**Computer Systems Technology**

British Columbia Institute of Technology

COMP 8005 - Assignment2- Testing

Albert Huang

Feb 28, 2018

# Table of Contents

[Table of Contents 2](#_Toc1153764661)

[1. Multi-Thread Testing 3](#_Toc169100092)

[1.1 Test Outline 3](#_Toc1557394089)

[1.2 Test Case Descriptions 4](#_Toc227366271)

[1.2.1 Test 1 4](#_Toc368101859)

[1.2.2 Test 2 8](#_Toc1363436755)

[1.2.3 Test 3 11](#_Toc1628933223)

[2. Multi-Process Testing 12](#_Toc401342786)

[2.1 Test Outline 12](#_Toc331289055)

[2.2 Test Case Descriptions 13](#_Toc2100779890)

[2.2.1 Test 1 13](#_Toc1320775756)

[2.2.2 Test 2 16](#_Toc1336565686)

[2.2.3 Test 3 18](#_Toc999330103)

# General Client Testing

## 1.1 Test Outline

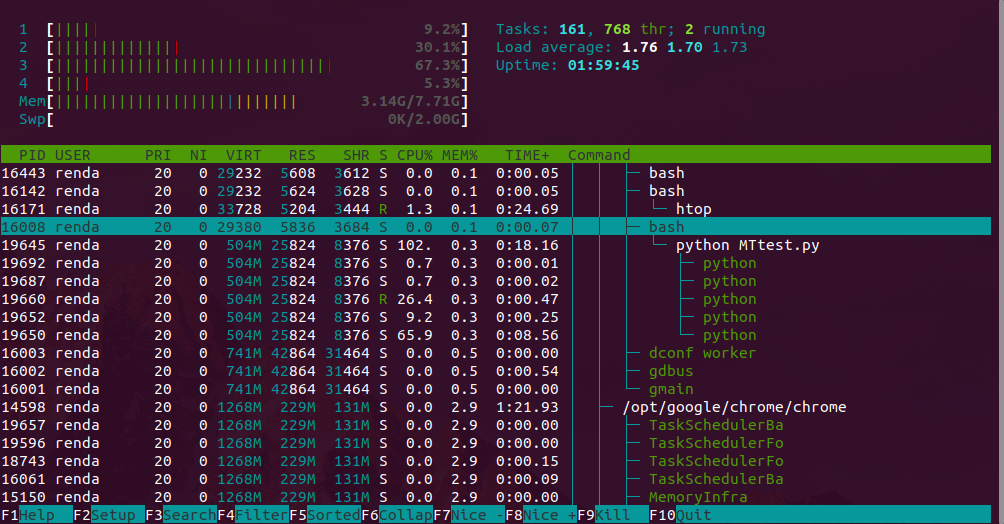
| Rule # | Test Description | Tool Used | Expected Results | Pass/Fail |
| --- | --- | --- | --- | --- |
| 1 | Program successfully connects to the multi-threaded server | Server/Client application | Server receives connection request from amount of clients. | Pass. Detailed results are attached. |
| 2 | Program successfully connects to the select server. | Server/Client application | Server receives connection request from amount of clients. | Pass. Detailed results are attached. |
| 3 | Program successfully connects to the epoll server. | Server/Client application | Server receives connection request from amount of clients. | Pass. Detailed results are attached. |
| 4 | Client creates processes  Server/Client equal to the number of cpu of the client host. | Server/Client application | In task manager processes = to the number of total cpus  on the PC and log file will show that. | Pass. Detailed results are attached. |
| 5 | Client creates the correct  amount of connections  specified by the user. | Server/Client application | Client set the total connections of one process, we can find the connection number by netstat command. | Pass. Detailed results are attached. |
| 6 | Client resent the data to server many times which is specified by the user. | Server/Client application | Client set the repeating times to resent the data to server, we can find from log file. | Pass. Detailed results are attached. |
| 7 | Client would sleep a while during sending work and the time interval is specified by the user. | Server/Client application | Client set the sleeping time of the gap between two sending data events, we can find the proof from log file. | Pass. Detailed results are attached. |
| 8 | Client shuts down after all child processes have  finished. | Server/Client application. | Client finishes and will exit. | Pass. Detailed results are attached. |
| 9 | Client print a brief report | Server/Client application. | When client close all connections will print a report and exit, we can find the report at the terminal. | Pass. Detailed results are attached. |

