TP 1 EX 2

Grupo 5:

Breno Fernando Guerra Marrão A97768

Tales André Rovaris Machado A96314

Inicialização

Usamos as bibliotecas pysmt e numpy para resovler o problema de alocação proposto

In [1]:
```
from pysmt.shortcuts import *
from pysmt.typing import BOOL
import numpy
```

Implementação

A seguinte função cria a variável x de acordo com a probabilidade p da borda e o tamanho n

In [2]:
```
def declare(n):
    x = {}
    for i in range(n):
        for j in range(n):
            x[i,j] = Symbol('x'+str(i)+','+str(j),INT)
    return x
```

In [3]:
```
def init(x,n,p,s):

    for i in range(0,n):
         s.add_assertion(Equals(x[i,0] , numpy.random.choice([Int(0),Int(1)], p=[p,1-p])))

    for j in range(1,n):
        s.add_assertion(Equals(x[0,j] , numpy.random.choice([Int(0),Int(1)], p=[p,1-p])))
    c = []
    centro = (numpy.random.choice(numpy.arange(2, n-1)),numpy.random.choice(numpy.arange(2, n-1)))
    c.append(centro)
    #print(centro)
    for i in range(-1,2):
        for j in range(-1,2):

            a = centro[0]+i
            b = centro[1]+j
            c.append((a,b))

            s.add_assertion(Equals(x[a,b], Int(1)))

    for i in range(1,n):
        for j in range(1,n):
            if (i,j) not in c:
                s.add_assertion(Equals(x[i,j] , Int(0)))
    return x
```

Função que retorna o número de vizinhos vivos em uma certa posição (a,b) na matriz curr de tamanho n.

$$\sum_{i\in[-1,0,1]j\in[-1,0,1]} curr_{i,j}$$

In [4]:
```
def checavizinhos(curr,a,b,n):
    if a == n-1:
        arr1 = [-1,0]
    else:
        arr1 = [-1,0,1]
    if b == n-1:
        arr2 = [-1,0]
    else:
        arr2 = [-1,0,1]

    aux = []
    for i in arr1:
        for j in arr2:
            if not(j==0 and i==0):
                #print(i,j)
                aux.append(curr[i+a,j+b])
    return sum(aux)
```

$where \quad c_{i,j} = checavizinhos(curr, i, j, n)$

In [5]:
```
def vive(c):
    x = Ite(Or(Equals(c,Int(2)),Equals(c,Int(3))),Int(1),Int(0))

    return x
def morre(c):
    x = Ite(Equals(c,Int(3)),Int(1),Int(0))

    return x
```

Função de transição da matriz curr para uma nova matriz fazendo a transição de cada celula .

$\forall i, j < n \quad if \quad curr_{i,j} = 1 \quad then \quad currnew_{i,j} = vive(checavizinhos(curr_{i,j})) \quad else \quad currnew_{i,j} = morre(checavizinhos(curri, j))$

In [6]:
```
def trans(curr,currnew,n,s):

    for i in range(n):
        s.add_assertion(Equals(currnew[i,0] , curr[i,0]))
    for j in range(n):
        s.add_assertion(Equals(currnew[0,j] , curr[0,j]))
    for i in range(1,n):
        for j in range(1,n):
            c = checavizinhos(curr,i,j,n)


            currnew[i,j] = Ite(Equals(curr[i,j],Int(1)),vive(c),morre(c))
```

Função que roda a transição k vezes

In [7]:
```
def roda(n,p,k,trans):
    with Solver(name="z3") as s:

        st = []

        for x in range (k):
            st.append(declare(n))

        init(st[0],n,p,s)

        for i in range(k-1):
            trans(st[i],st[i+1],n,s)

        if s.solve():

            for h in range(k):
                for i in range(n):
                    for j in range(n):
                        if s.get_value(st[h][i,j]) == Int(1):
                            print("X",end="|")
                        else:
                            print(".",end="|")
                    print()
                print("-------------")
```

