Universidad Nacional Autónoma de México

Facultad de Ingeniería

División de Ingeniería Mecánica e Industrial

Laboratorio de Cómputo de Ingeniería Mecatrónica (1964)

Profesor: Miguel Serrano Reyes
Semestre 2022-1

Práctica No. 1

Programación Concurrente

Nombre del Estudiante:
Jasso Garduño Juan José

Actividad 1. Desarrollar un script que acceda a 200 sitios web de manera secuencial y medir el tiempo que le lleva hacer tal operación

```python
#Actividad 1, Alumno: Juan José Jasso Garduño

import time     #Se importa la biblioteca time, se usa para medir el tiempo
import requests #Se importa la biblioteca requests, se usa para enviar peticiones HTTP

#Se crea una función para descargar información de un sitio
#Se imprime el número de archivo que se está leyendo y el sitio de donde proviene
def descargar_sitio(url,sesion):
    with sesion.get(url) as respuesta:
        print("Leyendo {} de {}".format(len((respuesta.content)),url))

#Se crea una función que recibe algunas url
#Después para cada url en sitios, se usa el método descargar sitio
def descargar_todo(sitios):
    with requests.Session() as sesion:
            for url in sitios:
                descargar_sitio(url,sesion)

#Si estamos en main, crea una lista con un par de sitios
#La lista de sitios se multiplica por 100 para tener 200 sitios para descargar
#Se establece un tiempo de inicio y se usa la función descargar todo, se le pasa como
#argumento la lista generada previamente
#Se crea una variable duración al finalizar la descarga
#Se imprime el número de sitios descargados y el tiempo que tardó
if __name__ == '__main__':
    sitios = [
    "https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html",
    "https://imagenenlaciencia.blogspot.com/2014/08/chamacos-mendigos.html"
    ]*100

    hora_inicio=time.time()
    descargar_todo(sitios)
    duracion=time.time()-hora_inicio
    print("Se descargaron {} sitios en {} segundos".
          format(len(sitios),duracion))
```

```
matplotlib.pyplot.plot.html
Leyendo 110310 de https://imagenenlaciencia.blogspot.com/
2014/08/chamacos-mendigos.html
Leyendo 74766 de https://matplotlib.org/stable/api/_as_gen/
matplotlib.pyplot.plot.html
Leyendo 110310 de https://imagenenlaciencia.blogspot.com/
2014/08/chamacos-mendigos.html
Leyendo 74766 de https://matplotlib.org/stable/api/_as_gen/
matplotlib.pyplot.plot.html
Leyendo 110310 de https://imagenenlaciencia.blogspot.com/
2014/08/chamacos-mendigos.html
Leyendo 74766 de https://matplotlib.org/stable/api/_as_gen/
matplotlib.pyplot.plot.html
Leyendo 110310 de https://imagenenlaciencia.blogspot.com/
2014/08/chamacos-mendigos.html
Leyendo 74766 de https://matplotlib.org/stable/api/_as_gen/
matplotlib.pyplot.plot.html
Leyendo 110310 de https://imagenenlaciencia.blogspot.com/
2014/08/chamacos-mendigos.html
Leyendo 74766 de https://matplotlib.org/stable/api/_as_gen/
matplotlib.pyplot.plot.html
Leyendo 110310 de https://imagenenlaciencia.blogspot.com/
2014/08/chamacos-mendigos.html
Se descargaron 200 sitios en 16.190971851348877 segundos
```

Actividad 2. Desarrollar un script que acceda a 200 sitios web mediante el uso de hilos y medir el tiempo que le lleva hacer tal operación