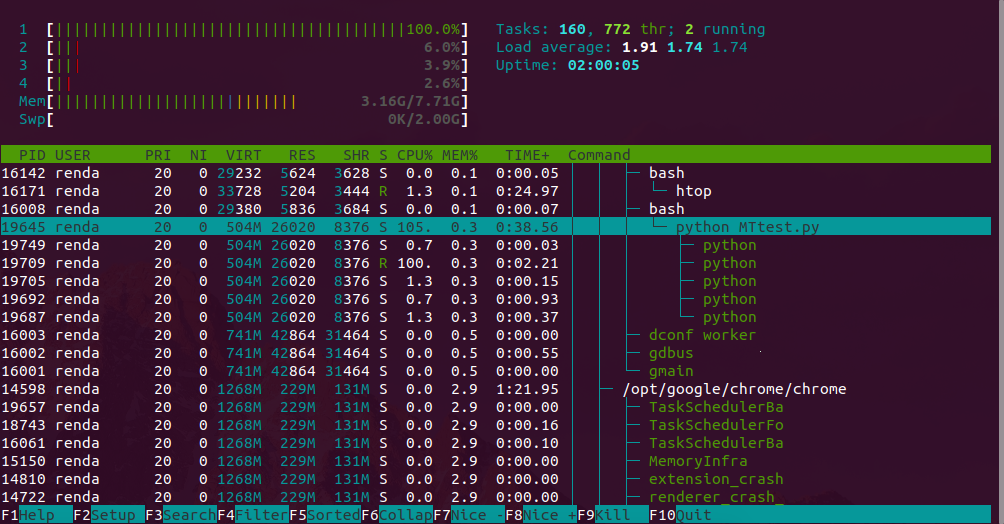
## 1.2 Test Case Descriptions

### 1.2.1 Test 1

This was a simple test for how about the performance the multi-threaded approach to do the CPU-bound job.

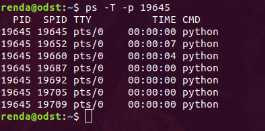
# **Run under Ubuntu OS:**





Data from command: htop

We can see that under the python MTtest.py process, there are some green python threads is running.



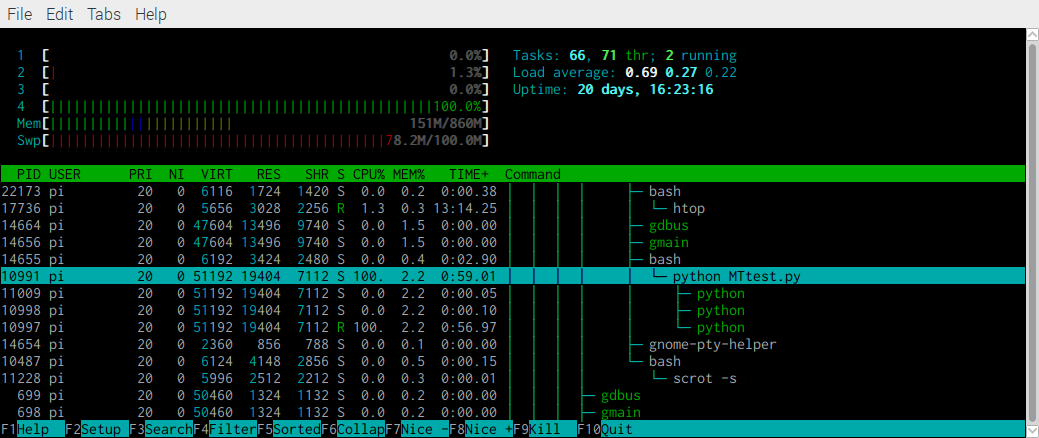
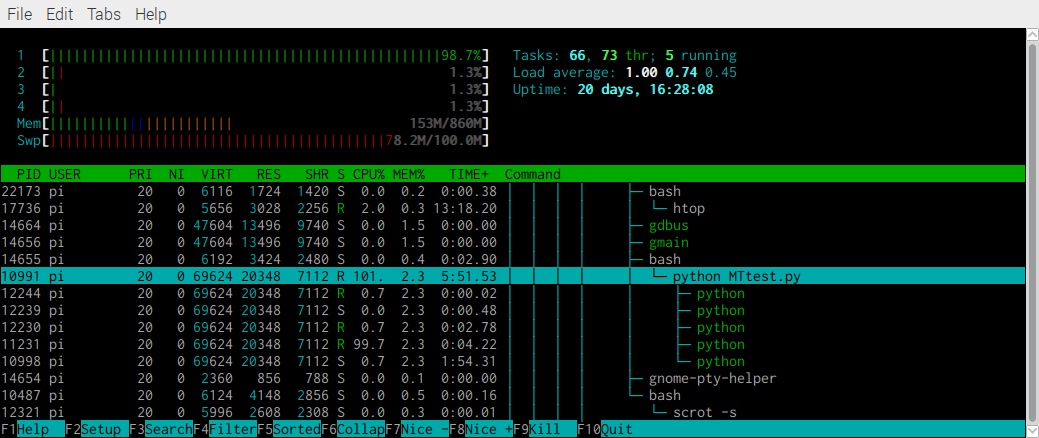
Data from command: ps

