R10725012 呂晟維
情境: Count of URL Access Frequency

- 每次 user 造訪一個網頁，就會留下一則改網站 domain name 的 Log，將 domain name 儲存在 array 中。(可以想像成實體機器的硬碟)
- 測試資料共有 1000 則 Log，每條 Log 都記錄了 user 造訪的網域名稱。情境中有 8 種網域被造訪，依照指定的數目產生測資並打亂。
- 在所有的網頁請求中，我們只監控 user 使用 "https://www.messenger.com" 和 "https://chat.openai.com" 的情形，看看他們是否偷懶聊天和找答案。

Map Stage

- 總共 1000 則 Log 切成 10 份區塊交由 10 個 map worker，個別執行關鍵字查詢的工作。
- Map worker 會接受一段區段的 array，由主程式指派資料區段 (實際上會給硬碟磁區的 start_index 和 end_index)。Map worker 會遍歷區段的 subarray 的記錄，若發現目標網域的紀錄，則輸出成 key-value pair <"domain": 1> 到一個 List 中。
- return 中間產物的 List of dict (key-value) 給主程式。

Reduce Stage

- 等待所有 map worker 計數完成並回傳，主程式會將所有的 List 蒐集並串接成一個 List of dict (可以想像成主程式知道 map worker 的資料的實際儲存位址)
- 將 Array 指派給 reduce worker 來進行加總。一個 reduce woker 負責累加一個 domain 的造訪次數，回傳一個單值表示最終的結果。實作上方便起見，使用一個 reduce woker 便完成加總的工作了。

最後主程式 display mapReduce 的過程和統計的結果。得出被監控的每個網域分別被造訪幾次。

產生測試資料

In [ ]:
```python
# len of 8 domain names
domains = [
    "https://chat.openai.com",
    "https://wiki.biligame.com",
    "https://music.youtube.com",
    "https://www.mybib.com",
    "https://www.messenger.com",
    "https://www.techpowerup.com",
    "https://imgur.com",
    "https://my.ntu.edu.tw"
]
counts = [100, 10, 88, 77, 500, 166, 33, 26]
sum(counts)
```

Out[ ]:
```
1000
```

In [ ]:
```python
# generate 1000 log data & shuffle with a seed
log_data = []
for domain, count in zip(domains, counts):
    log_data.extend([domain] *count)

import random
random.Random(44444).shuffle(log_data)
log_data[:5]
```

Out[ ]:
```
['https://www.messenger.com',
 'https://www.messenger.com',
 'https://www.techpowerup.com',
 'https://www.messenger.com',
 'https://www.messenger.com']
```

Step: Map

input: log data after splited

In [ ]:
```python
def map_worker(data: list):
    result = {}
    for record in data:
        if record == "https://www.messenger.com":
            result[record] = result.get(record, 0) + 1
        elif record == "https://chat.openai.com":
            result[record] = result.get(record, 0) + 1
    return result
```

Step: Reduce

input: the intergrated map result

In [ ]:
```python
def reduce_worker(results:list[dict], target_domain:str=None):
    final_result = {}
    for result in results:
        for domain, count in result.items():
            if target_domain and domain != target_domain:
                continue
            final_result[domain] = final_result.get(domain, 0) + count
    return final_result
```

Full Code

- https://docs.python.org/3/library/multiprocessing.html#using-a-pool-of-workers

In [ ]:
```python
from multiprocessing import Pool

def map_worker(data):
    result = {}
    for record in data:
        if record == "https://www.messenger.com":
            result[record] = result.get(record, 0) + 1
        elif record == "https://chat.openai.com":
            result[record] = result.get(record, 0) + 1
    return result

def reduce_worker(results:list[dict], target_domain:str=None):
    final_result = {}
    for result in results:
        for domain, count in result.items():
            if target_domain and domain != target_domain:
                continue
            final_result[domain] = final_result.get(domain, 0) + count
    return final_result

def main(log_data: list):
    # Split log_data into chunks for map workers
    chunk_size = len(log_data) // 10
    chunks = [log_data[i:i+chunk_size] for i in range(0, len(log_data), chunk_size)

    # Map stage - process chunks using map workers
    with Pool(processes=10) as pool:
        map_results:list[dict] = pool.map(map_worker, chunks)
        # print(map_results)

    # Reduce stage - process map results using reduce worker
    reduce_result = reduce_worker(map_results)

    # Display the result
    for domain, count in reduce_result.items():
        print(f"Target Domain: {domain}, Visits: {count}")

    return map_results, reduce_result

if __name__ == '__main__':
    print(f"Raw data lenght: {len(log_data)} \n\n*結果*")
    map_results, reduce_result = main(log_data)

    print('\n回頭檢視 Map 的過程:')
    for i,m in enumerate(map_results):
        print(f"map worker [{i}]'s output: {m}")
```

```
Raw data lenght: 1000

*結果*
Target Domain: https://www.messenger.com, Visits: 500
Target Domain: https://chat.openai.com, Visits: 100

回頭檢視 Map 的過程:
map worker [0]'s output: {'https://www.messenger.com': 50, 'https://chat.openai.co
m': 8}
map worker [1]'s output: {'https://www.messenger.com': 46, 'https://chat.openai.co
m': 14}
map worker [2]'s output: {'https://www.messenger.com': 53, 'https://chat.openai.co
m': 9}
map worker [3]'s output: {'https://www.messenger.com': 49, 'https://chat.openai.co
m': 6}
map worker [4]'s output: {'https://www.messenger.com': 52, 'https://chat.openai.co
m': 8}
map worker [5]'s output: {'https://www.messenger.com': 59, 'https://chat.openai.co
m': 9}
map worker [6]'s output: {'https://www.messenger.com': 49, 'https://chat.openai.co
m': 9}
map worker [7]'s output: {'https://www.messenger.com': 50, 'https://chat.openai.co
m': 11}
map worker [8]'s output: {'https://chat.openai.com': 11, 'https://www.messenger.co
m': 48}
map worker [9]'s output: {'https://www.messenger.com': 44, 'https://chat.openai.co
m': 15}
```

這邊我們驗證計算結果，確認 messenger.com 有 500 次造訪，openai.com 有 100 次造訪。

In [ ]:
```python
# validate the counts
import pandas as pd
pd.Series(log_data).value_counts()
```

Out[ ]:
```
https://www.messenger.com      500
https://www.techpowerup.com    166
https://chat.openai.com        100
https://music.youtube.com       88
https://www.mybib.com           77
https://imgur.com               33
https://my.ntu.edu.tw           26
https://wiki.biligame.com       10
dtype: int64
```

In [ ]: