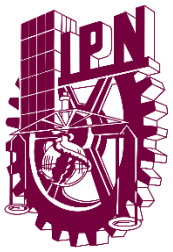


INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE COMPUTO

PRACTICA 1

APLICACIONES PARA COMUNICACIONES EN RED

GRUPO 3CV17

INTEGRANTES:

- MORA GUZMAN JOSE ANTONIO
- MIRANDA QUIJANO MANUEL ALEJANDRO
- MURGUIA JIMENEZ DANIEL

Introducción

Un hilo es la menor de las estructuras lógicas de programación que se ejecuta de forma secuencial por parte del planificador del sistema operativo

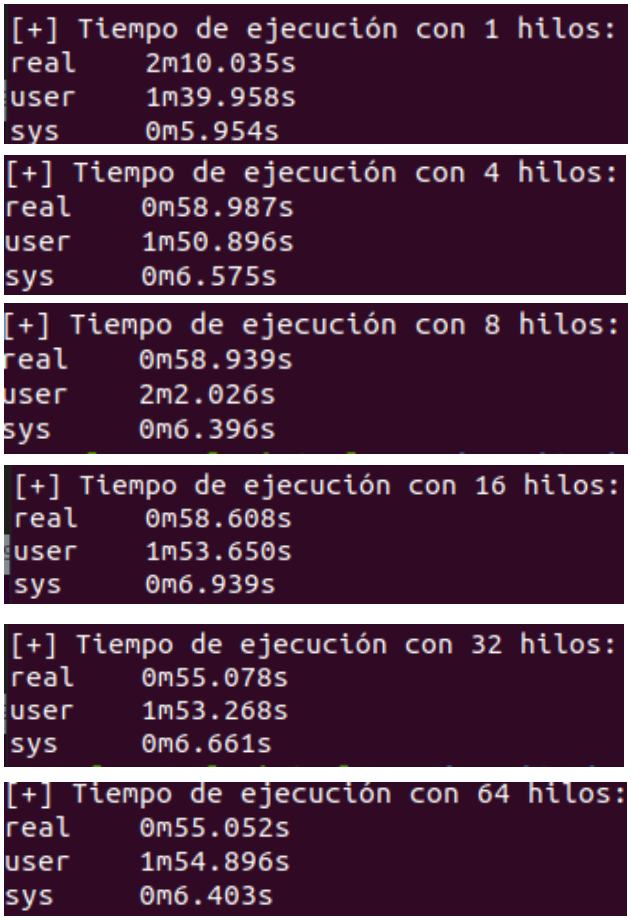
Los hilos son más “ligeros” que sus hermanos mayores (los procesos) ya que muchos de los recursos que es necesario reservar para lanzar y ejecutar un proceso, son compartidos entre distintos hilos dentro de un mismo proceso.

Para esta práctica se estará trabajando con hilos para resolver dos problemáticas las cuales son:

A. Realizar un conjunto de pruebas al programa de multiplicación de matrices, los tamaños de las matrices serán de 2000 x 2000 en ambas matrices y se utilizarán primeramente 1,4,8,16,32,64 hilos. En cada prueba realizada se deberá medir el tiempo de ejecución y al finalizar todas las pruebas se deberán graficar los tiempos de ejecución vs número de hilos. Describir el comportamiento de la gráfica.
B. Realizar un programa con hilos, el cual se encargue de leer archivos de una carpeta, cada archivo será trabajado por un hilo, este hilo se deberá de encargar de contabilizar la aparición de las siguientes palabras: casa, jardín, pelota, juego, amor, enojo. Al finalizar el conteo total de palabras el hilo debe reportar la cantidad de aparición de cada una de ellas en el texto y deberá generar un reporte del porcentaje de aparición (Total de palabras/ Aparición de cada palabra). Esta información se deberá enviar de retorno hacia el proceso padre, el cual al final mostrará en porcentaje de aparición de palabras en todos los archivos utilizados.

