MAE0221 PROBABILIDADE I

Nome: Ygor Sad Machado **Número USP: 8910368**

Lista 7 **Data: 27/05/2020**

Considere inicialmente o seguinte código de *setup*. Note que ele possui a definição de algumas funções que serão utilizadas em todas as simulações a seguir.

```python
from random import random, seed

# Fixa a semente do gerador de números aleatórios para manter
# a consistência nos resultados da simulação
seed(1)

# Calcula a distância Manhattan entre dois pontos p1 e p2
def manhattan_distance(p1, p2):
    return sum(abs(a-b) for a, b in zip(p1, p2))

# Gera um ponto aleatório num grid quadrado de lado L
def rand_coord(l):
    return [random() * l, random() * l]

# Realiza o experimento um determinado número de vezes para
# um valor de L e um número de ambulâncias fixados
def experiment(num, l, ambulances):
    sum_distance = 0

    for i in range(int(num)):
        accident = rand_coord(l)

        # Gera uma lista contendo o número de ambulâncias recebido
        coord_ambulances = [rand_coord(l) for j in range(ambulances)]

        # Soma somente a menor distância entre alguma ambulância e
        # o local do acidente
        sum_distance += min(
            [manhattan_distance(accident, amb) for amb in coord_ambulances]
        )

    return sum_distance / num
```

Item a.

Neste item estamos interessados em descobrir a esperança para a distância entre dois pontos usando $L = 1$ como parâmetro. Para isso foi usado o seguinte código:

```python
def item_a():
    print("A. ")

    for n in [1e5, 1e6, 1e7]:
        expectation = experiment(num=n, l=1, ambulances=1)
        print("{it:.1e} iterations => {e}".format(it=n, e=expectation))

    print("\n")
```

A execução dessa função gera como saída:

```
1.0e+05 iterations => 0.666830572178
1.0e+06 iterations => 0.666817365745
1.0e+07 iterations => 0.666774043139
```

Isso sugere uma convergência no valor da esperança, de modo que $\mathbb{E} \approx 0.66666...$ parece ser uma boa aproximação para ela.

Item b.

Já neste item, estamos interessados em descobrir uma relação entre o valor L e a esperança $\mathbb{E}$ para a distância, estando o número de ambulâncias fixado em $n = 1$. Para isso usamos o seguinte código:

```python
def item_b():
    print("B. ")

    for length in range(2,8):
        expectation = experiment(num=1e6, l=length, ambulances=1)
        print("L={l} => {e}".format(l=length, e=expectation))

    print("\n")
```

Cuja saída é:

```
L=2 => 1.333307524
L=3 => 2.00006665411
L=4 => 2.66444358418
L=5 => 3.33203882442
L=6 => 3.99962416282
L=7 => 4.6727126035
```

Note que todos os valores que obtivemos correspondem a dois terços do valor de seus respectivos L, o que nos sugere que a esperança pode ser aproximada por $\mathbb{E} = \frac{2L}{3}$.

Item c.

Já para estas simulações, vamos fixar o número de ambulâncias em $n = 2$ e executar o código abaixo.

```python
def item_c():
    print("C. ")

    for n in [1e5, 1e6, 1e7]:
        expectation = experiment(num=1e6, l=1, ambulances=2)
        print("{it:.1e} iterations => {e}".format(it=n, e=expectation))

    print("\n")
```

Para ele obtivemos a seguinte saída:

```
1.0e+05 iterations => 0.487881704132
1.0e+06 iterations => 0.487956896953
1.0e+07 iterations => 0.487897822258
```

Essa saída nos sugere que um boa aproximação para a esperança nessas condições é $\mathbb{E} \approx 0.488$.

Item d.

Por fim, neste item queremos descobrir o menor número de ambulâncias necessárias para que a esperança da distância entre o local do acidente e a ambulância mais próxima seja $\frac{L}{3}$. Para isso fixamos um valor inicial arbitrariamente alto para $\mathbb{E}$ e vamos incrementando o número de ambulâncias até que $\mathbb{E} \leq \frac{L}{3}$. Além disso, variamos L e o número de experimentos de modo a ter um panorama para diferentes tamanhos de *grid*. O código abaixo foi usado nessa simulação.

```python
def item_d():
    print("D.")

    for n in [1e5, 1e6, 1e7]:
        print("{it:.1e} iterations".format(it=n))

        for length in range(1,5):
            num_ambulances = 0
            expectation = 1e20

            while (expectation >= length / 3.0):
                num_ambulances += 1
                expectation = experiment(num=n, l=length, ambulances=num_ambulances)

            print(
                "L={l} | expectation={e} => {amb} ambulances".format(
                    l=length, e=expectation, amb=num_ambulances
                )
            )
        print("\n")
```

A execução de dessa função tem como saída:

```
1.0e+05 iterations
L=1 | expectation=0.31074490733 => 5 ambulances
L=2 | expectation=0.622679090939 => 5 ambulances
L=3 | expectation=0.934401355886 => 5 ambulances
L=4 | expectation=1.24493205269 => 5 ambulances

1.0e+06 iterations
L=1 | expectation=0.310965785717 => 5 ambulances
L=2 | expectation=0.621694269052 => 5 ambulances
L=3 | expectation=0.932674487682 => 5 ambulances
L=4 | expectation=1.24288063528 => 5 ambulances

1.0e+07 iterations
L=1 | expectation=0.311061783846 => 5 ambulances
L=2 | expectation=0.621796002463 => 5 ambulances
L=3 | expectation=0.932712301752 => 5 ambulances
L=4 | expectation=1.24334908597 => 5 ambulances
```

Observe que, independentemente do número de experimentos realizados ou do tamanho de L, o resultado dos experimentos aponta sempre para 5 ambulâncias. Parece razoável então afirmar que o número mínimo de ambulâncias é constante e sempre igual a 5.