Instituto Federal Catarinense - Campus Videira.
Ciência da Computação.
Turma: 2020.
Inteligência Artificial.
Docente: Prof. Dr. Manassés Ribeiro
Discente: Gabriel Solanha.

Roteiro do algoritmo genético:

- Identifique a **natureza do problema** de otimização;

É um problema de maximização (do lucro final) e tem quatro critérios para serem otimizados.

- Identifique o **conjunto de variáveis** do problema;

As variáveis são:

- carga 1 dianteira, carga 1 central, carga 1 traseira.
- carga 2 dianteira, carga 2 central, carga 2 traseira.
- carga 3 dianteira, carga 3 central, carga 3 traseira.
- carga 4 dianteira, carga 4 central, carga 4 traseira.

Totalizando 12 variáveis ou seja 12 genes.
cada uma recebe um nome abreviado conforme a abaixo listado:

```
def perform_fitness(p):
    c1_d = float(p[0])  # carga 1 compartimento diânteiro
    c1_c = float(p[1])  # carga 1 compartimento central
    c1_t = float(p[2])  # carga 1 compartimento traseio
    c2_d = float(p[3])  # carga 2 compartimento diânteiro
    c2_c = float(p[4])  # carga 2 compartimento central
    c2_t = float(p[5])  # carga 2 compartimento traseio
    c3_d = float(p[6])  # carga 3 compartimento diânteiro
    c3_c = float(p[7])  # carga 3 compartimento central
    c3_t = float(p[8])  # carga 3 compartimento traseio
    c4_d = float(p[9])  # carga 4 compartimento diânteiro
    c4_c = float(p[10])  # carga 4 compartimento central
    c4_t = float(p[11])  # carga 4 compartimento traseio
```

- Desenvolva uma **codificação apropriada** para o conjunto de variáveis;

Para gerar a primeira codificação deve-se limitar cada variável pelo máximo de tipo de carga disponível indo até o máximo de 16 toneladas que é o máximo do maior compartimento

- Identifique e **formule matematicamente as restrições** do problema;

São quatro grandes grupos de restrições sendo elas:

- Restrições peso do compartimento:

Essa primeira se refere ao peso máximo de cada compartimento, sendo assim para cada tipo de carga de um compartimento x eu somo esses todos tipos de carga diminuída do meu maximo e equalizo.

ficando conforme a figura abaixo:

```
# restrições peso do compartimento
h[1] = np.maximum(0, float((c1_d+c2_d+c3_d+c4_d)-10000)
                   ) / 10000  # compartimento 1
h[2] = np.maximum(0, float((c1_c+c2_c+c3_c+c4_c)-16000)
                   ) / 16000  # compartimento 2
h[3] = np.maximum(0, float((c1_t+c2_t+c3_t+c4_t)-8000)
                   ) / 8000  # compartimento 3
```

- Restrição volumétrica:

Análogo ao de cima porém o cálculo do volume é feito com o cálculo do volume de acordo com o volume máximo de cada compartimento é feito o quanto está sendo ocupado pelas cargas dentro dele segue as fórmulas feitas:

```
# restrição volumétrica
# volume comp 1
h[4] = np.maximum(
    0, float((.48*c1_d+.65*c2_d+.58*c3_d+.39*c4_d)-6800))/6800
# volume comp 2
h[5] = np.maximum(
    0, float((.48*c1_c+.65*c2_c+.58*c3_c+.39*c4_c)-8700))/8700
# volume comp 3
h[6] = np.maximum(
    0, float((.48*c1_t+.65*c2_t+.58*c3_t+.39*c4_t)-5300))/5300
```

- Restrição peso da carga:

Para cada peso de carga temos um máximo de peso que temos disponível para carregar mais do que isso o tipo de carga não terá mais, logo deve-se somar quanto tem de cada tipo de carga sendo usado e se diminuir da quantia disponível depois fazer-se uma equalização.

As restrições ficam as seguintes:

```
# restrião peso da carga
#carga 1
h[7] = np.maximum(0, float((c1_d+c1_c+c1_t)-18000)
                   ) / 18000
#carga 2
h[8] = np.maximum(0, float((c2_d+c2_c+c2_t)-15000)
                   ) / 15000
#carga 3
h[9] = np.maximum(0, float((c3_d+c3_c+c3_t)-23000)
                   ) / 23000
#carga 4
h[0] = np.maximum(0, float((c4_d+c4_c+c4_t)-12000)
                   ) / 12000
```

- Restrição de proporção:

Como a distribuição vai somente de 0 a 1 não precisa-se equaliza-la somente verificar a proporção e fazer menos o valor correto dela e pegar o seu módulo, para fazer a proporção basta somar o peso total das cargas no compartimento e dividir pela soma total de todas as

cargas no avião que fica assim:

```
#restrição de proporção
h[10] = abs(float((c1_c+ c2_c + c3_c + c4_c)/sum(p))-0.470588)
h[11] = abs(float((c1_d+ c2_d + c3_d + c4_d)/sum(p))-0.294118)
h[12] = abs(float((c1_t+ c2_t + c3_t + c4_t)/sum(p))-0.235294)
```

- Identifique a(s) **função(s) objetivo(s)**;

A função objetivo é a função lucro que é calculado através da soma do lucro de cada uma das cargas somadas em seus tipos de cargas específicos tudo isso dividido por 12350 que é um valor teórico inalcançável que remete a um caso onde conseguimos usar ao máximo as cargas mais caras em todo o avião. A fórmula segue conforme ilustrado pelo código abaixo:

```
fit = float(0.310*(c1_d+c1_c+c1_t) + 0.380*(c2_d+c2_c+c2_t) +
            0.350*(c3_d+c3_c+c3_t) + 0.285*(c4_d+c4_c+c4_t)) / 12350
```

- Desenvolva uma **função de fitness adequada**;

A função de fitness adequada é sempre a função fit menos as restrições no nosso caso as restrições são arrays, logo ficou assim:

```
fit = float(fit - sum(h))
```