Desarrollo

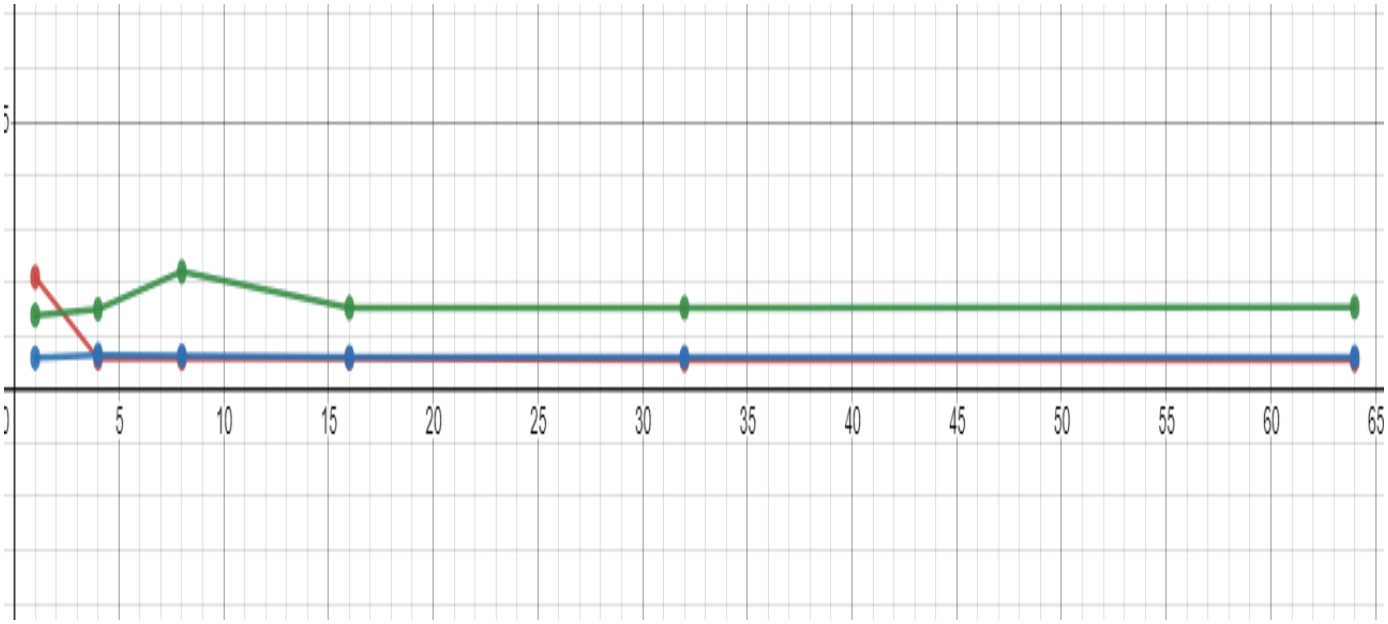
Para el inciso A) se trabajó sobre el programa de multiplicación de matrices en C el cual se desarrolló durante clases, a dicho programa solo se le agrego un printf el cual se usó para marcar el tiempo de ejecución y con cuantos hilos se estaba trabajando, este programa se ejecutó con " time ./cal" donde cal es el nombre del programa y nos devolvió los siguientes resultados:



Después de eso se procedió a acomodar los valores en una tabla la cual quedo de la siguiente manera:

Práctica 1: A			
Número de hilos	Tiempos (min)		
	real	usr	sys
1	2.10	1.39	0.59
4	0.58	1.50	0.65
8	0.58	2.2	0.63
16	0.58	1.53	0.6
32	0.55	1.53	0.6
64	0.55	1.54	0.6

Graficando la tabla anterior con el numero de hilos en el eje de las x y el tiempo en el eje de las y pudimos observar lo siguiente:



El color rojo indica la gráfica correspondiente al número de hilos en relación con el tiempo real.

El color verde indica la gráfica correspondiente al número de hilos en relación con el tiempo de usuario

El color verde indica la gráfica correspondiente al número de hilos en relación con el tiempo del sistema.

Como podemos observar para el sistema el usar mas de un hilo hace que el tiempo aumente un poco y se mantenga un tiempo constante.

Para el usuario usar de 1 a 8 hilos hace que el tiempo aumente considerablemente, aunque después de eso el tiempo baja y se mantiene.

