# Parallelization of Web Scraper Program

Pranav Vinod

University of Massachusetts, Dartmouth

DSC-520: High Performance Scientific Computing

Professor Alfa Heryudono

## ABSTRACT

This project implements a Web Scraping Program in a parallel fashion using Message Passing Interface technique which allows for access to multiple processors. This project has used Python libraries Selenium and MPI4PY to fetch information from the web and implement MPI technique respectively. Comparison between runtimes will be for serial implementation and parallel implementation using 2 and 4 cores.

## AIM

The aim of this project is to create a web scraping program and compare its performance on serial implementation and parallel implementation through the following parameters:

1. Speedup
2. Efficiency

The information is extracted about the most watched videos on www.youtube.com by visiting each video link and scraping the following information from the webpage:

1. Title of the video
2. Number of views
3. Number of likes on the video
4. Uploaded date

# INTRODUCTION

This project was programmed in Python language using its SELENIUM library for web scraping and MPI4PY for parallel implementation.

**Selenium** is a powerful tool for controlling web browsers through programs and performing browser automation. Through its array of functions, selenium allows users to create controlled step by step interactions with a webpage and access information from it through user commands. It is compatible with all major OS and browsers. Even though selenium allows for automation, for this project manual testing has been chosen.

**Mpi4py or MPI for Python** provides Python bindings for the Message Passing Interface (MPI) standard, allowing Python application to exploit multiple processors on workstations, clusters and supercomputers. MPI allows for message passing between multiple computers or as in the case of this project, multiple processors.

## IMPLEMENTATION

Web Scraper Program:

- The program visits a playlist on YouTube which contains the most viewed videos on YouTube until now and grabs the link for each video in the playlist and saves them in the form of a list.
- Then the driver for Google Chrome then iterates through that list, visiting each link in the list and extracting information about title, number of views, number of likes on the video and the uploaded date of the video in the form of a list.
- Since the information received from the video webpage contains extra information, that information is then removed from the item and by dropping certain columns from the above-mentioned list.
- Finally, the list containing information is added to a Pandas Dataframe.

This process is repeated for each video in the playlist and repeated for 3 more playlists.

Parallelization:

- To parallelize the web scraper program, each core is assigned a single playlist to scrape information from.
- Within each core, the web scraper program is implemented, and the data extracted is saved locally on the core in a Pandas Dataframe.

- At the end of every local run, information about runtime and the entire Dataframe is sent to core with rank = 0, where Dataframes from every core are appended to form a single Dataframe.
- The Dataframe is then saved in the form of a csv.

Set 1: Playlist 1

Set2: Playlist 1 , Playlist 2, Playlist 3, Playlist 4

## RESULTS

Speedup and Efficiency we calculated using the following formulas:

