Task 별 구현 설명

Task 1. 그래프에서 중복된 간선과 loop을 모두 제거하는 기능을 MapReduce로 구현하기

Mapper에서는 value로 들어오는 edge에 대하여 노드 번호가 작은 노드가 앞에 오도록 재구성하여 write 해주었다. 이 과정에서 노드 번호가 같은 경우는 loop이므로 write 하지 않고, 노드 번호가 작은 노드를 앞에 오도록 하면서 중복 에지가 제거되는 효과가 생긴다.

Reducer에서는 들어온 key를 그대로 write 해준다.

Driver에서는 Mapper의 output key가 IntPairWritable 형태이므로 partitioner를 구현하여 적절한 reducer로 들어갈 수 있도록 해준다. [그림1]과 [그림2]를 통해 partitioner를 구현해주어야 각 reducer 별로 적절한 양의 일이 분배된다.

```
C:\Users\rosy0\eclipse-workspace\triangle_count\src\test\resources\wiki-topcats.txt.out>find /v /c "" part-r-00000

---------- PART-R-00000: 10364968

C:\Users\rosy0\eclipse-workspace\triangle_count\src\test\resources\wiki-topcats.txt.out>find /v /c "" part-r-00001

---------- PART-R-00001: 8058496

C:\Users\rosy0\eclipse-workspace\triangle_count\src\test\resources\wiki-topcats.txt.out>find /v /c "" part-r-00002

---------- PART-R-00002: 7754119
```

[그림1. Partitioner 구현 전의 output file별 line 수]

```
C:\Users\rosy0\eclipse-workspace\triangle_count\src\test\resources\wiki-topcats.txt.out>find /v /c "" part-r-00000

---------- PART-R-00000: 8483115

C:\Users\rosy0\eclipse-workspace\triangle_count\src\test\resources\wiki-topcats.txt.out>find /v /c "" part-r-00001

---------- PART-R-00001: 8481318

C:\Users\rosy0\eclipse-workspace\triangle_count\src\test\resources\wiki-topcats.txt.out>find /v /c "" part-r-00002

---------- PART-R-00002: 8479774
```

[그림2. Partitioner 구현 후의 output file별 line 수]

Task 2. 간선 (u, v)에 대해서 다음 조건에 따라 u와 v의 순서를 변경

Mapper1과 Reducer1은 Task1과 같다.

Mapper2에서는 중복과 loop가 제거된 edge들을 입력으로 받은 후, 1. key = 앞 노드, value = edge, 2. key = 뒷 노드, value = edge 형태로 두 번 write 해준다.

Reducer2에서는 key로 들어온 노드의 degree를 계산해준다. 그 후 iterable 형태로 들어온 value들 각각을 key로 하고, 계산한 degree를 value에 붙여서 write 해준다. 이렇게 하면 edge의 한 쪽 노드의 degree 계산이 완료된 것이다.

Mapper3은 입력을 그대로 write 해준다.

Reducer3은 Reducer2에서 계산하지 않은 반대편 노드의 degree를 계산하고 edge의 두 노드의 degree 값을 비교하여 degree 값이 작은 노드를 왼쪽, 큰 노드를 오른쪽에 두어 write 한다. 같은 경우에는 노드 번호가 작은 애를 왼쪽, 큰 애를 오른쪽에 두어 write 한다.

Task 3. 삼각형 MapReduce 알고리즘에 1) Task 1의 결과 그래프와 2) Task 2의 결과 그래프를 각각 입력했을 때, 성능 비교하기

삼각형 구하기 MapReduce의 경우 기존의 코드에서 약간 수정하여 사용하였다.

Mapper1의 경우 기존에는 노드 번호를 기준으로 edge를 재구성하여 write하던 것을, 이미 Task2에서 edge는 degree에 맞게 재구성하였으므로 들어온 key의 첫 번째 노드를 output key로, 두 번째 노드를 output value로 하여 write 해주었다.

Reducer1에서는 wedge를 만들어주었다. Wedge를 만들 땐 중심이 되는 노드와 연결된 두 노드를 노드 번호가 작은 애를 왼쪽, 큰 애를 오른쪽에 오도록 하여 write 해주었다. 이 순서는 이후에 edge 순서와 같아야 한다.

MapperForEdges에서는 다시 텍스트 파일로부터 edge를 읽어 Reducer1에서 write한 wedge와 순서가 맞도록 edge를 재구성한다. 시간이 없어 실제로 측정해보지는 못하였지만, MapperForEdges의 입력을 wiki-topcats.txt 파일이 아닌 Task1의 결과물 파일을 binary 파일로 저장하여 이를 받아오면, 중복과 loop이 제거되므로(용량이 거의 절반으로 줄어듦) 더욱 빠르게 map 함수를 실행할 수 있을 것 같다.

MapperForWedges에서는 들어온 key, value를 그대로 다시 write 해준다.