Finalmente, el tiempo real nos muestra que usar más de un hilo hace que el tiempo disminuya mucho y se mantenga sin importar el numero de hilos a usar,

Para el inciso B) de la practica se trabajo en el lenguaje de programación Python debido a que para el equipo nos fue más sencillo el manejar el problema por medio de este lenguaje de programación, por fines prácticos a continuación se muestra el código desarrollado para resolver la problemática con los debidos comentarios explicando las partes importantes del programa:

Código inciso B

```
1. import os
2. import threading as th
3.
4.
5. def worker(ruta, nombre, n_palabras, d_conteo, d_frecuencias):
6.     ##Se abre el archivo correspondiente al hilo
7.     f = open(ruta, encoding='utf-8')
8.     texto = f.read()
9.     f.close()
10.    texto = texto.split()
11.    #Se obtiene el total de palabras del archivo
12.    total_palabras = len(texto)
13.    #se agrega el numero total de palabras del archivo correspondiente a
14.    ##una lista del hilo padre
15.    n_palabras.append(total_palabras)
16.    ##Se obtiene el vocabulario de palabras del archivo
17.    vocabulario = sorted(list(set(texto)))
18.    conteos = list()
19.    frecuencias = list()
20.    #Se itera sobre el archivo obteniendo el conteo y frecuencias de
   palabras
21.    for palabra in vocabulario:
22.        conteo = texto.count(palabra)
23.        frecuencia = conteo/total_palabras
24.        conteos.append((palabra, conteo))
25.        frecuencias.append((palabra, frecuencia))
26.    ##Se agregan los conteos y frecuencias de cada palabra a los
   diccionarios
27.    ##correspondientes del hilo padre
28.    d_conteo[nombre] = conteos
29.    d_frecuencias[nombre] = frecuencias
30.
31.
32. if __name__ == '__main__':
33.    ruta = "./Archivos"
34.    contenido = os.listdir(ruta)
35.    hilos = list()
36.    conteos_totales = list()
37.    conteos_indi = dict()
38.    frecuencias_indi = dict()
39.
40.    ##Se crean el mismo numero de hilos que archivos en la carpeta
41.    for archivo in contenido:
```

```
42.         t = th.Thread(target=worker, args=(ruta+'/'+archivo, archivo,
43.                                            conteos_totales, conteos_indi,
44.                                            frecuencias_indi,))
45.         hilos.append(t)
46.         t.start()
47.         t.join()
48.
49.     cuenta_total = 0
50.     ##Se obtiene el numero total de palabras de todos los archivos
51.     for cuenta in conteos_totales:
52.         cuenta_total += cuenta
53.     ##Se obtienen el conteo total de cada palabra en todos los archivos
54.     ##Y la frecuencia total de cada palabra en todos los archivos y se
55.     ##almacenan los valores en los diccionarios correspondientes
56.     conteo_vocabulario = dict(zip(['amor','casa','enojo','jardin','juego',
57.                                    'pelota'],[0,0,0,0,0,0]))
58.     frecuencias_vocabulario = dict()
59.     for key in conteos_indi:
60.         lista = conteos_indi.get(key)
61.         for tupla in lista:
62.             conteo_vocabulario[tupla[0]] += tupla[1]
63.     for key in conteo_vocabulario:
64.         frecuencias_vocabulario[key] = conteo_vocabulario[key]/cuenta_total
65.
66.     ##Se imprimen los datos que calculo cada hilo al ejecutarse
67.     print("\nConteo de palabras de cada archivo:")
68.     for key in conteos_indi:
69.         lista = conteos_indi.get(key)
70.         print("\nArchivo: "+key)
71.         for tupla in lista:
72.             print(tupla[0]+": %d"%tupla[1])
73.
74.     print("\nFrecuencia de palabras de cada archivo:")
75.     for key in frecuencias_indi:
76.         lista = frecuencias_indi.get(key)
77.         print("\nArchivo: "+key)
78.         for tupla in lista:
79.             print(tupla[0]+": %f"%tupla[1])
80.
81.     ##Se imprimen los datos de los valores totales calculados por el hilo
   padre
82.     print("\nConteo de palabras total en los archivos:")
83.     for key in conteo_vocabulario:
84.         print("\n"+key+": %d" %conteo_vocabulario[key])
85.
86.     print("\nFrecuencias de palabras en todos los archivos:")
87.     for key in frecuencias_vocabulario:
88.         print("\n"+key+": %f" %frecuencias_vocabulario[key])
```

