通訊網路實驗 Lab2 Report

110511254 徐煜絨

一、實驗內容

這次實驗使用 iMatix 和許多人組成的社群開發的 ZeroMQ (ZMQ)來達成訊息傳輸，其特點包含速度快且支援 python 和 java 等大部分流行程式語言。

ZMQ 有三種常見模式，包括 Request-Reply, Publish-Subscribe, Parallel-Pipeline。實驗一是 Request-Reply 模式，指的是 server 和 client 端要互相收送資料，client 端在發出 request 後，server 需要 reply。實驗二則是 Publish-Subscribe，為單向的 data distribution，由 server 持續發送資料給有訂閱主題的 client。

二、程式碼（自己寫的地方）

Q1 分散式加法計算

Server:

Server 端接收 client 傳來的數字（要轉成 int），運算後傳回給 client。

```
a, b = socket.recv_multipart()
c = int(a) + int(b)
print("Compute " + a + " + " + b + " and send response")
socket.send_multipart([str(c), str(os.getpid())])
```

Client:

Client 會發送隨機產生的 a, b 兩數字到 server 端做運算，之後再接收 server 傳回來的計算結果並 print 出來。

```
print("Compute " + a + " + " + b)
socket.send_multipart([a, b])
rep = socket.recv_multipart()
print("= " + rep[0] + " (from worker " + rep[1] + ")")
```

Q2 溫溼度感測

Server:

Server 會將測量結果傳給 client

```
humidity, temp = Adafruit_DHT.read_retry(sensor, gpio)
```

Client:

Client 要訂閱才拿得到 server 傳來的資料

```
topicfilter1 = "humidity"
topicfilter2 = "temp"
socket.setsockopt(zmq.SUBSCRIBE, topicfilter1)
socket.setsockopt(zmq.SUBSCRIBE, topicfilter2)
```

Client 會印出接收到的資料，再來把 10 筆溫度、濕度各別相加，最後得平均。

```
mes = socket.recv()
print(mes)
splitted_str = mes.split()
temp_sum += int(splitted_str[2])
mes = socket.recv()
print(mes)
splitted_str = mes.split()
humid_sum += int(splitted_str[2])
```

三、實驗結果

Q1 分散式加法計算

Client:

```
pi@raspberrypi:~$ sudo python Lab2_1_client.py
Compute 84 + 48
= 132 (from worker 990)
Compute 50 + 80
= 130 (from worker 998)
Compute 46 + 11
= 57 (from worker 1006)
Compute 9 + 88
= 97 (from worker 990)
Compute 48 + 66
= 114 (from worker 998)
Compute 82 + 51
= 133 (from worker 1006)
Compute 77 + 100
= 177 (from worker 990)
Compute 43 + 94
= 137 (from worker 998)
Compute 14 + 56
= 70 (from worker 1006)
pi@raspberrypi:~$
```

Server:

```
pi@raspberrypi:~ $ sudo python Lab2_1_server.py
Worker 990 is running ...
Compute 84 + 48 and send response
Compute 9 + 88 and send response
Compute 77 + 100 and send response
```

```
pi@raspberrypi:~ $ sudo python Lab2_1_server.py
Worker 998 is running ...
Compute 50 + 80 and send response
Compute 48 + 66 and send response
Compute 43 + 94 and send response
```

```
pi@raspberrypi:~ $ sudo python Lab2_1_server.py
Worker 1006 is running ...
Compute 46 + 11 and send response
Compute 82 + 51 and send response
Compute 14 + 56 and send response
```

Q2 溫濕度感測

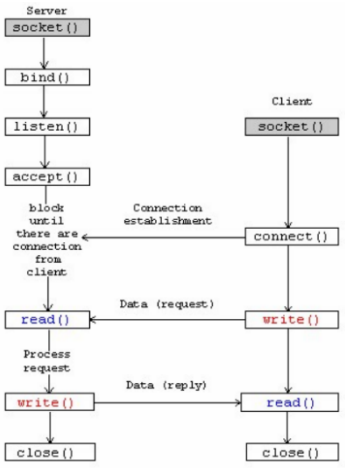
```
pi@raspberrypi:~ $ sudo python Lab2_2_client.py
Start!!

humidity = 51
temp = 27
humidity = 51
temp = 27
humidity = 51
temp = 27
humidity = 51
temp = 27
humidity = 52
temp = 27
humidity = 53
temp = 27
humidity = 54
temp = 28
humidity = 55
temp = 28
humidity = 56
temp = 28
humidity = 57
temp = 28
avg temperature: 53.1
avg humidity: 27.4
```

