Experiment 6

Aim : Implement Matrix Multiplication and Word Frequency Count using Map Reduce

Theory :

BDT Experiment 6 DATE:

Kreena Shah
60004210243
C'32

Aim : Implement matrix multiplication & word frequency count using map reduce.

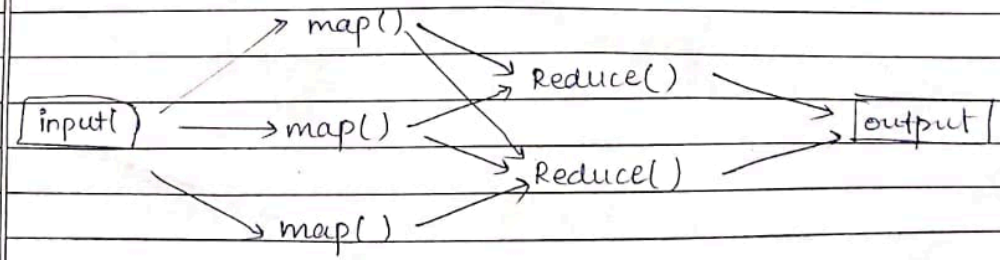
Theory :

Map Reduce is a programming model within the Hadoop Framework that is used to access big data stored in Hadoop File System.

Map Reduce facilitates concurrent processing by splitting petabyte of data into smaller chunks, & processing them in parallel on Hadoop Commodity servers.

Map Reduce is a paradigm which has two phases :

- The Map function takes input from the disk as <Key, value> pairs, processes them & produces another set of intermediate <Key, value> pairs as output
- The Reduce function also takes inputs as <Key, value> pairs & produces <Key, value> pairs as output.



Conclusion : Thus we understood & implemented map reduce & implemented matrix multiplication & map reduce functionality.

wordFreqCount.py

```python
import re

file = open("wordFreqCount.txt", 'r')
sen_dict_list = []
count_dic = {}

# Mapper
for line in file:
    line = re.sub("\.|\;|\(|\)| \:", " ", line.lower())
    wordlist = line.split()
    dic = {}
    for word in wordlist:
        dic[word] = dic.get(word, 0) + 1
    sen_dict_list.append(dic)

for dic in sen_dict_list:
    for key, val in dic.items():
        print(key, val)
    print()

#Reducer
for dic in sen_dict_list:
    for word, count in dic.items():
        count_dic[word] = count_dic.get(word, 0) + 1

key_list = sorted(dic.keys())
print("Final frequency count: ")
for key in key_list:
    print(key, " : ", count_dic[key])
```

wordFreqCount.txt

```
cow dog cat
mouse cow dog
rat rabbit mouse cow
```

```
PS C:\Users\Admin\OneDrive\Desktop\sem6\BDI\Pracs> py .\wordFreqCount.py
cow 1
dog 1
cat 1

mouse 1
cow 1
dog 1

rat 1
rabbit 1
mouse 1
cow 1

Final frequency count:
cow  : 3
mouse  : 2
rabbit  : 1
rat  : 1
```

matrixMul.py

```python
print("Enter the size of matrix X (rows columns):")
X_rows, X_cols = map(int, input().split())
print("Enter the size of matrix Y (rows columns):")
Y_rows, Y_cols = map(int, input().split())
print("Enter elements of matrix X :")
X = []
for i in range(X_rows):
    row = list(map(int, input().split()))
    X.append(row)
print("Enter elements of matrix Y :")
Y = []
for i in range(Y_rows):
    row = list(map(int, input().split()))
    Y.append(row)
result = [[0 for _ in range(Y_cols)] for _ in range(X_rows)]

def mapper(i, j):
    partial_result = 0
    for k in range(len(Y)):
        partial_result += X[i][k] * Y[k][j]
    return (i, j, partial_result)

def reducer(results):
    for i, j, partial_result in results:
        result[i][j] = partial_result

mapped_values = [mapper(i, j) for i in range(X_rows) for j in range(Y_cols)]
reducer(mapped_values)
print("Result of matrix multiplication:")
for r in result:
    print(r)
```

```
PS C:\Users\Admin\OneDrive\Desktop\sem6\BDI\Pracs> py .\matrixMul.py
Enter the size of matrix X (rows columns):
3 2
Enter the size of matrix Y (rows columns):
2 2
Enter elements of matrix X :
1 2
2 1
3 4
Enter elements of matrix Y :
1 2
1 3
Result of matrix multiplication:
[3, 8]
[3, 7]
[7, 18]
```

Conclusion : Thus, we implemented Matrix Multiplication and Word Frequency Count using Map Reduce.