MCIS 6263 – Big Data Assignment 3 : MapReduce

*Name :**Saranya Balasubramaniyan***

*Student ID : **999901316***

Contents

MapReduce

MapReduce is a programming model for processing large data sets with a parallel, distributed algorithm on a cluster. MapReduce facilitates concurrent processing by splitting petabytes of data into smaller chunks and processing them in parallel on Hadoop commodity servers.

Objective

The objective of the assignment is to show the difference between single-processor vs multi-processor runs, when applying the Map-Reduce model. For this purpose, a python program to find the longest string is executed in the Oracle virtual box environment with 4 processor and 5GB data setting.

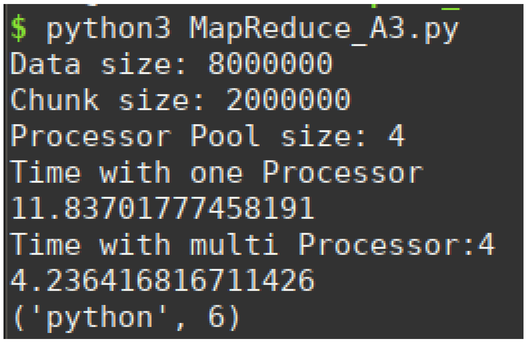
Laptop Specifications

Processor: Intel® Core™ i5 – 8250U CPU @1.60GHz 1.80GHz
Installed Memory : 8 GB
System Type: 64 bit operating system, x64- based processor

Cores: 4
Logical Processors : 8
Base Speed: 1.80GHz

Case 1

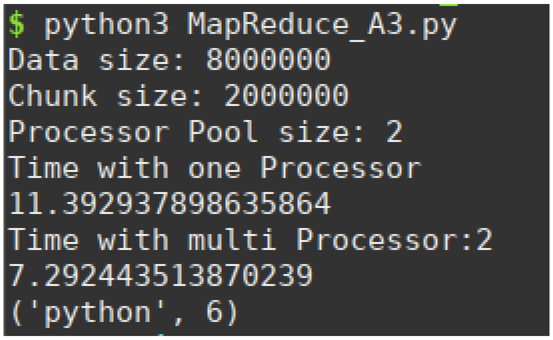
The use case shows the time taken to process the data with one processor and the time with 4 processors. We can see that the time taken has reduced by around 64%



```
$ python3 MapReduce_A3.py
Data size: 8000000
Chunk size: 2000000
Processor Pool size: 4
Time with one Processor
11.83701777458191
Time with multi Processor:4
4.236416816711426
('python', 6)
```

Case 2

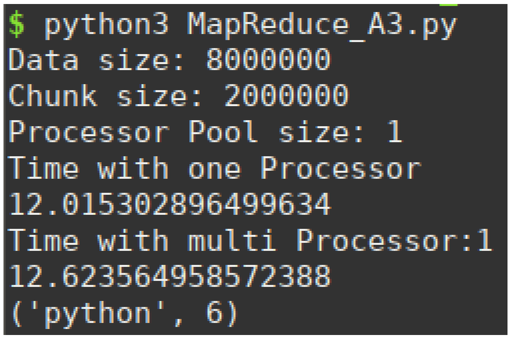
The use case shows the time taken to process the data with one processor and the time with 2 processors. We can see that the time taken has reduced by around 36%



```
$ python3 MapReduce_A3.py
Data size: 8000000
Chunk size: 2000000
Processor Pool size: 2
Time with one Processor
11.392937898635864
Time with multi Processor:2
7.292443513870239
('python', 6)
```

Case 3

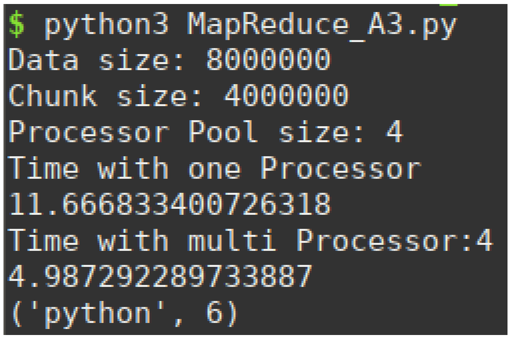
The use case shows the time taken to process the data with one processor and the time with 1 processor assigned in the multiple processor assignment. We can see that the time taken has not much variation.



```
$ python3 MapReduce_A3.py
Data size: 8000000
Chunk size: 2000000
Processor Pool size: 1
Time with one Processor
12.015302896499634
Time with multi Processor:1
12.623564958572388
('python', 6)
```

Case 4

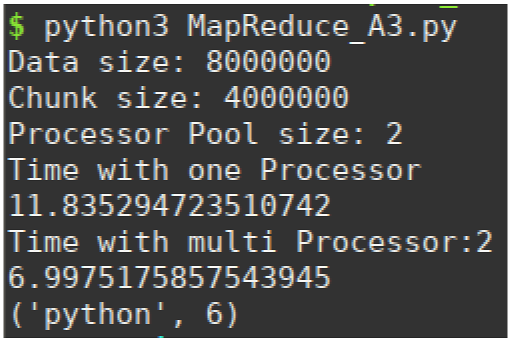
The use case shows the time taken to process the data with one processor and the time with 4 processors with an increase in the chunk size. We can see that the time taken has reduced by around 64%



```
$ python3 MapReduce_A3.py
Data size: 8000000
Chunk size: 4000000
Processor Pool size: 4
Time with one Processor
11.666833400726318
Time with multi Processor:4
4.987292289733887
('python', 6)
```

Case 5

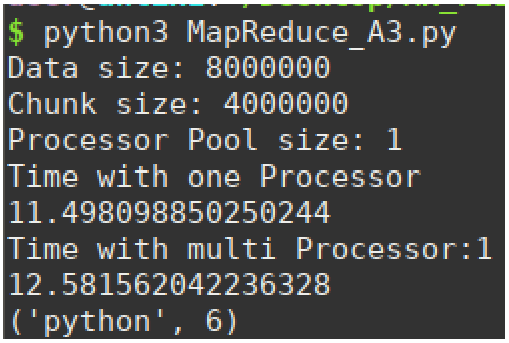
The use case shows the time taken to process the data with one processor and the time with 2 processors. We can see that the time taken has reduced by around 36%



```
$ python3 MapReduce_A3.py
Data size: 8000000
Chunk size: 4000000
Processor Pool size: 2
Time with one Processor
11.835294723510742
Time with multi Processor:2
6.9975175857543945
('python', 6)
```

Case 6

The use case shows the time taken to process the data with one processor and the time with 1 processor assigned in the multiple processor assignment with an increase in the chunk size. We can see that the time taken has not much variation.



```
$ python3 MapReduce_A3.py
Data size: 8000000
Chunk size: 4000000
Processor Pool size: 1
Time with one Processor
11.498098850250244
Time with multi Processor:1
12.581562042236328
('python', 6)
```