# Ai genetic Algorithms

الاسم:أحمد سمير عبد الفتاح امين

Our problem is:

Suppose a genetic algorithm uses chromosomes of the form x = a b c d e f g h with a fixed length of eight genes. Each gene can be any digit between 0 and 9. Let the fitness of individual x be calculated as:

**f(x) = (a + b) − (c + d) + (e + f) − (g + h)**

and let the initial population consist of four individuals with the following chromosomes: **x1 = 6 5 4 1 3 5 3 2**

**x2 = 8 7 1 2 6 6 0 1**

**x3 = 2 3 9 2 1 2 8 5**

**x4 = 4 1 8 5 2 0 9 4**

how we going solve this problem:

Initial Population

work

Fitness function

Select by Rating

mutation

New Population

Best solution

n != 100

n = 100

1)Initial Population

**x1 = 6 5 4 1 3 5 3 2**

**x2 = 8 7 1 2 6 6 0 1**

**x3 = 2 3 9 2 1 2 8 5**

**x4 = 4 1 8 5 2 0 9 4**

2)work function

def work(pop,n):

#put all genes we have in dec where keys = values from fitness function and values = the gene

    for i in pop:

        dec[fitness(i)]=i

3)fitness function

#fitness function to evaluate fitness for each gene

def fitness(gene):

    x=(gene[0]+gene[1])-(gene[2]+gene[3])+(gene[4]+gene[5])-(gene[6]+gene[7])

    return x

4)select by Rating and take best two

#choose based of order the out put and choose best two gene from them

    best=sorted(dec.keys(),reverse=True)

    #create new gene based on mutation

    gene1=dec[best[0]]

    gene2=dec[best[1]]

5)send data to mutation function

gene3=mutation(dec.pop(best[0]))

gene4=mutation(dec.pop(best[1]))

6)mutation function

#create mutation operator on best\_1 gene and best\_best\_2 gene

def mutation(best\_1):

    li=best\_1.copy()

    li[random.randint(0,7)]=random.randint(0,9)

    return li

7)new population

newpop=[gene1,gene2,gene3,gene4]

    print(newpop)

    work(newpop,n+1)

8)check the number of iteration

if n==100:

        print("the best solution after 100 iteration is :",pop[0])

        return