 0.0
m = BLOCKED
v = 24.69
V = 25.62
Estado: 71
t = 3.5
tm = 0.1
m = BLOCKED
v = 24.55
V = 24.55
Estado: 72
t = 3.6
tm = 0.2
m = BLOCKED
v = 23.48
V = 23.48
Estado: 73
t = 3.6
tm = 0.0
m = FREE
v = 23.48
V = 23.48
Estado: 74
t = 3.7
tm = 0.1
m = FREE
v = 22.48
V = 23.41
Estado: 75
t = 3.7
tm = 0.0
m = STOPPING
v = 22.48
V = 23.41
Estado: 76
t = 3.7
tm = 0.0
m = BLOCKED
v = 22.48
V = 23.41
Estado: 77
t = 3.8
tm = 0.1
m = BLOCKED
v = 22.34
V = 22.34

Estado: 78
t = 3.9
tm = 0.2
m = BLOCKED
v = 21.27
V = 21.27
Estado: 79
t = 3.9
tm = 0.0
m = FREE
v = 21.27
V = 21.27
Estado: 80
t = 4.0
tm = 0.1
m = FREE
v = 20.27
V = 21.2
Estado: 81
t = 4.0
tm = 0.0
m = STOPPING
v = 20.27
V = 21.2
Estado: 82
t = 4.0
tm = 0.0
m = BLOCKED
v = 20.27
V = 21.2
Estado: 83
t = 4.1
tm = 0.1
m = BLOCKED
v = 20.13
V = 20.13
Estado: 84
t = 4.2
tm = 0.2
m = BLOCKED
v = 19.06
V = 19.06
Estado: 85
t = 4.2
tm = 0.0
m = FREE
v = 19.06
V = 19.06
Estado: 86
t = 4.3
tm = 0.1
m = FREE
v = 18.06
V = 18.99
Estado: 87
t = 4.3
tm = 0.0
m = STOPPING
v = 18.06

V = 18.99
Estado: 88
t = 4.3
tm = 0.0
m = BLOCKED
v = 18.06
V = 18.99
Estado: 89
t = 4.4
tm = 0.1
m = BLOCKED
v = 17.92
V = 17.92
Estado: 90
t = 4.5
tm = 0.2
m = BLOCKED
v = 16.85
V = 16.85
Estado: 91
t = 4.5
tm = 0.0
m = FREE
v = 16.85
V = 16.85
Estado: 92
t = 4.6
tm = 0.1
m = FREE
v = 15.85
V = 16.78
Estado: 93
t = 4.6
tm = 0.0
m = STOPPING
v = 15.85
V = 16.78
Estado: 94
t = 4.6
tm = 0.0
m = BLOCKED
v = 15.85
V = 16.78
Estado: 95
t = 4.7
tm = 0.1
m = BLOCKED
v = 15.71
V = 15.71
Estado: 96
t = 4.8
tm = 0.2
m = BLOCKED
v = 14.64
V = 14.64
Estado: 97
t = 4.8
tm = 0.0
m = FREE

v = 14.64
V = 14.64
Estado: 98
t = 4.9
tm = 0.1
m = FREE
v = 13.64
V = 14.57
Estado: 99
t = 4.9
tm = 0.0
m = STOPPING
v = 13.64
V = 14.57
Estado: 100
t = 4.9
tm = 0.0
m = BLOCKED
v = 13.64
V = 14.57
Estado: 101
t = 5.0
tm = 0.1
m = BLOCKED
v = 13.5
V = 13.5
Estado: 102
t = 5.1
tm = 0.2
m = BLOCKED
v = 12.43
V = 12.43
Estado: 103
t = 5.1
tm = 0.0
m = FREE
v = 12.43
V = 12.43
Estado: 104
t = 5.2
tm = 0.1
m = FREE
v = 11.43
V = 12.36
Estado: 105
t = 5.2
tm = 0.0
m = STOPPING
v = 11.43
V = 12.36
Estado: 106
t = 5.2
tm = 0.0
m = BLOCKED
v = 11.43
V = 12.36
Estado: 107
t = 5.3
tm = 0.1

