컴퓨터 네트워크

[실습 5]

컴퓨터공학과

201702042 우정균

1. 멀티 프로세스 서버 소스코드

```python
import os
import sys
import socket
import time
import multiprocessing

HOST = ''
PORT = 8080

def worker(conn, addr):
    print(f'Connected by {addr}')
    data = conn.recv(1500)
    ptr = data.find('\r\n'.encode('utf-8'))
    header = data[:ptr]
    left = data[ptr:]
    request = header.decode('utf-8')
    method, path, protocol = request.split(' ')
    print(f'Received: {method} {path} {protocol}')
    if not data:
        return
    if path == '/':
        path = '/index.html'
    path = f'.{path}'
    if not os.path.exists(path):
        path = './notfound.html'
        header = 'HTTP/1.1 404 Not Found\r\n'
    else :
        header = 'HTTP/1.1 200 OK\r\n'

    if os.path.splitext(path)[1] != '.jpg' or os.path.splitext(path)[1] == '.jpeg':
        with open(path, 'r') as f:
            body = f.read()
            body = body.encode('utf-8')
    else :
        with open(path, 'rb') as f:
            body = f.read()

    header = f'{header}Server: Our server\r\n'
    header = f'{header}Connection: close\r\n'

    if os.path.splitext(path)[1] == '.html':
        header = f'{header}Content-Type: text/html;charset=utf-8\r\n'
        header = f'{header}Accept: text/html, text/css, text/javascript, image/
        jpeg\r\n'
      if os.path.splitext(path)[1] == '.css':
        header = f'{header}Content-Type: text/css; charset=utf-8\r\n'
      if os.path.splitext(path)[1] == '.js':
        header = f'{header}Content-Type: text/javascript; charset=utf-8\r\n'
      if os.path.splitext(path)[1] == '.jpg' or os.path.splitext(path)[1] == '.jpeg':
        header = f'{header}Content-Type: image/jpeg\r\n'
      header = f'{header}Content-Length: {len(body)}\r\n'
      header = f'{header}\r\n'
      header = header.encode('utf-8')
      response = header + body
      conn.sendall(response)

  with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
      s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
      s.bind((HOST, PORT))
      s.listen(1)
      print(f'Start server with {sys.argv}')
      while True:
          try:
              conn, addr = s.accept()
              process = multiprocessing.Process(target=worker,
                                                args=(conn, addr))
              process.start()
              print(f'Start child worker {process}')
          except KeyboardInterrupt:
              print('Shutdown server')
              for process in multiprocessing.active_children():
                  print('Terminate process {0}'.format(process))
                  process.terminate()
                  process.join()
              break
```

실습자료의 코드에서 worker 함수를 완성 시켜줬다. worker 함수에는 기존 서버 코드에서 클라이언트를 accept 한 뒤의 코드를 거의 그대로 옮겼다.