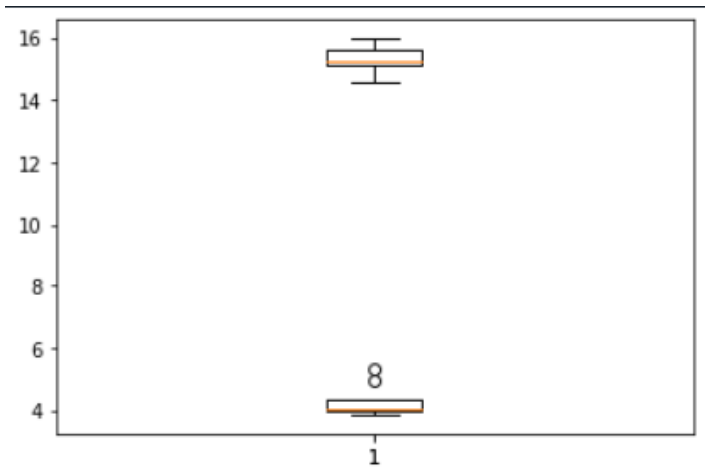
```python
#Actividad 2, Alumno: Juan José Jasso Garduño

import time     #Se importa la biblioteca time, se usa para medir el tiempo
import requests #Se importa la biblioteca requests, se usa para enviar peticiones HTTP
import threading #Se importa la biblioteca threading para el uso de hilos
import concurrent.futures #Provee una interfaz para ejecutar scripts de forma asincrónica

#Se crea un objeto de la clase threading.local
hilo_local = threading.local()

#Se crea una función para el uso de hilos con peticiones
def get_sesion():
    if not hasattr(hilo_local,"session"):
        hilo_local.session=requests.Session()
    return hilo_local.session

#Se crea una función para descargar información de un sitio, se le debe pasar una url
#Se obtiene una sesión
#Con la sesión se imprime el número de archivo que se está leyendo y el sitio de donde proviene
def descargar_sitio(url):
    sesion=get_sesion()
    with sesion.get(url) as respuesta:
        print("Leyendo {} de {}".format(len((respuesta.content)),url))

#Se crea una función que recibe una lista de urls
#Se usan caracteristicas concurrentes con 5 trabajadores
#Se usa el método descargar sitio junto con el ejecutor creado previamente para descargar los sitios
def descargar_todo(sitios):
    with concurrent.futures.ThreadPoolExecutor(max_workers = 5) as ejecutor:
            ejecutor.map(descargar_sitio,sitios)


#Si estamos en main, crea una lista con un par de sitios
#La lista de sitios se multiplica por 100 para tener 200 sitios para descargar
#Se establece un tiempo de inicio y se usa la función descargar todo, se le pasa como
#argumento la lista generada previamente
#Se crea una variable duración al finalizar la descarga
#Se imprime el número de sitios descargados y el tiempo que tardó
if __name__ == '__main__':
    sitios = ["https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html",
    "https://imagenenlaciencia.blogspot.com/2014/08/chamacos-mendigos.html"]*100

    hora_inicio=time.time()
    descargar_todo(sitios)
    duracion=time.time()-hora_inicio
    print("Se descargaron {} sitios en {} segundos".
          format(len(sitios),duracion))
```

```
Leyendo 161 de https://imagenenlaciencia.blogspot.com/2014/08/
chamacos-mendigos.html
Leyendo 161 de https://imagenenlaciencia.blogspot.com/2014/08/
chamacos-mendigos.html
Leyendo 74766 de https://matplotlib.org/stable/api/_as_gen/
matplotlib.pyplot.plot.html
Leyendo 74766 de https://matplotlib.org/stable/api/_as_gen/
matplotlib.pyplot.plot.html
Leyendo 74766 de https://matplotlib.org/stable/api/_as_gen/
matplotlib.pyplot.plot.html
Leyendo 161 de https://imagenenlaciencia.blogspot.com/2014/08/
chamacos-mendigos.html
Leyendo 74766 de https://matplotlib.org/stable/api/_as_gen/
matplotlib.pyplot.plot.html
Leyendo 161 de https://imagenenlaciencia.blogspot.com/2014/08/
chamacos-mendigos.html
Leyendo 161 de https://imagenenlaciencia.blogspot.com/2014/08/
chamacos-mendigos.html
Leyendo 161 de https://imagenenlaciencia.blogspot.com/2014/08/
chamacos-mendigos.html
Leyendo 161 de https://imagenenlaciencia.blogspot.com/2014/08/
chamacos-mendigos.html
Se descargaron 200 sitios en 4.191666603088379 segundos
```