m = BLOCKED
v = 11.29
V = 11.29
Estado: 108
t = 5.4
tm = 0.2
m = BLOCKED
v = 10.22
V = 10.22
Estado: 109
t = 5.4
tm = 0.0
m = FREE
v = 10.22
V = 10.22
Estado: 110
t = 5.5
tm = 0.1
m = FREE
v = 9.22
V = 10.15
Estado: 111
t = 5.5
tm = 0.0
m = STOPPING
v = 9.22
V = 10.15
Estado: 112
t = 5.5
tm = 0.0
m = BLOCKED
v = 9.22
V = 10.15
Estado: 113
t = 5.6
tm = 0.1
m = BLOCKED
v = 9.08
V = 9.08
Estado: 114
t = 5.7
tm = 0.2
m = BLOCKED
v = 8.01
V = 8.01
Estado: 115
t = 5.7
tm = 0.0
m = FREE
v = 8.01
V = 8.01
Estado: 116
t = 5.8
tm = 0.1
m = FREE
v = 7.01
V = 7.94
Estado: 117
t = 5.8

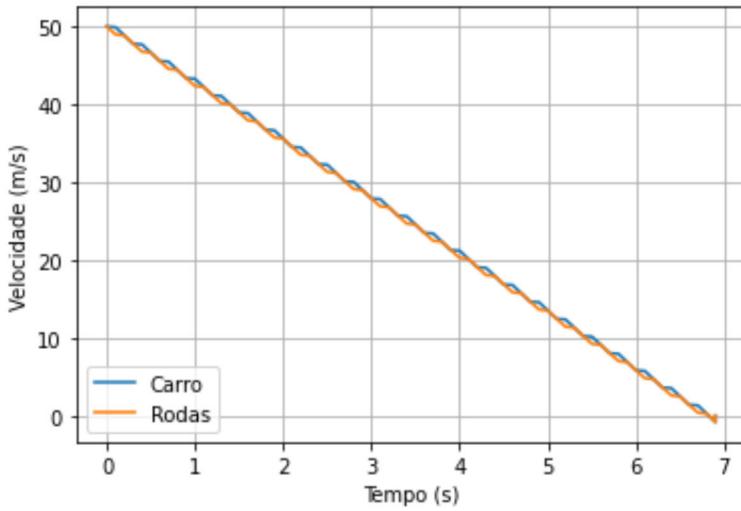
```
tm = 0.0
m = STOPPING
v = 7.01
V = 7.94
Estado: 118
t = 5.8
tm = 0.0
m = BLOCKED
v = 7.01
V = 7.94
Estado: 119
t = 5.9
tm = 0.1
m = BLOCKED
v = 6.87
V = 6.87
Estado: 120
t = 6.0
tm = 0.2
m = BLOCKED
v = 5.8
V = 5.8
Estado: 121
t = 6.0
tm = 0.0
m = FREE
v = 5.8
V = 5.8
Estado: 122
t = 6.1
tm = 0.1
m = FREE
v = 4.8
V = 5.73
Estado: 123
t = 6.1
tm = 0.0
m = STOPPING
v = 4.8
V = 5.73
Estado: 124
t = 6.1
tm = 0.0
m = BLOCKED
v = 4.8
V = 5.73
Estado: 125
t = 6.2
tm = 0.1
m = BLOCKED
v = 4.66
V = 4.66
Estado: 126
t = 6.3
tm = 0.2
m = BLOCKED
v = 3.59
V = 3.59
Estado: 127
```