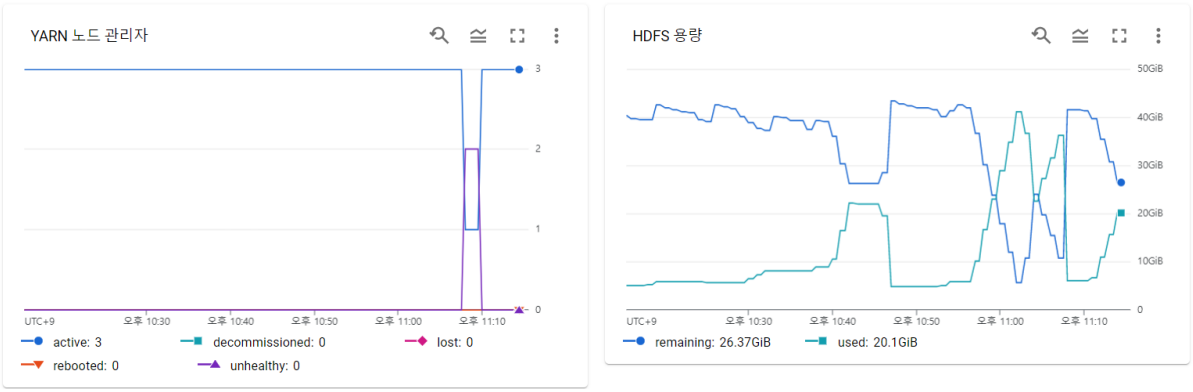
마지막으로 Reducer2에서는 들어온 value들에 -1이 있고(edge가 있고), wedge의 중심도 있다면 삼각형이 이루어지는 것이므로 삼각형을 write 해준다.

Task3 결과

- wedge의 수

1) 입력이 Task 1

1-1) wiki-topcats.txt 이용

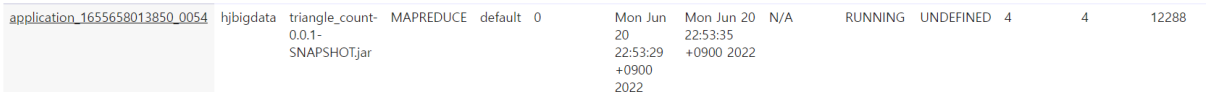


[그림 3]

```
2022-06-20 14:02:38,270 INFO mapreduce.Job:  map 100% reduce 23%
2022-06-20 14:03:02,350 INFO mapreduce.Job:  map 100% reduce 46%
2022-06-20 14:03:03,353 INFO mapreduce.Job:  map 100% reduce 68%
2022-06-20 14:05:04,696 INFO mapreduce.Job:  map 100% reduce 69%
2022-06-20 14:07:07,037 INFO mapreduce.Job:  map 33% reduce 23%
2022-06-20 14:07:10,046 INFO mapreduce.Job:  map 0% reduce 0%
2022-06-20 14:07:27,092 INFO mapreduce.Job:  map 5% reduce 0%
2022-06-20 14:07:33,107 INFO mapreduce.Job:  map 9% reduce 0%
2022-06-20 14:07:39,123 INFO mapreduce.Job:  map 13% reduce 0%
2022-06-20 14:07:45,140 INFO mapreduce.Job:  map 17% reduce 0%
2022-06-20 14:07:51,156 INFO mapreduce.Job:  map 21% reduce 0%
2022-06-20 14:07:57,171 INFO mapreduce.Job:  map 22% reduce 0%
2022-06-20 14:08:13,213 INFO mapreduce.Job:  map 33% reduce 0%
2022-06-20 14:08:32,270 INFO mapreduce.Job:  map 39% reduce 0%
2022-06-20 14:08:38,285 INFO mapreduce.Job:  map 43% reduce 0%
2022-06-20 14:08:44,301 INFO mapreduce.Job:  map 47% reduce 0%
2022-06-20 14:08:50,319 INFO mapreduce.Job:  map 52% reduce 0%
2022-06-20 14:08:56,334 INFO mapreduce.Job:  map 55% reduce 0%
2022-06-20 14:09:02,349 INFO mapreduce.Job:  map 56% reduce 0%
2022-06-20 14:09:19,395 INFO mapreduce.Job:  map 67% reduce 0%
2022-06-20 14:09:31,429 INFO mapreduce.Job:  map 72% reduce 7%
2022-06-20 14:09:32,431 INFO mapreduce.Job:  map 72% reduce 15%
2022-06-20 14:09:34,436 INFO mapreduce.Job:  map 72% reduce 22%
2022-06-20 14:09:37,443 INFO mapreduce.Job:  map 76% reduce 22%
2022-06-20 14:09:43,461 INFO mapreduce.Job:  map 80% reduce 22%
2022-06-20 14:09:49,475 INFO mapreduce.Job:  map 84% reduce 22%
2022-06-20 14:09:55,490 INFO mapreduce.Job:  map 88% reduce 22%
2022-06-20 14:10:01,506 INFO mapreduce.Job:  map 89% reduce 22%
2022-06-20 14:10:18,550 INFO mapreduce.Job:  map 100% reduce 22%
2022-06-20 14:10:20,555 INFO mapreduce.Job:  map 100% reduce 35%
2022-06-20 14:10:22,561 INFO mapreduce.Job:  map 100% reduce 46%
2022-06-20 14:10:26,572 INFO mapreduce.Job:  map 100% reduce 61%
2022-06-20 14:10:28,582 INFO mapreduce.Job:  map 100% reduce 66%
2022-06-20 14:10:32,592 INFO mapreduce.Job:  map 100% reduce 67%
2022-06-20 14:12:08,861 INFO mapreduce.Job:  map 100% reduce 68%
2022-06-20 14:13:09,036 INFO mapreduce.Job:  map 100% reduce 69%
```