Al ejecutar el programa mostrado anteriormente nos arrojó los siguientes resultados:

```
In [3]: runfile('K:/Usuarios/dan_1/Documents/ESCOM/
AplicacionesDeRedesParaLaComunicacion/Practica1/Lectura de archivos/
lectura_archivos.py', wdir='K:/Usuarios/dan_1/Documents/ESCOM/
AplicacionesDeRedesParaLaComunicacion/Practica1/Lectura de archivos')

Conteo de palabras de cada archivo:

Archivo: A1.txt
amor: 7
casa: 7
enojo: 7
jardin: 7
juego: 7
pelota: 7

Archivo: A10.txt
amor: 29
casa: 4
enojo: 29
jardin: 4
juego: 29
pelota: 4

Archivo: A11.txt
amor: 1
casa: 39
enojo: 1
jardin: 39
juego: 7
pelota: 7
```

```
Archivo: A12.txt
amor: 56
casa: 2
enojo: 2
jardin: 2
juego: 43
pelota: 22

Archivo: A13.txt
amor: 2
casa: 109
enojo: 2
jardin: 2
juego: 2
pelota: 2

Archivo: A14.txt
amor: 1
casa: 1
enojo: 1
jardin: 1
juego: 1
pelota: 85

Archivo: A15.txt
amor: 18
casa: 18
enojo: 87
jardin: 18
juego: 18
pelota: 18
```

```
Archivo: A11.txt
amor: 0.010638
casa: 0.414894
enojo: 0.010638
jardin: 0.414894
juego: 0.074468
pelota: 0.074468

Archivo: A12.txt
amor: 0.440945
casa: 0.015748
enojo: 0.015748
jardin: 0.015748
juego: 0.338583
pelota: 0.173228

Archivo: A13.txt
amor: 0.016807
casa: 0.915966
enojo: 0.016807
jardin: 0.016807
juego: 0.016807
pelota: 0.016807

Archivo: A14.txt
amor: 0.011111
casa: 0.011111
enojo: 0.011111
jardin: 0.011111
juego: 0.011111
pelota: 0.944444
```

```
Archivo: A4.txt
amor: 1
casa: 1
enojo: 1
jardin: 1
juego: 37
pelota: 37

Archivo: A5.txt
amor: 30
casa: 1
enojo: 1
jardin: 1
juego: 30
pelota: 1

Archivo: A6.txt
amor: 29
casa: 1
enojo: 29
jardin: 1
juego: 1
pelota: 1

Archivo: A7.txt
amor: 2
casa: 24
enojo: 2
jardin: 24
juego: 2
pelota: 24
```

```
Archivo: A2.txt
amor: 1
casa: 14
enojo: 1
jardin: 14
juego: 1
pelota: 1

Archivo: A20.txt
amor: 76
casa: 76
enojo: 76
jardin: 76
juego: 76
pelota: 76

Archivo: A3.txt
amor: 1
casa: 1
enojo: 1
jardin: 18
juego: 1
pelota: 18

Archivo: A4.txt
amor: 1
casa: 1
enojo: 1
jardin: 1
juego: 37
pelota: 37
```

```
Archivo: A8.txt
amor: 3
casa: 3
enojo: 3
jardin: 25
juego: 25
pelota: 25

Archivo: A9.txt
amor: 58
casa: 1
enojo: 1
jardin: 1
juego: 58
pelota: 58

Frecuencia de palabras de cada archivo:

Archivo: A1.txt
amor: 0.166667
casa: 0.166667
enojo: 0.166667
jardin: 0.166667
juego: 0.166667
pelota: 0.166667

Archivo: A10.txt
amor: 0.292929
casa: 0.040404
enojo: 0.292929
jardin: 0.040404
juego: 0.292929
pelota: 0.040404
```

```
Archivo: A15.txt
amor: 0.101695
casa: 0.101695
enojo: 0.491525
jardin: 0.101695
juego: 0.101695
pelota: 0.101695

Archivo: A16.txt
amor: 0.092593
casa: 0.092593
enojo: 0.092593
jardin: 0.092593
juego: 0.314815
pelota: 0.314815

Archivo: A17.txt
amor: 0.087805
casa: 0.302439
enojo: 0.043902
jardin: 0.234146
juego: 0.146341
pelota: 0.185366

Archivo: A18.txt
amor: 0.392070
casa: 0.039648
enojo: 0.039648
jardin: 0.039648
juego: 0.145374
pelota: 0.343612
```

```
Archivo: A19.txt
amor: 0.011111
casa: 0.011111
enojo: 0.500000
jardin: 0.011111
juego: 0.455556
pelota: 0.011111

Archivo: A2.txt
amor: 0.031250
casa: 0.437500
enojo: 0.031250
jardin: 0.437500
juego: 0.031250
pelota: 0.031250

Archivo: A20.txt
amor: 0.166667
casa: 0.166667
enojo: 0.166667
jardin: 0.166667
juego: 0.166667
pelota: 0.166667

Archivo: A3.txt
amor: 0.025000
casa: 0.025000
enojo: 0.025000
jardin: 0.450000
juego: 0.025000
pelota: 0.450000
```

```
Archivo: A16.txt
amor: 20
casa: 20
enojo: 20
jardin: 20
juego: 68
pelota: 68

Archivo: A17.txt
amor: 18
casa: 62
enojo: 9
jardin: 48
juego: 30
pelota: 38

Archivo: A18.txt
amor: 89
casa: 9
enojo: 9
jardin: 9
juego: 33
pelota: 78

Archivo: A19.txt
amor: 1
casa: 1
enojo: 45
jardin: 1
juego: 41
pelota: 1
```

```
Archivo: A4.txt
amor: 0.012821
casa: 0.012821
enojo: 0.012821
jardin: 0.012821
juego: 0.474359
pelota: 0.474359

Archivo: A5.txt
amor: 0.468750
casa: 0.015625
enojo: 0.015625
jardin: 0.015625
juego: 0.468750
pelota: 0.015625

Archivo: A6.txt
amor: 0.467742
casa: 0.016129
enojo: 0.467742
jardin: 0.016129
juego: 0.016129
pelota: 0.016129

Archivo: A7.txt
amor: 0.025641
casa: 0.307692
enojo: 0.025641
jardin: 0.307692
juego: 0.025641
pelota: 0.307692
```

```
Archivo: A8.txt
amor: 0.035714
casa: 0.035714
enojo: 0.035714
jardin: 0.297619
juego: 0.297619
pelota: 0.297619

Archivo: A9.txt
amor: 0.327684
casa: 0.005650
enojo: 0.005650
jardin: 0.005650
juego: 0.327684
pelota: 0.327684

Conteo de palabras total en los archivos:

amor: 443

casa: 394

enojo: 327

jardin: 312

juego: 510

pelota: 571
```