t = 6.3
tm = 0.0
m = FREE
v = 3.59
V = 3.59
Estado: 128
t = 6.4
tm = 0.1
m = FREE
v = 2.59
V = 3.52
Estado: 129
t = 6.4
tm = 0.0
m = STOPPING
v = 2.59
V = 3.52
Estado: 130
t = 6.4
tm = 0.0
m = BLOCKED
v = 2.59
V = 3.52
Estado: 131
t = 6.5
tm = 0.1
m = BLOCKED
v = 2.45
V = 2.45
Estado: 132
t = 6.6
tm = 0.2
m = BLOCKED
v = 1.38
V = 1.38
Estado: 133
t = 6.6
tm = 0.0
m = FREE
v = 1.38
V = 1.38
Estado: 134
t = 6.7
tm = 0.1
m = FREE
v = 0.38
V = 1.31
Estado: 135
t = 6.7
tm = 0.0
m = STOPPING
v = 0.38
V = 1.31
Estado: 136
t = 6.7
tm = 0.0
m = BLOCKED
v = 0.38
V = 1.31

```

Estado: 137
t = 6.8
tm = 0.1
m = BLOCKED
v = 0.24
V = 0.24
Estado: 138
t = 6.9
tm = 0.2
m = BLOCKED
v = -0.83
V = -0.83
Estado: 139
t = 6.9
tm = 0.2
m = STOPPED

```



b

I. "o veículo imobiliza-se completamente em menos de t segundos"

$$P_1(m = \text{STOPPED} \wedge t < t_i)$$

II. "a velocidade V diminui sempre com o tempo"

$$P_2(t < t' \longrightarrow V > V')$$

```
In [ ]: def prop1(s):
    return And(s['m']==STOPPED, s['t']<8)

def prop2(s,p):
    return Implies(s['t']<p['t'], s['V']>p['V'])
```

d

Verificação das propriedades anteriores utilizando as funções das aulas $bmc_{eventually}$ e bmc_{always}

BMC_EVENTUALLY: dada uma função que gera uma cópia das variáveis do estado, um predicado que testa se um estado é inicial, um predicado que testa se um par de estados é uma transição válida, uma propriedade cuja inevitabilidade se pretende verificar, e um número positivo K, usa o Z3 para encontrar um contra-exemplo para essa propriedade considerando apenas os primeiros K estados de execução do programa.

```
In [ ]: def bmc_eventually(declare,init,trans,prop,K,v):
    for k in range(1,K+1):
        s = Solver()
        traco = [declare(i) for i in range(k)]
        s.add(init(traco[0],v))
        for i in range(k-1):
            s.add(trans(traco[i],traco[i+1]))
        for i in range(k):
            s.add(Not(prop(traco[i])))

    status=s.check()
    if status==sat:
        m=s.model()
        for i in range(k):
            print(i)
            for v in traco[i]:
                print(v,"=",m[traco[i][v]])
        return
    print("A proposição pode ser verdadeira")
```

```
In [ ]: bmc_eventually(declare,init,trans,prop1,140,50)
```

A proposição pode ser verdadeira

BMC_ALWAYS : dada uma função que gera uma cópia das variáveis do estado, um predicado que testa se um estado é inicial, um predicado que testa se um par de estados é uma transição válida, um invariante a verificar, e um número positivo K, usa o Z3 para verificar se esse invariante é sempre válido nos primeiros K-1 passos de execução do programa, ou devolve um contra-exemplo mínimo caso não seja.

```
In [ ]: def bmc_always(declare,init,trans,prop,K,v):
    for k in range(1,K+1):
        s = Solver()
        traco = [declare(i) for i in range(k)]
        s.add(init(traco[0],v))
        for i in range(k-1):
            s.add(trans(traco[i],traco[i+1]))
        if k!=1:
            s.add(Not(prop(traco[k-2],traco[k-1])))
        else:
            s.add(traco[k-1] ['t'] !=0)

    if s.check() ==sat :
        m = s.model()
        print('A propriedade falha')
        for i in range(k):
            print("Estado: ",i)
            for v in traco[i]:
                r = m[traco[i][v]]
                if r.sort() != RealSort():
                    print(v,'=',r)
                else:
                    print(v,'=',float(r.numerator_as_long())/float(r.denominator_as_long()))
        return
    print("A propriedade é válida em traços de tamanho até "+str(K))
```

```
In [ ]: bmc_always(declare,init,trans,prop2,140,50)
```

A propriedade é válida em traços de tamanho até 200