Actividad 3. Tenemos dos formas distintas de descargar los sitios web, Crea un nuevo script donde ejecutes 10 veces cada una de ellas, guarda los tiempo de ejecución de las dos condiciones y genera un boxplot comparativo entre los tiempos obtenidos.

```python
#Actividad 3, Alumno: Juan José Jasso Garduño

#Se importa la biblioteca time, se usa para medir el tiempo
#Se importa la biblioteca requests, se usa para enviar peticiones HTTP
#Se importa la biblioteca threading para el uso de hilos
#Se importa la biblioteca concurrent.futures que provee una interfaz para ejecutar scripts de forma asincrónica
#Se importa la biblioteca matplotlib para el uso de gráficas
import time
import requests
import threading
import concurrent.futures
import matplotlib.pyplot as plt

#Se crea un objeto de la clase threading.local
hilo_local = threading.local()

#Se crean 2 listas vacías para almacenar la duración de cada iteración dependiendo del método para descarga
lista_1=[]
lista_2=[]

#Se crea un ciclo con 10 iteraciones para descargar sitios de manera paralela usando hilos
#Se guarda en una lista la duración que tuvo cada iteración
#Se imprime el número de sitios descargados junto con su duración

for i in range(10):
    print("------------------------------------------")
    print("Paralela iteración {}".format(i))
    print("------------------------------------------")
    def get_sesion():
        if not hasattr(hilo_local,"session"):
            hilo_local.session=requests.Session()
        return hilo_local.session

    def descargar_sitio(url):
        sesion=get_sesion()
        with sesion.get(url) as respuesta:
            print("Leyendo {} de {}".format(len((respuesta.content)),url))

    def descargar_todo(sitios):
        with concurrent.futures.ThreadPoolExecutor(max_workers = 5) as ejecutor:
                ejecutor.map(descargar_sitio,sitios)

    if __name__ == '__main__':
        sitios = ["https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html",
        "https://imagenenlaciencia.blogspot.com/2014/08/chamacos-mendigos.html"]*100

        hora_inicio=time.time()
        descargar_todo(sitios)
        duracion=time.time()-hora_inicio
        lista_1.append(duracion)
        print("Se descargaron {} sitios en {} segundos".
              format(len(sitios),duracion))
```

```python
#Se crea un ciclo con 10 iteraciones para descargar sitios de manera secuencial
#Se guarda en una lista la duración que tuvo cada iteración
#Se imprime el número de sitios descargados junto con su duración

for j in range(10):
    print("-----------------------------------------")
    print("Secuencial iteración {}".format(j))
    print("-----------------------------------------")
    def descargar_sitio(url,sesion):
        with sesion.get(url) as respuesta:
            print("Leyendo {} de {}".format(len((respuesta.content)),url))

    def descargar_todo(sitios):
        with requests.Session() as sesion:
                for url in sitios:
                    descargar_sitio(url,sesion)

    if __name__ == '__main__':
        sitios = ["https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html",
        "https://imagenenlaciencia.blogspot.com/2014/08/chamacos-mendigos.html"]*100

        hora_inicio=time.time()
        descargar_todo(sitios)
        duracion=time.time()-hora_inicio
        lista_2.append(duracion)
        print("Se descargaron {} sitios en {} segundos".
              format(len(sitios),duracion))

#Al terminar ambos procesos se muestran las listas con los tiempos que tuvo cada método, respectivamente

print("\n")
print("Termino de ejecuciones \n")
print("Tiempos de manera paralela [s]: ")
print(lista_1)
print("\nTiempos de manera secuencial [s] : ")
print(lista_2)

#Se muestra un boxplot para los tiempos de duración que tuvo cada método, respectivamente.

plt.boxplot(lista_1)
plt.boxplot(lista_2)
plt.show()
```




```
Termino de ejecuciones

Tiempos de manera paralela [s]:
[5.3096983432769775, 4.064146518707275, 4.353618621826172,
3.9893314838409424, 3.8522398471832275, 4.04638671875,
3.8986728191375732, 4.009282827377319, 4.920030355453491,
4.295286655426025]

Tiempos de manera secuencial [s] :
[15.626420021057129, 15.203309297561646, 15.137123346328735,
14.541875123977661, 15.323013305664062, 15.142770051956177,
15.9590904712677, 15.01565170288086, 15.897516965866089,
15.51624846458435]
```

Conclusiones:

Como se puede observar en la salida de la actividad 1 y 2, se tienen una duración de 16.19 [s] y 4.19[s] respectivamente, por lo que al hacer la descarga usando hilos es aproximadamente 4 veces más rápido.

En cuando a la actividad 3, podemos observar el boxplot inferior, el cual corresponde a las descargas usando hilos, se observa que la mayoría de las descargas tardaron alrededor de 4 segundos, y sólo en 2 ocasiones más de 4.9 segundos, siendo estos outliers. En cuando al boxplot para las descargas secuenciales la mayoría de las descargas fueron de alrededor de 15 segundos.