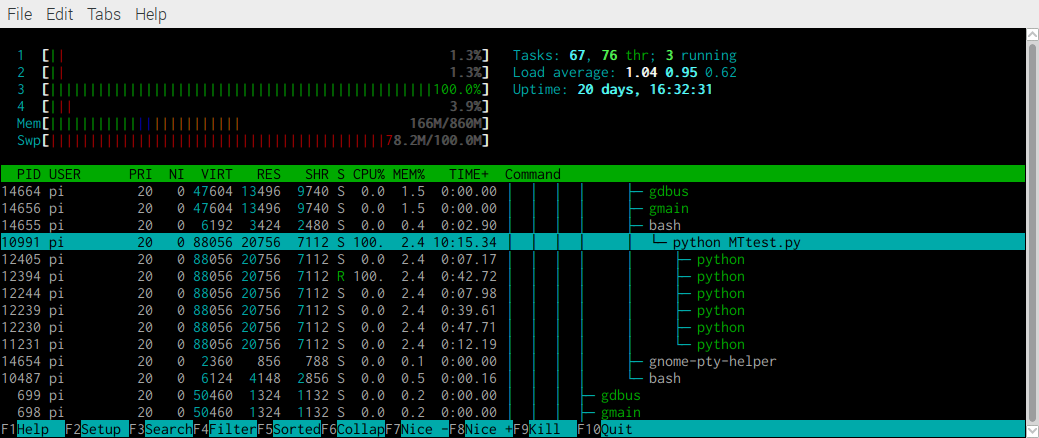
We can find that the threads under the parent process pid: 19645

