PA-2a Report

Yunjie Xu

A20303205

Methodology:

The external sort is designed by several parts:

In main function: it reads the input type—2GB or 20GB and located to input folder and open the corresponded file. If it's 2GB file, it passes to output function directly using quick sort function sorted this chunk of 2GB data in memory without multi-threads, and restored array to file.

If it is 20GB file need to sort, first it passes to initialized function which initialize mutex and input files for future using.

And passing to splitInputFile function to split 20GB file to 10 2GB files and labeling these file as out(0-9).txt to disk

And then using ArraySort function which using 2 threads to read these 2 files into memory each time and sorting these array in memory using quick sort method.

Quick sort part has two functions, one is quick sort and another is partition to locate correct pivot.

After file partition, it generated 10 files with sorted. And using mergefile function to merge sorted file together.

The last part generate function generates final output file, and clear file function clear all the temporary files in tmp folder.

Runtime environment setting:

I bound the compute node to run all my program and Linux Sort program. Bind node code is below:

```
srun -n 1 -p compute --pty /bin/bash
```

For Linux sort, I directly used command below:

```
LC_ALL=C sort /input/data-2GB.in -o /tmp/ data-2GB.in for 2GB data sort and LC_ALL=C sort /input/data-20GB.in -o /tmp/ data-20GB.in
```

Problem:

It seems that multi-threads take more time than single thread, because when I used multi-thread, I partitioned input file to pieces and then read 2 piece each time, and sorted, which increase 2 I/O, and I think read and write take more time than sort time, though time complexity for read and write is O(n) and sort is O(nlogn).

Discussion:

For my sort program, sort 2G data took 31s-40s, and it just read all the file to memory and used quick sort to sort data in memory and write back to disk. I don't use multithread in this part, because I think it would take longer time for thread interaction or spend more time in read and write file

For linux sort, it used around 27s to sort 2G data, and I think it also use same way as I used for sort data.

The total throughput for 2GB data is [2GB(read)+2GB(write)]/time.

For my sort program running 20G data took 560s to finish sort, and it took 3 time IO, total throughput is 219M/s

For linux sort running 20GB data took around 420s, it much fast than what I run in my program, because file size is larger than memory, it should use external sort, the minimal read is 2 and write is 2 times of file size, thus I think linux sort I/O should be 80G

Experiment	Shared Memory (1VM 2GB)	Linux Sort (1VM 2GB)	Shared Memory (1VM 20GB)	Linux Sort (1VM 20GB)
Compute Time (sec)	31	27	560	431
Data Read (GB)	2	2	60	40
Data Write (GB)	2	2	60	40
I/O Throughput (MB/sec)	132	157	219	190