$S_p = T_s / T_p$ *where*

*$S_p$ : Speedup using p cores*

*$T_s$ : Runtime in serial implementation*

*$T_p$ : Runtime in parallel implementation*

*p : Number of Cores*

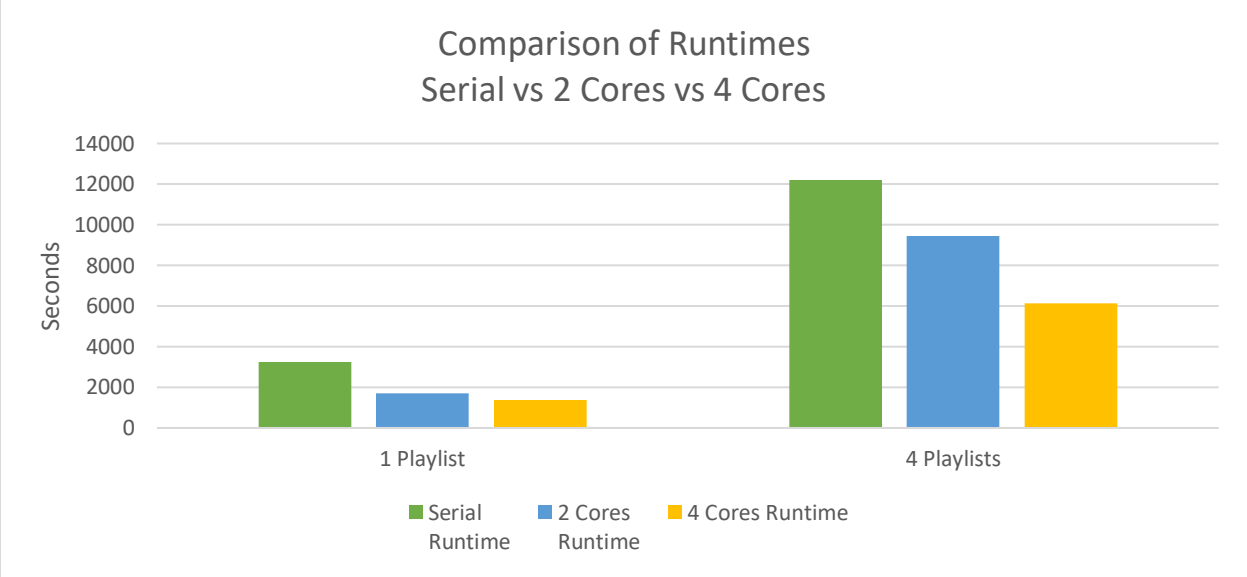
$E_p = S_p / P$ *where,*

*$E_p$ : Efficiency with P cores*

*$S_p$ : Speedup using p cores*

*P : Number of Cores*

| Configuration | Serial Runtime(s) | 2 Cores Runtime(s) | 4 Cores Runtime(s) |
|---|---|---|---|
| 1 Playlist | 3242.4 | 1718.5 | 1380.97 |
| 4 Playlists | 12187.7 | 9443.7 | 6134 |



| Config | 2 Cores Speedup | 2 Cores Efficiency | 4 Cores Speedup | 4 Cores Efficiency |
|---|---|---|---|---|
| 1 Playlist | 1.88 | 0.9433 | 2.347 | 0.5869 |
| 4 Playlists | 1.29 | 0.645 | 1.98 | 0.496 |

## CONCLUSION

From the data we can see that on parallelizing the program we achieve a significant speedup and efficiency. Even though speed up achieved is greater when using 4 cores, efficiency is greater when using 2 cores. This could be because of time taken to send and receive messages between cores which scales as the number of cores increases.

In addition to the hardware of the computer and the program algorithm itself, one of the factors affecting the time taken for each process is the speed and performance of the internet connection. Because of a slow internet connection, one might encounter unexpected results as with the case of lesser efficiency with greater number of cores.

In conclusion, it can safely be said that a web scraping program experiences significant speedup and efficiency on parallelization.

## REFERENCES


1. https://www.browserstack.com/guide/python-selenium-to-run-web-automation-test
2. https://www.geeksforgeeks.org/selenium-python-tutorial/amp/
3. https://mpi4py.readthedocs.io/en/stable

CODE: SERIAL

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time
import pandas as pd
import random

c=['Title','Likes','Views','Uploaded Date']
df = pd.DataFrame(columns=c)
rate = [i/10 for i in range(10)]
num=0
PATH = "/Users/pranavvinod/Documents/GitHub/HPSC-Project/chromedriver"
driver = webdriver.Chrome(PATH)
start = time.time()
driver.get("https://www.youtube.com/playlist?list=PLirAqAtl_h2r5g8xGajEwdXd3x1sZh8hC")

html = driver.find_element(By.TAG_NAME,"html")
for i in range(10):
    time.sleep(1)
    html.send_keys(Keys.PAGE_DOWN)

links = [x.get_attribute('href') for x in driver.find_elements(By.ID,"video-title")]
print(len(links))

for link in links[0:10]:
    try:
        print(num)
        num+=1
        driver.get(link)
```

```
        time.sleep(4)

        names = driver.find_elements(By.ID, "info")
        x = (names[2].text.split("\n"))
        if(len(x)>6):
            if(x[0][0]=='#'):
                x.pop(0)
        x.remove("DISLIKE")
        x.remove("SHARE")
        x.remove("SAVE")
        s = x[1].split(' views')
        for i in s:
            x.append(i)
        x.pop(1)
        # print(x)
        df.loc[len(df.index)]=x
    except:
        continue
# # print(df)
# # time.sleep(random.choice(rate))

# print(len(links))
# end = time.time()
# print("The time taken is: ",end-start)
print(df.head())
driver.quit()

#############################################################################################

driver = webdriver.Chrome(PATH)
driver.get("https://www.youtube.com/playlist?list=PLirAqAtl_h2o4xCWaBsDH3BKNQs8YqLCL")
html = driver.find_element_by_tag_name("html")
for i in range(95):
    time.sleep(1)
    html.send_keys(Keys.PAGE_DOWN)
links = [x.get_attribute('href') for x in driver.find_elements(By.ID,"video-title")]
```