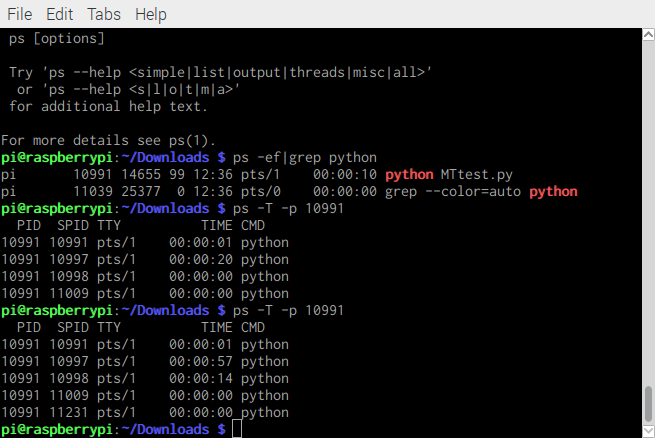
# **After the test:**

2018-01-23-024016_378x54_scrot

# **Run under Raspbian OS:**

****

****Data from command: htop

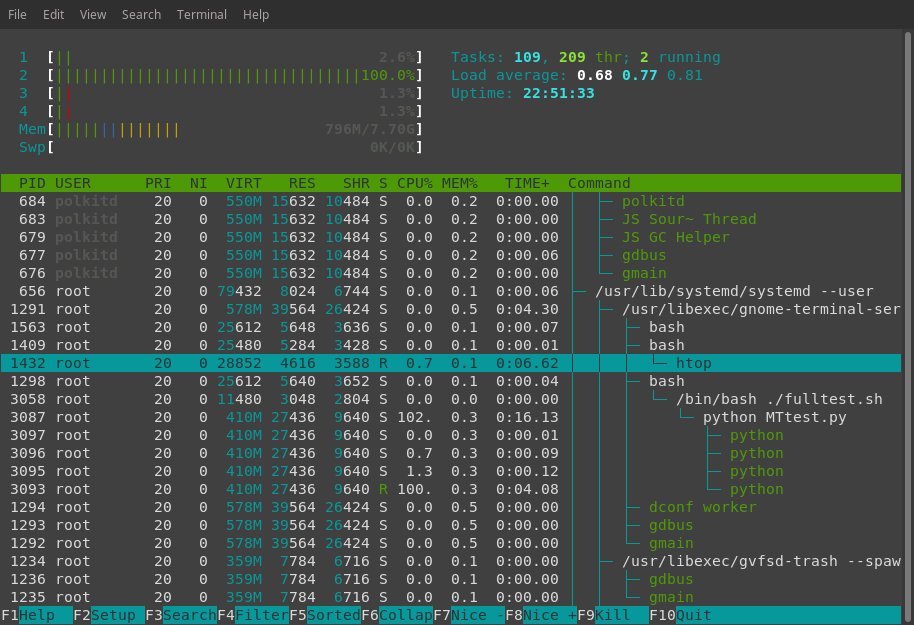


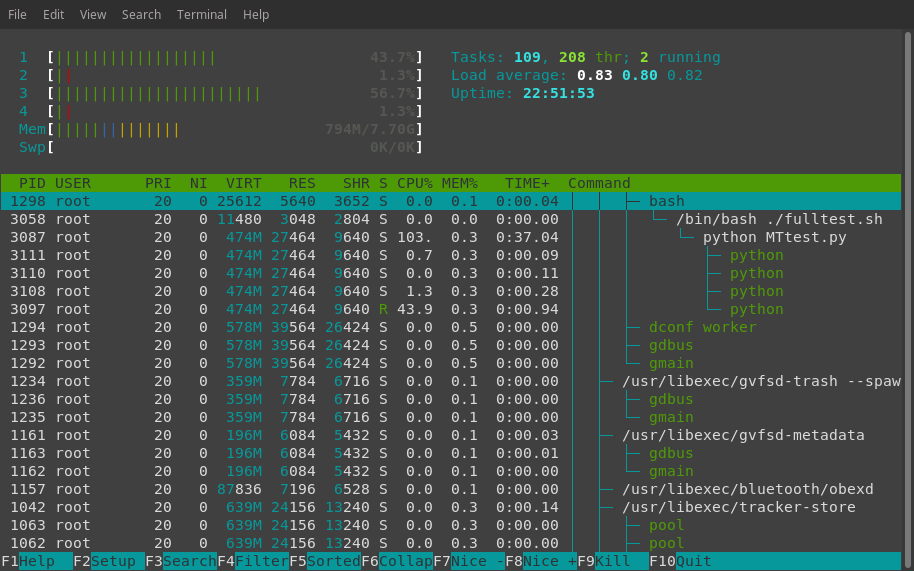
Data from command: ps

# **After the test:**



# **Run under Fedora OS:**





Data from command: htop

# **After the test:**

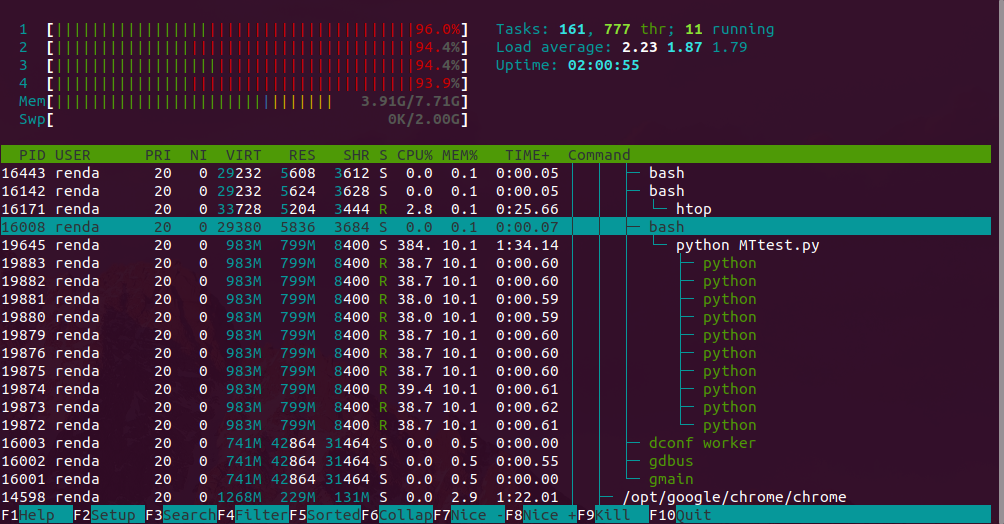


Test passes, all threads have done for CPU-bound test.

### 1.2.2 Test 2

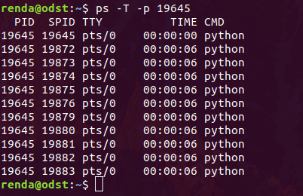
This was a simple test for how about the performance the multi-threaded approach to do the IO-bound job.

# **Run under Ubuntu OS:**



Data from command: htop

We can see that under the python MTtest.py process, there are 10 green python threads is running.