Invariantes

Todos os estados acessíveis contém pelo menos uma célula viva

$$\sum_{1<a<n, 1<b<n} x_{a,b} \geq 1$$

Toda a célula normal está viva pelo menos uma vez em algum estado acessível

$$\forall_{1<a<n} \cdot \forall_{1<b<n} \quad \sum_{x \in lista} x_{a,b} \geq 1$$

In [8]:
```
def pelomenosumaviva(x,n):

    return GE(sum(x[i,j] for i in range(1,n) for j in range(1,n)),Int(1))
```

Toda a célula normal está viva pelo menos uma vez em algum estado acessível

$$\forall_{1<a<n} \cdot \forall_{1<b<n} \quad \sum_{x \in lista} x_{a,b} \geq 1$$

In [9]:
```
def vivaumavez(lista,n,a,b):
    return GE(sum(x[a,b] for x in lista),Int(1))
```

Funções que testam os invariantes

In [10]:

```python
def testinva(n,p,k,trans,inv):

    with Solver(name="z3") as s:


        st = []

        for x in range (k):
            st.append(declare(n))

        init(st[0],n,p,s)

        for i in range(k-1):
            trans(st[i],st[i+1],n,s)
            s.add_assertion(Not(inv(st[i+1],n)))

        if s.solve():
            print("invariante a não é valido")
            return
        print("invariante a valido")

def testinvb(n,p,k,trans,inv):

    with Solver(name="z3") as s:

        st = []

        for x in range (k):
            st.append(declare(n))

        init(st[0],n,p,s)

        for i in range(k-1):
            trans(st[i],st[i+1],n,s)



        s.add_assertion(Not(And([inv(st,n,a,b) for a in range(n) for b in range(n)])))

        if s.solve():
            print("invariante b não é valido")
            return
        print("invariante b valido")

testinva(10,0.5,10,trans,pelomenosumaviva)
testinvb(10,0.5,10,trans,vivaumavez)
```

```
invariante a valido
invariante b não é valido
```

Testes

podemos observar neste caso que em 20 passos diferentes multiplas celulas tem celulas mortas sempre, mostrando que o invariante b não é valido

In [11]:

```python
roda(20,1,20,trans)
```

```
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|X|X|X|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|X|X|X|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|X|X|X|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
```

In [12]:
```python
roda(15,0.5,10,trans)
```

```
.|X|X|.|X|.|X|.|.|.|.|.|X|X|X|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|X|X|X|.|.|.|.|.|.|.|.|.|
X|.|.|X|X|X|.|.|.|.|.|.|.|.|.|
X|.|.|X|X|X|.|.|.|.|.|.|.|.|.|
-------------
.|X|X|.|X|.|X|.|.|.|.|.|X|X|X|
.|.|.|.|.|.|.|.|.|.|.|.|.|X|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
```

In [13]:
```python
roda(15,0,10,trans)
```

```
X|X|X|X|X|X|X|X|X|X|X|X|X|X|X|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|X|X|X|.|.|.|.|.|
X|.|.|.|.|.|.|X|X|X|.|.|.|.|.|
X|.|.|.|.|.|.|X|X|X|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
-------------
X|X|X|X|X|X|X|X|X|X|X|X|X|X|X|
X|.|X|X|X|X|X|X|X|X|X|X|X|X|.|
X|X|.|.|.|.|.|.|.|.|.|.|.|.|.|
```

In [14]:
```python
roda(100,0.225,10,trans)
```

```
X|X|X|.|X|X|.|X|X|X|.|X|.|.|.|X|.|X|X|X|X|X|X|X|.|X|X|X|X|X|.|X|X|.|X|X|X|X|X|X|X|X|X|X|X|.|.|X|X|.|X|X|X|X|.|X|.|
X|.|.|X|X|X|X|X|.|.|X|X|X|.|X|X|X|.|.|.|X|X|X|X|X|X|.|X|X|X|X|X|X|X|X|X|X|X|.|X|X|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
X|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|.|
```

In [ ]:
```python

```