```
for link in links:
    try:
        driver.get(link)
        time.sleep(4)

        names = driver.find_elements(By.ID, "info")
        x = (names[2].text.split("\n"))
        if(len(x)>6):
            if(x[0][0]=='#'):
                x.pop(0)
        x.remove("DISLIKE")
        x.remove("SHARE")
        x.remove("SAVE")
        s = x[1].split(' views')
        for i in s:
            x.append(i)
        x.pop(1)
        # print(x)
        df.loc[len(df.index)]=x
    except:
        continue

print(len(links))
driver.quit()

# ################################################################################

driver = webdriver.Chrome(PATH)
driver.get("https://www.youtube.com/playlist?list=PLirAqAtl_h2p57Njt3QJDtwxAPZENJrIp")
html = driver.find_element_by_tag_name("html")
for i in range(95):
    time.sleep(1)
    html.send_keys(Keys.PAGE_DOWN)
links = [x.get_attribute('href') for x in driver.find_elements(By.ID,"video-title")]
for link in links:
    try:
```

```
        driver.get(link)
        time.sleep(4)

        names = driver.find_elements(By.ID, "info")
        x = (names[2].text.split("\n"))
        if(len(x)>6):
            if(x[0][0]=='#'):
                x.pop(0)
        x.remove("DISLIKE")
        x.remove("SHARE")
        x.remove("SAVE")
        s = x[1].split(' views')
        for i in s:
            x.append(i)
        x.pop(1)
        df.loc[len(df.index)]=x
    except:
        continue

print(len(links))
driver.quit()

# ################################################################################################

driver = webdriver.Chrome(PATH)
driver.get("https://www.youtube.com/playlist?list=PLirAqAtl_h2rTbOXU2Oc-7WBBHmFrnyUC")
html = driver.find_element_by_tag_name("html")
for i in range(95):
    time.sleep(5)
    html.send_keys(Keys.PAGE_DOWN)
links = [x.get_attribute('href') for x in driver.find_elements(By.ID,"video-title")]
for link in links:
    try:
        driver.get(link)
        time.sleep(4)

        names = driver.find_elements(By.ID, "info")
```

```
        x = (names[2].text.split("\n"))
        if(len(x)>6):
            if(x[0][0]=='#'):
                x.pop(0)
        x.remove("DISLIKE")
        x.remove("SHARE")
        x.remove("SAVE")
        s = x[1].split(' views')
        for i in s:
            x.append(i)
        x.pop(1)
        df.loc[len(df.index)]=x
    except:
        continue


print(len(links))
driver.quit()
end = time.time()
print('Time taken is: ',end-start)
df.to_csv("most_views_serial.csv")