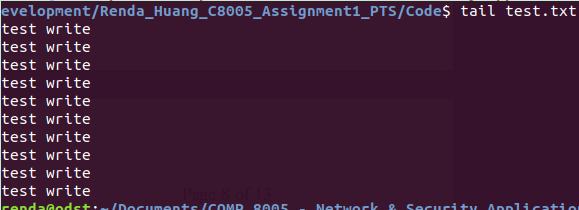
Data from command: ps

We can find that the threads under the parent process still pid: 19645

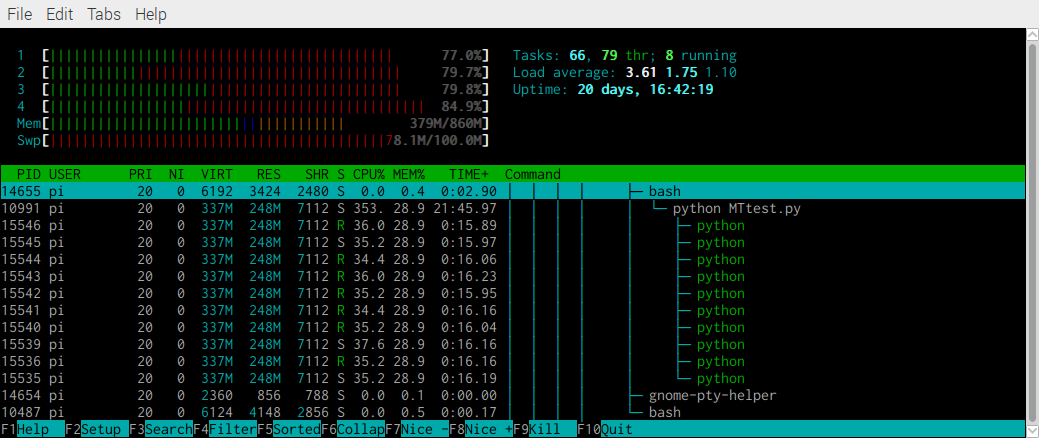
# **After the test:**

2018-01-23-030724_374x52_scrot

2018-01-23-030906_632x48_scrot

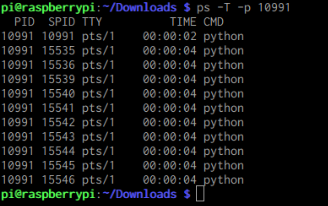


# **Run under Raspbian OS:**



Data from command: htop

We can see that under the python MTtest.py process, there are 10 green python threads is running.



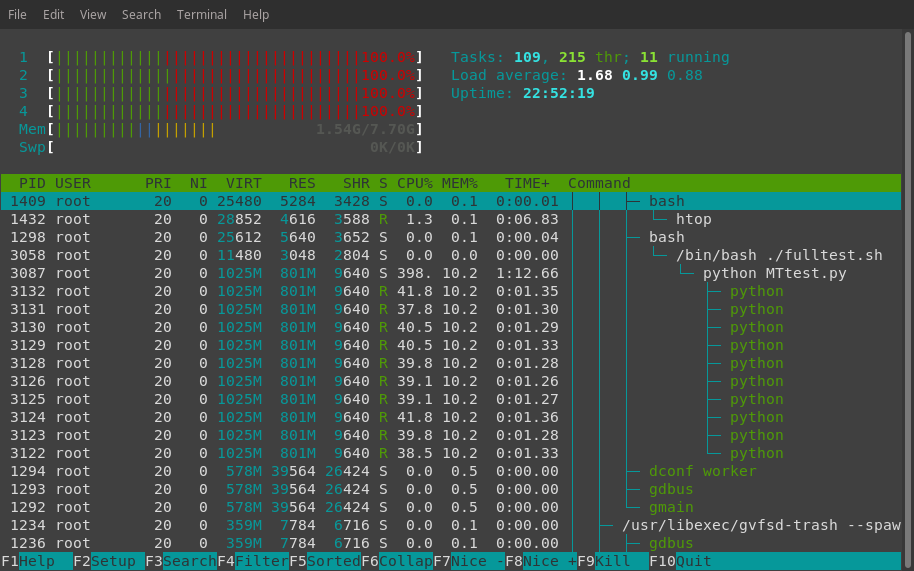
Data from command: ps

We can find that the threads under the parent process pid: 10991

# **After the test:**

# 

# **Run under Fedora OS:**



Data from command: htop

We can see that under the python MTtest.py process, there are 10 green python

# **After the test:**



Test passes, all threads have done for IO-bound test.

### 1.2.3 Test 3

This was a simple test for how about the performance the multi-threaded approach to send HTTP request to ask a web page.

# **Run under Ubuntu OS:**

2018-01-23-032549_490x58_scrot

# **Run under Raspbian OS:**



# **Run under Fedora OS:**



Test passes, tall threads have successfully request a web page.