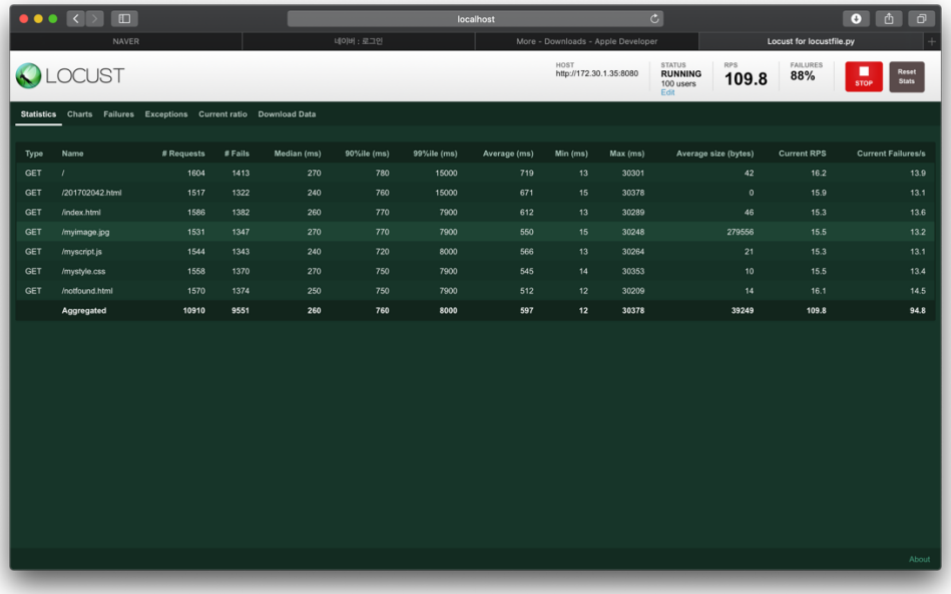
2. 멀티 쓰레드 서버 소스코드

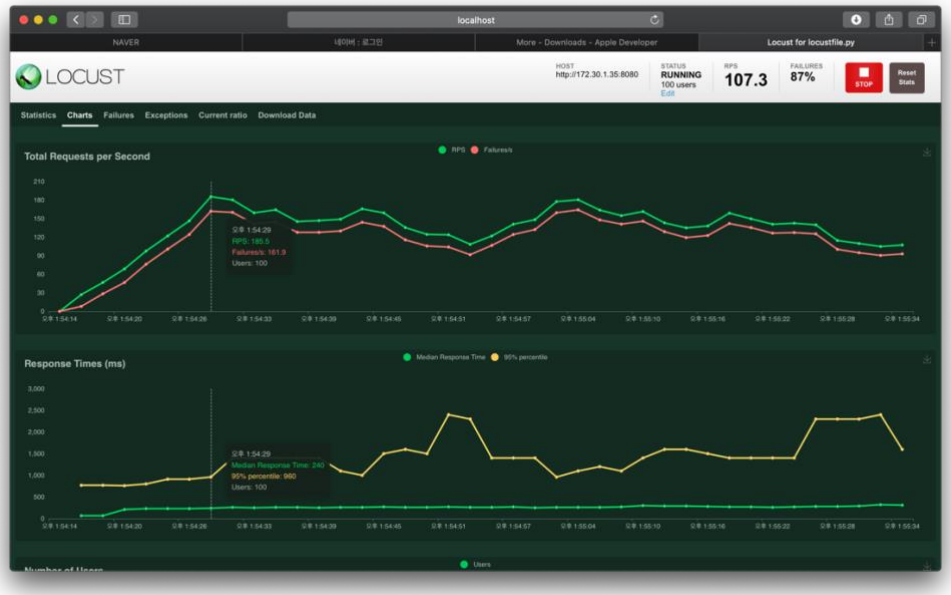
```python
import os
import sys
import socket
import time
import threading

HOST = ''
PORT = 8080

def worker(conn, addr):
    print(f'Connected by {addr}')
    data = conn.recv(1500)
    ptr = data.find('\r\n'.encode('utf-8'))
    header = data[:ptr]
    left = data[ptr:]
    request = header.decode('utf-8')
    method, path, protocol = request.split(' ')
    print(f'Received: {method} {path} {protocol}')
    if not data:
        return
    if path == '/':
        path = '/index.html'
    path = f'.{path}'
    if not os.path.exists(path):
        path = './notfound.html'
        header = 'HTTP/1.1 404 Not Found\r\n'
    else :
        header = 'HTTP/1.1 200 OK\r\n'

    if os.path.splitext(path)[1] != '.jpg' or os.path.splitext(path)[1] == '.jpeg':
        with open(path, 'r') as f:
            body = f.read()
            body = body.encode('utf-8')
    else :
        with open(path, 'rb') as f:
            body = f.read()

    header = f'{header}Server: Our server\r\n'
    header = f'{header}Connection: close\r\n'

    if os.path.splitext(path)[1] == '.html':
        header = f'{header}Content-Type: text/html;charset=utf-8\r\n'
        header = f'{header}Accept: text/html, text/css, text/javascript, image/
        jpeg\r\n'
    if os.path.splitext(path)[1] == '.css':
        header = f'{header}Content-Type: text/css; charset=utf-8\r\n'
    if os.path.splitext(path)[1] == '.js':
        header = f'{header}Content-Type: text/javascript; charset=utf-8\r\n'
    if os.path.splitext(path)[1] == '.jpg' or os.path.splitext(path)[1] == '.jpeg':
        header = f'{header}Content-Type: image/jpeg\r\n'
    header = f'{header}Content-Length: {len(body)}\r\n'
    header = f'{header}\r\n'
    header = header.encode('utf-8')
    response = header + body
    conn.sendall(response)


with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    s.bind((HOST, PORT))
    s.listen(1)
    print(f'Start server with {sys.argv}')
    while True:
        try:
            conn, addr = s.accept()
            thread = threading.Thread(target=worker,
                                      args=(conn, addr))
            thread.start()
            print(f'Start child worker {thread}')
        except KeyboardInterrupt:
            print('Shutdown server')
            for thread in threading.enumerate():
                if thread.getName() == 'MainThread':
                    continue
                print('Join thread {0}'.format(thread))
                thread.join(timeout=1)
            break
```

실습자료의 코드에서 worker 함수를 완성 시켜줬다. 멀티 프로세스 서버의 코드와 똑같다.