[그림 4]

10:40 이전에는 입력이 Task 2일 때 wedge의 개수를 구하고 있었고, 그 이후로는 입력이 Task1일 때 wedge의 개수를 구하였다. [그림 3]의 HDFS 용량의 used를 보면 11:00 이후로 두 번 아래로 꺾이는 것을 볼 수 있는데, 이는 [그림 4]에서 map 진행도가 100% -> 33%, 33% -> 0%를 의미한다고 생각한다. 또한 YARN 노드 관리자를 보았을 때 unhealthy한 노드가 2개 생긴 것을 보고 실행을 중지하였다. 이는 wedge를 구하는 reduce 과정에서 발생한 일이다.



[그림 5]

Ctrl+c를 입력하여 코드의 실행을 중지하였지만 [그림 5]와 같이 계속해서 RUNNING 상태를 유지하며 메모리를 차지하고 있어 다음 job을 실행할 수 없었다. 따라서 클러스터를 모두 중지시킨 후 다시 시작하였다.

1-2) fb.txt 이용

```
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat fb.txt.out/part-r-00000 | wc -l
1024998
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat fb.txt.out/part-r-00001 | wc -l
1411868
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat fb.txt.out/part-r-00002 | wc -l
1538596
```

[그림 6. Wedge 개수]

```
real    2m13.190s
user    0m8.690s
sys     0m0.505s
```

[그림 7. 소요 시간]

2) 입력이 Task 2

2-1) wiki-topcats.txt 이용

```
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat wiki-topcats.txt.out/part-r-00000 | wc -l
105321138
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat wiki-topcats.txt.out/part-r-00001 | wc -l
105268989
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat wiki-topcats.txt.out/part-r-00002 | wc -l
105148041
```

[그림 8. Wedge 개수]

```
real    27m40.541s
user    0m14.394s
sys     0m1.431s
```

[그림 9. 소요 시간]

2-2) fb.txt 이용

```
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat fb.txt.out/part-r-00000 | wc -l
652590
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat fb.txt.out/part-r-00001 | wc -l
644619
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat fb.txt.out/part-r-00002 | wc -l
625170
```

[그림 10. Wedge 개수]

```
real    4m11.892s
user    0m10.004s
sys     0m0.622s
```

[그림 11. 소요 시간]

=> 결론: Task 2를 이용하면 Task 1에 비하여 더 적은 wedge를 구할 수 있다. 하지만 수행해야 하는 job의 수가 더 많아 시간은 상대적으로 더 오래 걸린다.

- 삼각형 구하기 실행 시간

1) 입력이 Task 1

1-1) wiki-topcats.txt 이용

측정 불가

1-2) fb.txt 이용

application_1655735407668_0009	hjbigdata	triangle_count-0.0.1-SNAPSHOT.jar	MAPREDUCE	default	0	Mon Jun 20 23:50:26 +0900 2022	Mon Jun 20 23:50:32 +0900 2022	Mon Jun 20 23:51:32 +0900 2022	FINISHED	SUCCEEDED

1분 6초 소요

2) 입력이 Task 2

2-1) wiki-topcats.txt 이용

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus
application_1655658013850_0028	hjbigdata	triangle_count-0.0.1-SNAPSHOT.jar	MAPREDUCE	default	0	Mon Jun 20 17:44:08 +0900 2022	Mon Jun 20 17:44:12 +0900 2022	Mon Jun 20 18:04:54 +0900 2022	FINISHED	SUCCEEDED

wiki-topcats.txt 데이터셋을 이용하여 삼각형을 구한 경우 대략 20분의 시간이 소요되었다.

2-2) fb.txt 이용

application_1655735407668_0013	hjbigdata	triangle_count-0.0.1-SNAPSHOT.jar	MAPREDUCE	default	0	Mon Jun 20 23:56:33 +0900 2022	Mon Jun 20 23:56:39 +0900 2022	Mon Jun 20 23:57:28 +0900 2022	FINISHED	SUCCEEDED

55초 소요

=> 결론: 삼각형 구하는데 소요되는 시간은 Task2를 입력으로 받을 경우가 더 짧다.

- 삼각형 수

1) 입력이 Task 2

1-1) wiki-topcats.txt 이용

```
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat wiki-topcats.txt.out/part-r-00000 | wc -l
17372478
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat wiki-topcats.txt.out/part-r-00001 | wc -l
17433881
hjbigdata@kmubigdata-cluster-m:~$ hdfs dfs -cat wiki-topcats.txt.out/part-r-00002 | wc -l
17300534
```

모든 output file의 line 수를 합하면 52,106,893으로 데이터셋 사이트에 나와있는 삼각형의 개수와 같다.