# Multi-Thread Testing

## 2.1 Test Outline

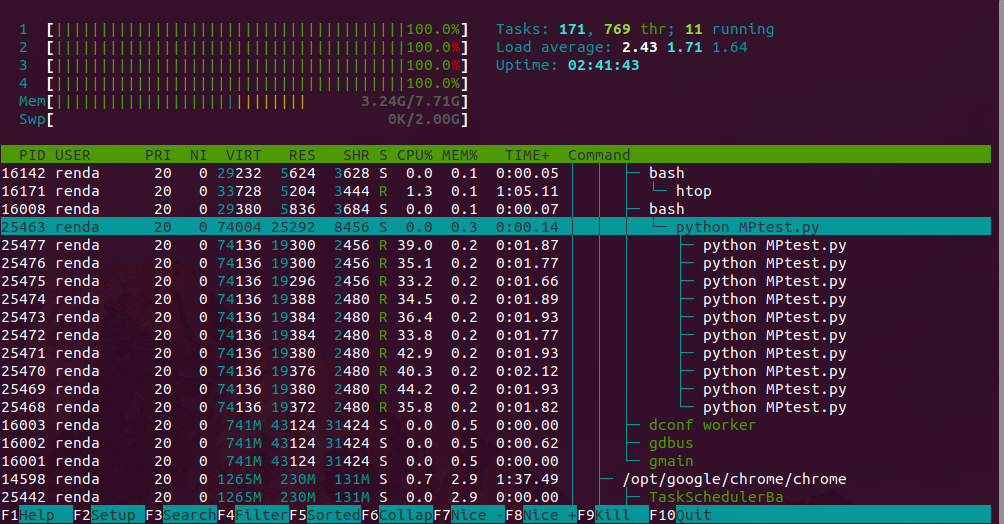
| Rule # | Test Description | Tool Used | Expected Results | Pass/Fail |
| --- | --- | --- | --- | --- |
| 1 | Program successfully receives connection request from clients. | Sever/Client  application. | When server receive connection request, it would picks it up. | Pass. Detailed results are attached. |
| 2 | Program successfully responds to the client | Sever/Client  application. | The byte size sent to Server. See  the server is the test case same size of message sent back. | Pass. Detailed results are attached. |
| 3 | Program successfully listens on the port user specified test | Sever/Client  application. | Change a port to start the service and make client side change the destination port number | Pass. Detailed results are attached. |
| 4 | Server logs print expected information | Sever/Client  application. | Log shows several level of log data. | Pass. Detailed results are attached. |

## 2.2 Test Case Descriptions

### 2.2.1 Test 1

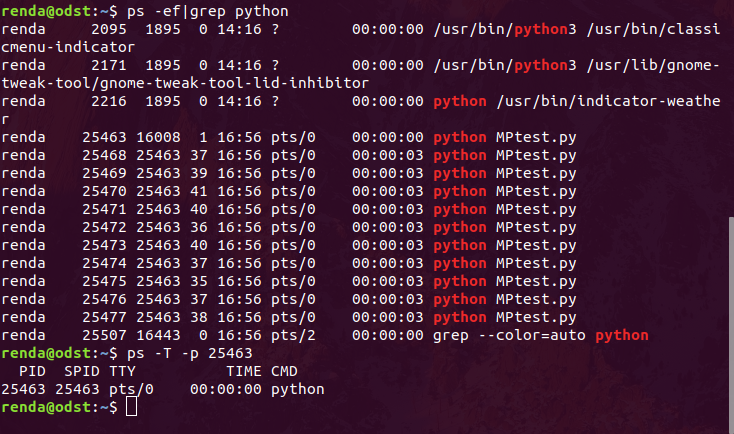
This was a simple test for how about the performance the multi-processed approach to do the CPU-bound job.

# **Run under Ubuntu OS:**



Data from command: htop

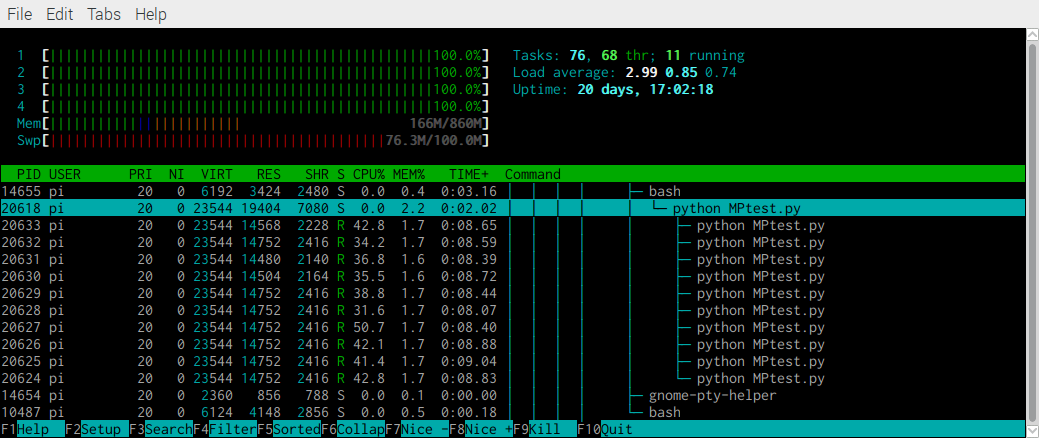
We can see that under the python MPtest.py process, there are 10 whit python Processes running.



