PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2523 - Sistemas Distribuidos - 2/2020

Tarea 3

Entrega: 16 Diciembre 2020, 20:00 hrs.

Objetivo

El objetivo de esta tarea es utilizar el lenguaje `Python` en conjunto con `Redis`, en un caso práctico de *sistemas distribuidos*. En esta entrega deben implementar una simulación de ejecución de tareas de forma asíncrona.

Python - Redis

Los siguientes recursos son útiles para comenzar con `Python` y `Redis Queue`.

1. Redis Queue: Task queues

Tareas a resolver

En esta tarea se debe construir una simulación de como validar un hash sha y encontrar un elemento que a partir del hash anterior genere uno nuevo con cierto formato especificado. Este debe ser capaz de recibir tres tipos de clientes, el cual consisten en cliente común, alto y VIP; donde el cliente de alto tiene prioridad en la ejecución sobre el cliente común, al igual que un VIP tendrá prioridad sobre los alto. Cada cliente podrá realizar un tipo de acción, el cuál se especificará más adelante.
El código debe:

- Leer un archivo de instrucciones (sección Input)
- Hacer encolamiento y ejecución de tareas de forma asíncrona
- Entregar el orden de atención de clientes y el resultado de este

Tipos de clientes:

- Cliente común: C
- Cliente alto: T
- Cliente VIP: V

Input y ejecución

El código podrá ser ejecutado leyendo el archivo instrucciones `instruction_file.txt` desde un *flag* ó como argumento.

1. Leer archivo *flag*: `python codigo.py instruction_path="./instruction_file.txt"`

2. Leer archivo como argumento: `python codigo.py instruction_file.txt`

El archivo `instruction_file.txt` es un archivo de instrucciones que contendrá por linea el tipo de cliente (X) en conjunto a su identificador numérico (i), de la forma Xi. Tambieén se incluye el $hash\_hexadigit$ inicial y valores de entrada para esta acción ($last\_k$ y end_string). Cada línea tiene esta forma:

- $Xi\ hash\_hexadigit\ last\_k\ end\_string$: indica que el cliente Xi ejecutará la acción de obtener el valor de k' a partir del $hash\_hexadigit$ en conjunto con las variables que se encuentran posterior a la instrucción. El cliente X puede ser de tipo C, T, o V.

Cada línea, con sus parámetros, deberá ejecutar la siguiente acción usando los argumentos que recibe

- Debe entregar la cantidad de iteraciones k' que se debe realizar para obtener un SHA hash que finalice con un string especifico end_string. Es por esto que la instrucción debe recibir un hash previo ($last\_hash$), un integer $last\_k$ y un string end_string. A continuación se adjunta un ejemplo de código que realiza esta acción:

```python
import hashlib

def validate_proof_of_work(last_k, last_hash, k, end_hash):
    sha = hashlib.sha256(f'{last_k}{last_hash}{k}'.encode())
    return sha.hexdigest()[:len(end_hash)] == end_hash

def generate_proof_of_work(last_k, last_hash, end_hash):
    k = 0
    while not validate_proof_of_work(last_k, last_hash, k, end_hash):
        k += 1

    return k
```

  Por ejemplo, a continuación se entrega un ejemplo de linea de instrucción:

$$C1\ 235f8338e0a691e5f99bea4b09082b89ea4165c498b46036614ff2f1081a6a46\ 0\ 0000$$

  La cual corresponde a entregar el hash previo creado a partir del string: $0testString0$, que correspondería a:

$$235f8338e0a691e5f99bea4b09082b89ea4165c498b46036614ff2f1081a6a46$$

$$last\_k = 0$$

$$end\_hash = 0000$$

  Del cual se espera que el valor k' sea igual a 17850.

  Para el funcionamiento de la tarea, se permite que se separe el funcionamiento del script que se encarga de hacer el encolamiento de las instucciones con respecto al $worker$ que realice la ejecución de la acción por cada instrucción. En caso de decidir dejar todo el funcionamiento junto o separado, es necesario que se especifique en el comienzo del informe o en un $ReadMe$.

Output

Todos los procesos deben registrar sus resultados en un mismo archivo de salida, donde se debe imprimir la instrucción que se ejecuta junto con el resultado. Este archivo debe respetar el orden de ejecución según prioridad y orden de aparición.

Informe

Debe describir el funcionamiento e implementación de cada mecánica según el tipo de cliente y acción que se pueda realizar.

Desglose de puntaje

Código (12 puntos)

- Creación de colas: 2 puntos.
- Ejecución asíncrona de colas: 6 puntos.
- Funcionamiento acción: 2 puntos.
- Generación de output correcto: 2 puntos.

Si el código no ejecuta completamente, no se puede evaluar esta parte de la tarea.

Informe (2 puntos)

- Formalidades: 0.5 punto por el informe en PDF, incluir nombre y cumplir el formato de entrega.
- Descripción de código e implementación de cola(s): 1.5 puntos.

Formato de entrega

El informe debe ser entregado en formato PDF junto al código en un .zip o .rar en canvas. El nombre del archivo comprimido debe ser `NúmeroDeAlumno_IIC2523-2_Tarea_3`.

Integridad académica

Los alumnos de la Escuela de Ingeniería de la Pontificia Universidad Católica de Chile deben mantener un comportamiento acorde a la Declaración de Principios de la Universidad. En particular, se espera que mantengan altos estándares de honestidad académica. Cualquier acto deshonesto o fraude académico está prohibido; los alumnos que incurran en este tipo de acciones se exponen a un Procedimiento Sumario. Es responsabilidad de cada alumno conocer y respetar el documento sobre Integridad Académica publicado por la Dirección de Docencia de la Escuela de Ingeniería.

Específicamente, para los cursos del Departamento de Ciencia de la Computación, rige obligatoriamente la siguiente política de integridad académica. Todo trabajo presentado por un alumno para los efectos de la evaluación de un curso debe ser hecho individualmente por el alumno, sin apoyo en material de terceros. Por "trabajo" se entiende en general las interrogaciones escritas, las tareas de programación u otras, los trabajos de laboratorio, los proyectos, el examen, entre otros. Si un alumno copia un trabajo, obtendrá nota final 1,1 en el curso y se solicitará a la Dirección de Docencia de la Escuela de Ingeniería que no le permita retirar el curso de la carga académica semestral. Por "copia" se entiende incluir en el trabajo presentado como propio partes hechas por otra persona.

Obviamente, está permitido usar material disponible públicamente, por ejemplo, libros o contenidos tomados de Internet, siempre y cuando se incluya la referencia correspondiente.

Lo anterior se entiende como complemento al Reglamento del Alumno de la Pontificia Universidad Católica de Chile. Por ello, es posible pedir a la Universidad la aplicación de sanciones adicionales especificadas en dicho reglamento.