四、實驗問題

1. 考慮本次 Q1 和 Q2 的實驗流程下來，請問 ZMQ 的 client 與 server 間的 connect、bind 順序不同的話會對實驗內容有影響嗎？請說明原因。

Server 端使用 bind 而 client 端使用 connect。分別測試 Q1 和 Q2，當先執行 client 再執行 server，還是可以正常運作。所以推測順序不同對輸出結果不會有影響。



如果先執行 server，當 client 端開始執行，client 就能立刻收到資料，顯示推播內容。但如果先執行 client，此時 ZMQ 會持續嘗試連線，直到 server 開始執行並建立 bind，此時連線成功，client 開始接收 server 船過去的 data。

2. 列舉 ZMQ 三種常見模式的優缺點。

(1) Request-Reply

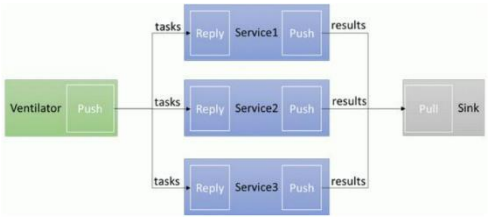
這種模式常見且比較有保障，當 server 沒有在線，client 可以發現並等待，因為在這種模式下，client 向 server 發出 request 後，server 一定要回復訊息給 client 端。然而，對於 basic request-reply 而言，當 request、reply 在運送過程中遺失，或是 server crashes，client 都會持續等待。另外，server 要對多個 client 收發訊息，可能導致 server 工作量太大。

(2) Publish-Subscribe

Server 會對所有 client 廣播，其發布的所有訊息都會被有訂閱的 client 接收，因為是 client 訂閱，所以可以有效過濾訊息。Publish-subscribe 在速度上雖然比 request-reply 快，但缺點是 client 沒有收到該筆訊息，訊息不會被保留，server 也不會知道傳輸上出問題並補發。

(3) Parallel-Pipeline

這是分散式的處理模式。由 ventilator 推送資料、worker (service) 進行資料分析、sink 彙整 worker 處理完的資料。這個模式的好處是可以同步運算大量資料，且比 publish-submit 多了資料快取和處理負載。最大的優點是當連線中斷時，資料不會遺失，再重新連線後資料會傳到指定位子。



3. 試想 ZMQ 三種常見模式（Request-Reply、Publish-Subscribe、Parallel-Pipeline）分別有哪些應用？（愈詳細且創新分數越高）

在交易系統或是股票系統中，request-reply 可以扮演重要角色。為了要即時買賣股票，在短時間內可能會有多筆資料和訊息同時湧入，此時資料會被送進 threads，所以 thread 必須有一定能力負荷所有資訊。由於 sockets 可以互相連結，同個 socket 又可以連結不同 address，使得不同 client 可以和相同 thread 連線。在處理多筆資料時，ZMQ 能同時用不同 thread 接收資料，並且反向將成功或失敗傳遞的通知回傳。

另外是檔案的儲存。因為每台裝置或硬碟的容量有限，如果可以透過 publish-subscribe 的模式讓相同網域內的儲存空間或硬碟相互連線，我

們可以在 server 上用不同的 topic 歸類檔案，隨後檔案就能逕自被傳到訂閱相關 topic 的 client 儲存空間中，或是加上其他功能，例如隨時回報剩餘儲存空間。

4. 本次實驗心得，你學到了什麼東西？

由於有現成的模組和技術，上一個主題也接觸過溫溼度感測器，所以操作起來比較沒這麼困難。但課後和同學討論第一題能平均分配的原因時並沒有得到一個明確的答案，在查詢後才發現 ZMQ 在給予各 worker 工作時有自動平均機制，以讓負載自動平衡。

Reference:

1. https://zake7749.github.io/2015/03/17/SocketProgramming/
2. https://reurl.cc/97Za08
3. http://www.runpc.com.tw/content/content.aspx?id=109829
4. https://zguide.zeromq.org/docs/chapter4/