#########################################################################################
###################
```

CODE : PARALLEL

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time
from mpi4py import MPI
import pandas as pd

PATH = "/Users/pranavvinod/Documents/GitHub/HPSC-Project/chromedriver"
comm = MPI.COMM_WORLD
rank = comm.Get_rank()
num=0
c=['Title','Likes','Views','Uploaded Date']
df = pd.DataFrame(columns=c)
df1 = pd.DataFrame(columns=c)
df2 = pd.DataFrame(columns=c)
df3 = pd.DataFrame(columns=c)


if rank==0:
    start = time.time()
    driver = webdriver.Chrome(PATH)
    driver.get("https://www.youtube.com/playlist?list=PLirAqAtl_h2r5g8xGajEwdXd3x1sZh8hC")
    html = driver.find_element_by_tag_name("html")
    for i in range(95):
        time.sleep(1)
        html.send_keys(Keys.PAGE_DOWN)

    links = [x.get_attribute('href') for x in driver.find_elements(By.ID,"video-title")]
    print(len(links))
    for link in links:
        try:
            num+=1
            driver.get(link)
            time.sleep(4)
```

```
        names = driver.find_elements(By.ID, "info")
        x = (names[2].text.split("\n"))
        if(len(x)>6):
            if(x[0][0]=='#'):
                x.pop(0)
        x.remove("DISLIKE")
        x.remove("SHARE")
        x.remove("SAVE")
        s = x[1].split(' views')
        for i in s:
            x.append(i)
        x.pop(1)
        df.loc[len(df.index)]=x
    except:
        continue

print(df.head())
end = time.time()
t1 = end - start
t2 = comm.recv(source = 1, tag=1)
t3 = comm.recv(source=2,tag=2)
t4 = comm.recv(source=3,tag=3)
t= max(t1,t2,t3,t4)
n2 = comm.recv(source = 1, tag=4)
n3 = comm.recv(source=2,tag=5)
n4 = comm.recv(source=3,tag=6)
d1 = comm.recv(source=1,tag=7)
df1 = df1.append(d1)
d2 = comm.recv(source=2,tag=8)
df1 = df1.append(d2)
d3 = comm.recv(source=3,tag=9)
df1 = df1.append(d3)
n= num+n2+n3+n4
df1.to_csv("most_views_parallel")

print("The total titles are: ",n)
```

```
    print("The time taken is: ",t)

#############################################################################################


if rank==1:
    start = time.time()
    driver = webdriver.Chrome(PATH)
    driver.get("https://www.youtube.com/playlist?list=PLirAqAtl_h2o4xCWaBsDH3BKNQs8YqLCL")
    html = driver.find_element_by_tag_name("html")
    for i in range(95):
        time.sleep(1)
        html.send_keys(Keys.PAGE_DOWN)
    links = [x.get_attribute('href') for x in driver.find_elements(By.ID,"video-title")]
    print(len(links))

    for link in links:
        try:
            num+=1
            driver.get(link)
            time.sleep(4)

            names = driver.find_elements(By.ID, "info")
            x = (names[2].text.split("\n"))
            if(len(x)>6):
                if(x[0][0]=='#'):
                    x.pop(0)
            x.remove("DISLIKE")
            x.remove("SHARE")
            x.remove("SAVE")
            s = x[1].split(' views')
            for i in s:
                x.append(i)
            x.pop(1)
            # print(x)
            df1.loc[len(df1.index)]=x
        except:
```

```
            continue

    print(df1.head())
    driver.quit()
    end = time.time()
    t2 = end-start
    comm.send(t2,dest = 0,tag = 1)
    comm.send(num,dest=0,tag=4)
    comm.send(df1,dest=0,tag=7)


 ##########################################################################################




if rank==2:
    start = time.time()
    driver = webdriver.Chrome(PATH)
    driver.get("https://www.youtube.com/playlist?list=PLirAqAtl_h2p57Njt3QJDtwxAPZENJrIp")
    html = driver.find_element_by_tag_name("html")
    for i in range(95):
        time.sleep(1)
        html.send_keys(Keys.PAGE_DOWN)

    links = [x.get_attribute('href') for x in driver.find_elements(By.ID,"video-title")]
    print(len(links))

    for link in links:
        try:
            num+=1
            driver.get(link)
            time.sleep(4)

            names = driver.find_elements(By.ID, "info")
            x = (names[2].text.split("\n"))
            if(len(x)>6):
```

```
            if(x[0][0]=='#'):
                x.pop(0)
        x.remove("DISLIKE")
        x.remove("SHARE")
        x.remove("SAVE")
        s = x[1].split(' views')
        for i in s:
            x.append(i)
        x.pop(1)
        # print(x)
        df2.loc[len(df2.index)]=x
    except:
        continue
print(df2.head())
driver.quit()
end = time.time()
t3 = end-start
comm.send(t3,dest = 0,tag = 2)
comm.send(num,dest=0,tag=5)
comm.send(df2,dest=0,tag=8)

##########################################################################################



if rank==3:
    start = time.time()
    driver = webdriver.Chrome(PATH)
    driver.get("https://www.youtube.com/playlist?list=PLirAqAtl_h2rTbOXU2Oc-7WBBHmFrnyUC")
    html = driver.find_element_by_tag_name("html")
    for i in range(95):
        time.sleep(1)
        html.send_keys(Keys.PAGE_DOWN)

    links = [x.get_attribute('href') for x in driver.find_elements(By.ID,"video-title")]
    print(len(links))
```

```
    for link in links:
        try:
            num+=1
            driver.get(link)
            time.sleep(4)

            names = driver.find_elements(By.ID, "info")
            x = (names[2].text.split("\n"))
            if(len(x)>6):
                if(x[0][0]=='#'):
                    x.pop(0)
            x.remove("DISLIKE")
            x.remove("SHARE")
            x.remove("SAVE")
            s = x[1].split(' views')
            for i in s:
                x.append(i)
            x.pop(1)
            print(x)
            df3.loc[len(df3.index)]=x
        except:
            continue

    print(df3.head())
    driver.quit()
    end = time.time()
    t4 = end-start
    comm.send(t4,dest = 0,tag = 3)
    comm.send(num,dest=0,tag=6)
    comm.send(df3,dest=0,tag=9)

MPI.Finalize()
```