3.성능 비교

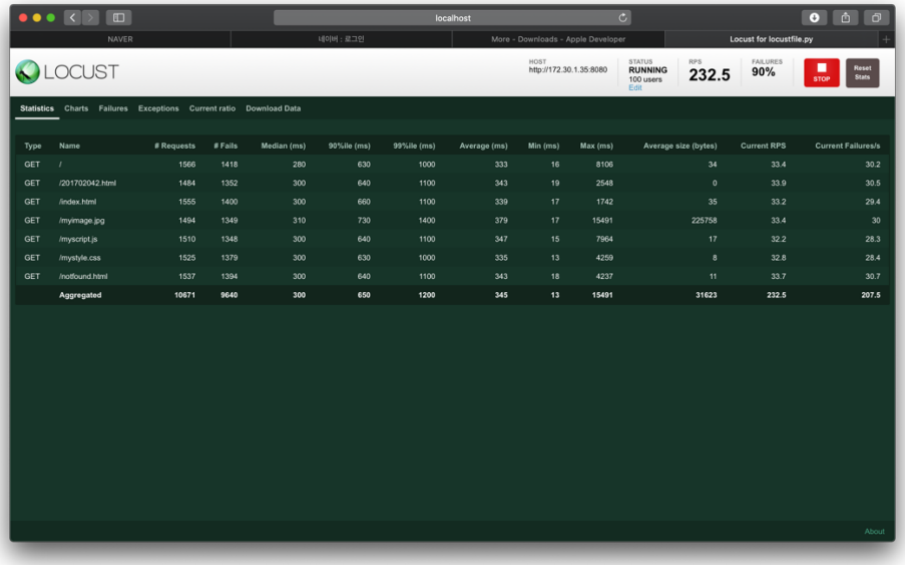
기존의 싱글 프로세스 서버

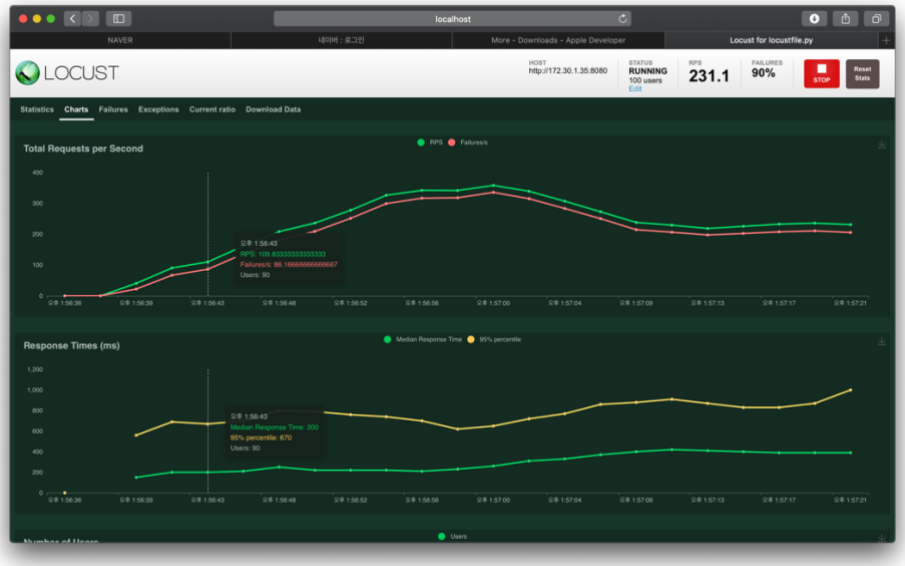




10910번의 요청을 했을 때, 평균 응답속도는 597ms이고 RPS는 100~200 사이를 왔다갔다한다.