Conclusiones

Daniel Murguia Jimenez:

Como nos hemos dado cuenta durante las clases y al desarrollar esta práctica, el manejo de hilos para la ejecución de problemas potencialmente complejos nos ayuda a disminuir la complejidad y el tiempo de ejecución, optimizando los programas para un manejo más eficiente de los datos; de igual manera al utilizar hilos podemos encargarnos de realizar diversas tareas de manera paralela por lo que podemos ahorrar poder de procesamiento al no utilizar procesos separados para esto.

Miranda Quijano Manuel Alejandro:

En esta primer práctica se observó una comparativa interesante entre las diferencias de usar hilos para tareas que requieren un largo procesamiento, con ello se reduce tiempo de ejecución. Agilizando el procesamiento de los programas que se realicen, se mantiene el ideal de cuidar los recursos en sistemas mucho más complejos donde sea ampliamente necesario.

Mora Guzman Jose Antonio:

La realización de esta practica fue para reafirmar los conocimientos ya vistos durante clases, y nos ayudo a ver mejor el como el manejar hilos hace que podamos realizar diversas tareas al mismo tiempo y así disminuimos el tiempo de ejecución, además de que nos sirvió para ver el manejo de hilos en el lenguaje Python y nos hizo dar cuenta que su manejo es mas sencillo en comparativa con otros lenguajes.