# **After the test:**

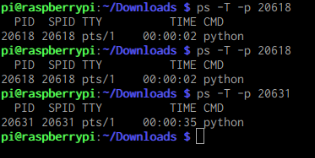
2018-01-23-033751_396x54_scrot

# **Run under Raspbian OS:**



Data from command: htop

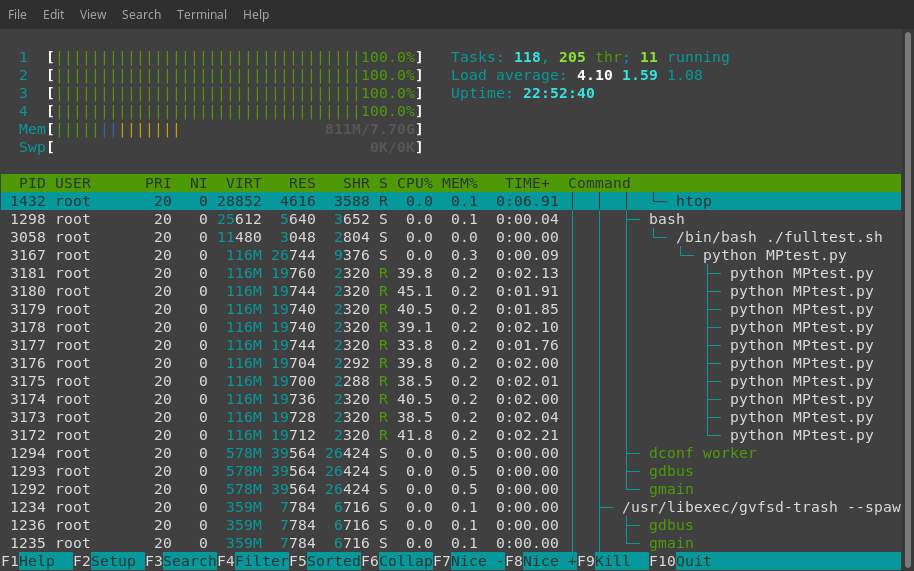
We can see that under the python MPtest.py process, there are 10 whit python Processes running.



# **After the test:**



# **Run under Fedora OS:**



Data from command: htop

We can see that under the python MPtest.py process, there are 10 whit python

# **After the test:**

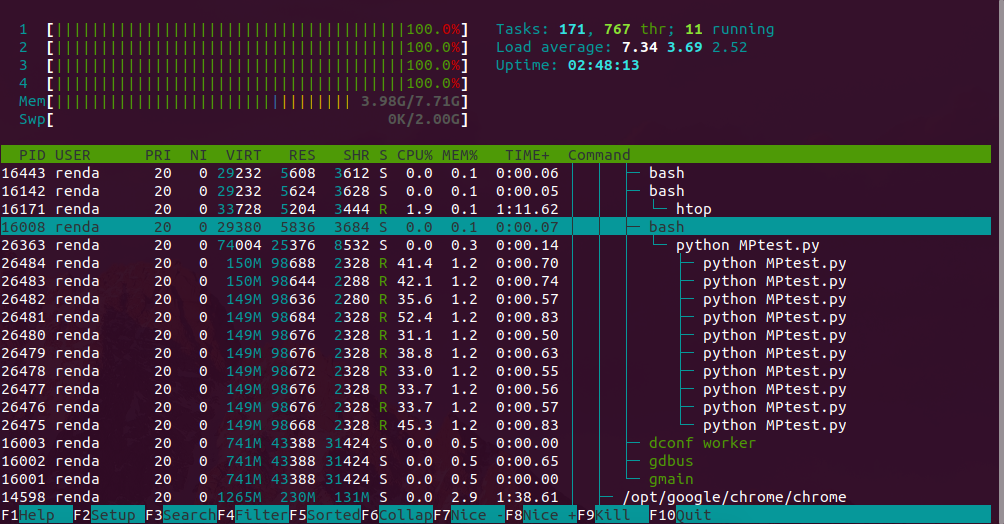


Test passes, all processes have done for CPU-bound test.