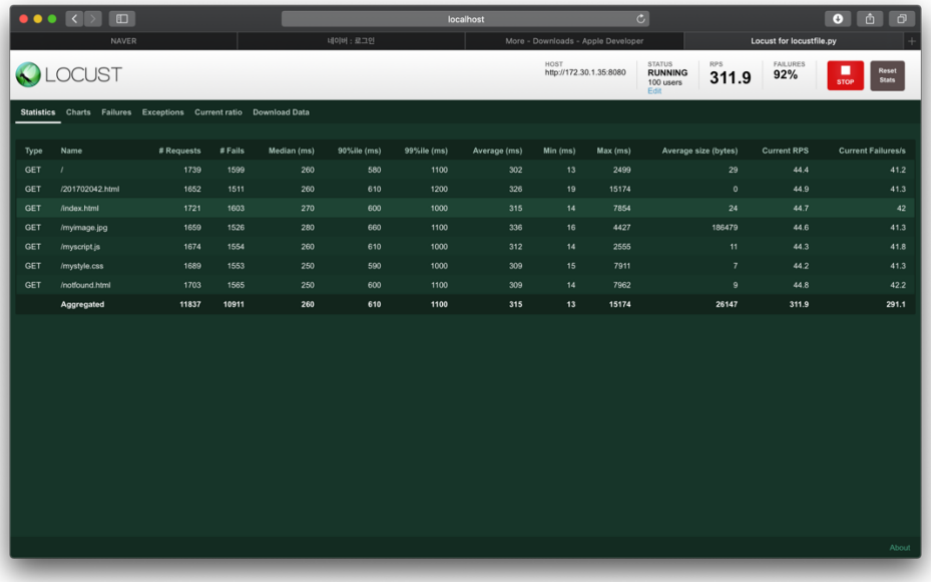
멀티 프로세스 서버

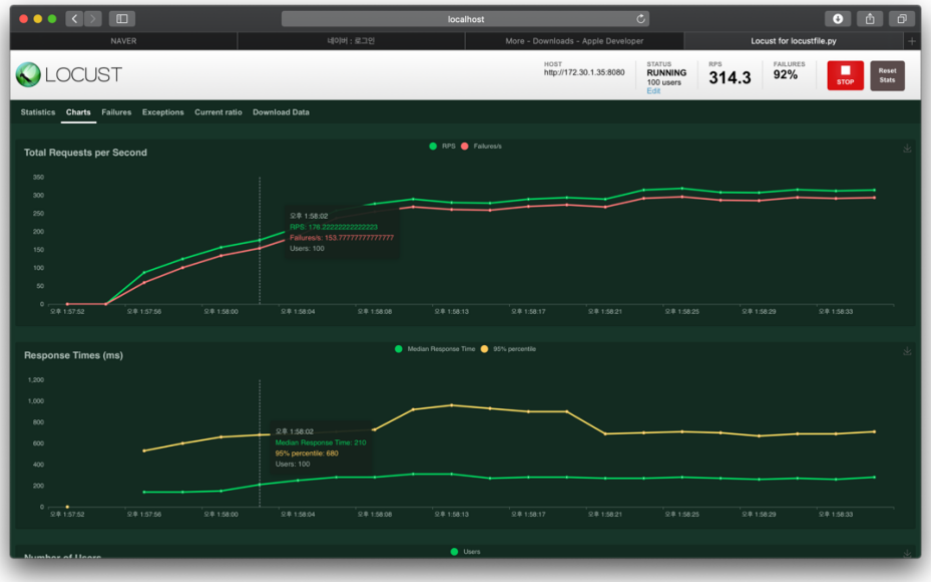




10671번의 요청을 했을 때, 평균 응답속도는 345ms이고 RPS는 200~300 초반대까지 나온다.

멀티 쓰레드 서버





11837번의 요청을 했을 때, 평균 응답속도는 315ms이고, RPS는 300내외의 값이 나온다.

성능 비교 정리

싱글 프로세스
- 평균 응답속도: 597ms
- RPS: 가변 100~200

멀티 프로세스
- 평균 응답속도: 345ms
- RPS: 가변 200~300

멀티 쓰레드
- 평균 응답속도: 315ms
- RPS: 가변 300 내외

싱글 프로세스 서버보다 멀티 프로세스 혹은 멀티 쓰레드를 사용하는 서버의 성능이 확실하게 좋다.
그리고 멀티 프로세스 서버보다 멀티 쓰레드 서버의 성능이 약간 더 좋게 나오는데, 멀티 프로세스를 사용하는 경우 프로세스간 통신 비용이 더 들기 때문이라고 생각한다. 그러나 멀티 스레드의 경우 하나의 프로세스 안에서 작동하므로 안정성이 멀티 프로세스보다 떨어진다는 단점이 있다.