Potenciar cluster de procesadores y temperatura

3 procesadores, la function de la potencia es Pt = p0, p1, p2

Cuando los 3 en reposo la temperatura es 35,

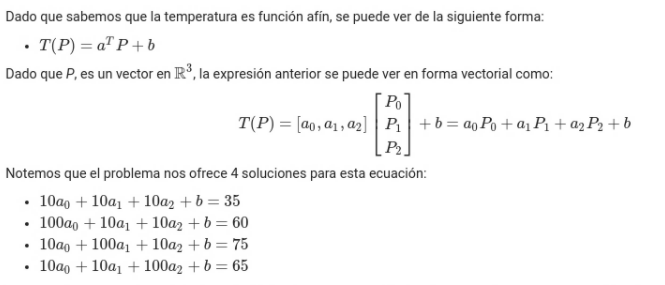
P = [10,10,10] = 35 es decir, la function dice, que cada uno con potencia de 10 dara temperature de 35

P=[100,10,10] = 60 si el primer procesador funcionando tendra potencia de 100 y generara 60 de temperature

P=[10,100,10] = 75

P=[10,10,100] = 65

Supongamos que los 3 procesadores a maxima potencia [100,100,100] que temperature alcanzara? No debe pasar de 150 o se deriten, cual es la potencia maxima que pueden alcanzar los procesadores sin pasar de 150



import numpy as np

A = np.array([

[10,10,10,1] ,

[100,10,10,1],

[10,100,10,1],

[10,10,100,1] ])

b = np.array([35,60,75,65])

a = np.linalg.solve(A,b)

b= a[3]

a = a[:3]

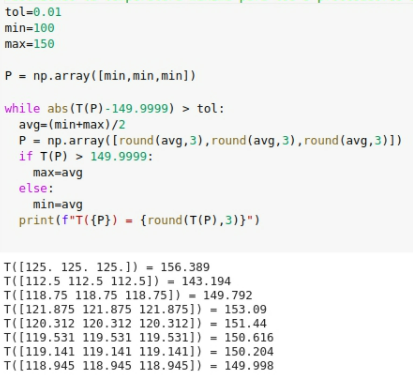
print("a^t ",a)

print(b)

a^t = [0.2777, 0.4444, 0.3333]

b = 24.4444





Otra opcion

**import** numpy **as** np

**from** sklearn.linear\_model **import** LinearRegression

**def** **datos\_experimentales**():

p\_0=np.array([10,10,10]) #en reposo

p\_1=np.array([100,10,10])

p\_2=np.array([10,100,10])

p\_3=np.array([10,10,100])

p=np.array([p\_0,p\_1,p\_2,p\_3])

t=np.array([35,60,75,65])

**return** p, t

**def** **solucion\_metodo1**(): #resolvinedo el sistema de ecuaciones

ecuacion1="10\*a0+10\*a1+10\*a2+b=35"

ecuacion2="100\*a0+10\*a1+10\*a2+b=60"

ecuacion3="10\*a0+100\*a1+10\*a2+b=70"

ecuacion4="10\*a0+10\*a1+100\*a2+b=65"

p=np.array([[10,10,10,1],[100,10,10,1],[10,100,10,1],[10,10,100,1]]) #matrix de coeficientes, potencias de los controladores

t=np.array([35,60,70,65]) #matriz de soluciones, temperaturas

a=np.linalg.solve(p,t) #obtengo los valores de a y b

**return** a

**def** **solucion\_metodo2**(): #usando minimos cuadrados, numpy.linalg.lstsq

# Experimental data:

p,t=datos\_experimentales()

A = np.column\_stack([p, np.ones(len(p))])

coefs = np.linalg.lstsq(A, t, rcond=-1)[0]

**return** coefs

**def** **solucion\_metodo3**(): #regresion lineal con sklearn.linear\_model

model = LinearRegression()

# Experimental data:

p,t=datos\_experimentales()

model.fit(X=p, y=t)

print("coeff: ", \*model.coef\_)

print("intercept: ", model.intercept\_)

**def** **run**():

print("\*\*\*\*\*\*\*\*metodo1, resolvinedo el sistema de ecuaciones\*\*\*\*\*\*\*")

print(solucion\_metodo1())

print("\*\*\*\*\*\*\*\*metodo2, usando minimos cuadrados, numpy.linalg.lstsq\*\*\*\*\*\*\*")

print(solucion\_metodo2())

print("\*\*\*\*\*\*\*\*metodo3, regresion lineal con sklearn.linear\_model\*\*\*\*\*\*\*")

solucion\_metodo3()

**if** \_\_name\_\_ == "\_\_main\_\_":

run()

Output:

Text

Description automatically generated