Money Bills Problem

# Task:

Utilizați un algoritm genetic pentru a rezolva problema: pentru plata unei sume S se pot folosi bancnote cu n valori nominale distincte (v[i], i=1,…,n). NU este disponibila bancnota cu valoare 1. Pentru fiecare valoare nominala este disponibila o cantitate limitata de bancnote (c[i], i=1,…,n). Găsiți o modalitate de plata care folosește numărul minim de bancnote. Este utilizat un algoritm genetic.

# Individual representation:

We consider an individual. For any we have where represents the amount of bills of type is used to pay the sum . We cannot pay less than 0 banknotes, thus we have the constraint .

Example:

we pay the sum using 2 banknotes of value 2, 7 of value 3, 1 of value 5, none of value 7, 2 of value 10, none of value 11 and none of value 13.

# Fitness function:

Our goal is to pay the sum using the minimum amount of bills. If we consider ,

and we wish to minimize this cost, then our fitness function can be defined as such:

We consider instead of just to avoid division by zero in the case of .

# Restrictions / feasibility check:

We must pay the whole sum , which means that the value of must be equal to . If the condition is not met, the individual is non-feasible. Otherwise, the individual is feasible.

Initial population generation:

Each individual in the population is randomly generated, by assigning a random value between 0 and , value that represents the chosen amount of banknotes of type .

while wasAccepted == False:  
 # we generate element by element in the individual  
 x\_fitness = 0  
 for j in range(parameters.n):  
 current\_value = np.random.randint(0, parameters.max\_bill\_amounts[j] + 1)  
 x += [current\_value

# after we have an individual, we check if it is feasible

if isFeasible(x) == True:  
 wasAccepted = True  
 pop += [x] # we add x to the population  
 x\_fitness = fitness\_func(x)  
 init\_pop\_fitness += [x\_fitness]  
 else:  
 x = []

After the generation, we check if the individual is feasible. If not, we try again. When we find a feasible individual, we add it to the initial population as well as its fitness value to our list of initial fitness values.

This mechanism will be repeated times where .