### 2.2.2 Test 2

This was a simple test for how about the performance the multi-processed approach to do the IO-bound job.

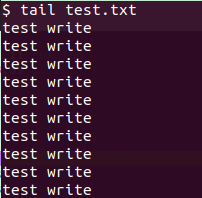
# **Run under Ubuntu OS:**



Data from command: htop

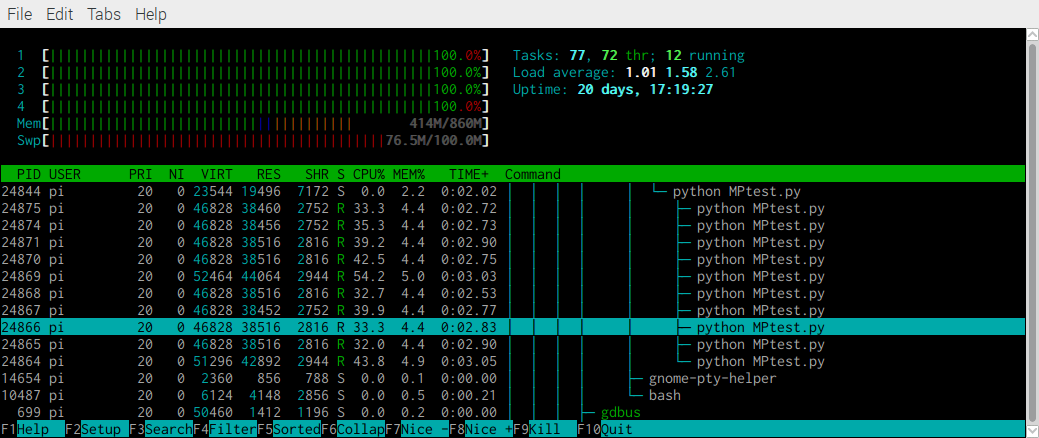
We can see that under the python MPtest.py process, there are 10 whit python Processes running.

# **After the test:**



2018-01-23-034802_218x44_scrot

# **Run under Raspbian OS:**

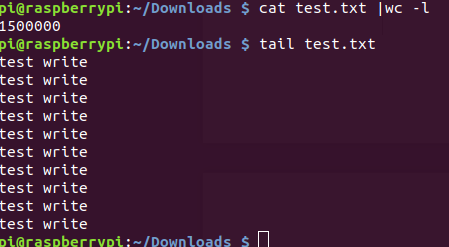


Data from command: htop

We can see that under the python MPtest.py process, there are 10 whit python Processes running.

# **After the test:**





# **Run under Fedora OS:**



Test passes, all processes have done for IO-bound test.

### 2.2.3 Test 3

This was a simple test for how about the performance the multi-processed approach to send HTTP request to ask a web page.

# **Run under Ubuntu OS:**

2018-01-23-035559_504x45_scrot

# **Run under Raspbian OS:**



# **Run under Fedora OS:**



Test passes, tall processes have successfully request a web page.

# Select Mode Server Testing

## 3.1 Test Outline

| Rule # | Test Description | Tool Used | Expected Results | Pass/Fail |
| --- | --- | --- | --- | --- |
| 1 | Program successfully receives connection request from clients. | Sever/Client  application. | When server receive connection request, it would picks it up. | Pass. Detailed results are attached. |
| 2 | Program successfully responds to the client | Sever/Client  application. | The byte size sent to Server. See  the server is the test case same size of message sent back. | Pass. Detailed results are attached. |
| 3 | Program successfully listens on the port user specified test | Sever/Client  application. | Change a port to start the service and make client side change the destination port number | Pass. Detailed results are attached. |
| 4 | Server logs print expected information | Sever/Client  application. | Log shows several level of log data. | Pass. Detailed results are attached. |

## 3.2 Test Case Descriptions

# Epoll Mode Server Testing

## 4.1 Test Outline

| Rule # | Test Description | Tool Used | Expected Results | Pass/Fail |
| --- | --- | --- | --- | --- |
| 1 | Program successfully receives connection request from clients. | Sever/Client  application. | When server receive connection request, it would picks it up. | Pass. Detailed results are attached. |
| 2 | Program successfully responds to the client | Sever/Client  application. | The byte size sent to Server. See  the server is the test case same size of message sent back. | Pass. Detailed results are attached. |
| 3 | Program successfully listens on the port user specified test | Sever/Client  application. | Change a port to start the service and make client side change the destination port number | Pass. Detailed results are attached. |
| 4 | Server logs print expected information | Sever/Client  application. | Log shows several level of log data. | Pass. Detailed results are attached. |

## 4.2 Test Case Descriptions