

INF503_Assignment_1_mp2525 - Solutions

A. Using the first 1 million reads, estimate and report the total CPU time and RAM it will take to initialize (fill up) the array data-structure with the entire 36 million reads. Note that this may mean creating a custom constructor to read first X reads rather than to the End-Of-File.

Ans:

```
[mp2525@wind ~/large-scale-data-structures/homework1] $ cat /scratch/mp2525/arrays_q1.txt
g++ -c homework.cpp -O3
g++ -c fasta.cpp -O3
g++ -o homework.exe homework.o fasta.o
The number of arguments passed: 3
The first argument is: /home/mp2525/large-scale-data-structures/homework1/./homework.exe
The second argument is: 1
The third argument is: /common/contrib/classroom/inf503/hw_dataset.fa
Time to read 1 million data: 0.302 seconds.
Deallocating all array memory..
Time to Deallocate memory: 0.025 seconds.
rm -rf *.o homework.exe
```

```
[mp2525@wind ~/large-scale-data-structures/homework1] $ jobstats -r -j 37223025
```

JobID	JobName	ReqMem	MaxRSS	ReqCPUS	UserCPU	Timelimit	Elapsed	State	JobEff
37223025	q1_arrays_mp+	9.77G	106M	1	00:01.440	00:10:00	00:01:03	COMPLETED	5.78

```
Memory : 01.07%
CPU : -
GPU : -
Time Limit : 10.50%
Efficiency Score: 5.79
```

To perform the header and genomic sequence read, I have used 50 as the length of genomic sequence length (fixed) and 8 as the length of the header number. The genomic sequence and header number are stored in an array of characters. These arrays are inserted in an array of size equal to the dataset (1 million). The time and space complexity of initialization of genomic sequence and header number is $O(2n) \sim O(n)$. The RAM used is reported by the MaxRSS column in the job status table.

The 1 million reads initialization have the following report:

CPU Time: 0.302 seconds

RAM used: 106M

Since the time and space, complexity is $O(n)$. It becomes easy to predict the CPU time and RAM in 36 million reads using the report from 1 million reads.

Estimated report for 36 million reads:

CPU Time: $0.302 * 36 = 10.872$ seconds

RAM used: $106 * 36 \text{ M} = 3816 \text{ M} = 3.726562 \text{ G}$

B. Test your prediction using the entire 36 million read set – report actual RAM and CPU time used. Refer to Monsoon workshop notes for help in estimating the actual runtime and RAM usage of your run. Were you accurate? If not, explain what you think caused the discrepancy.

Ans:

```
[mp2525@wind ~/large-scale-data-structures/homework1]$ cat /scratch/mp2525/arrays_q2.txt
g++ -c homework.cpp -O3
g++ -c fasta.cpp -O3
g++ -o homework.exe homework.o fasta.o
The number of arguments passed: 3
The first argument is: /home/mp2525/large-scale-data-structures/homework1/./homework.exe
The second argument is: 2
The third argument is: /common/contrib/classroom/inf503/hw_dataset.fa
Time to read 36 million data: 10.966 seconds.
Deallocating all array memory..
Time to Deallocate memory: 0.875 seconds.
rm -rf *.o homework.exe
```

```
[mp2525@wind ~/large-scale-data-structures/homework1]$ jobstats -r -j 37223030
=====
JobID      JobName      ReqMem      MaxRSS      ReqCPUS      UserCPU      TimeLimit      Elapsed      State      JobEff
=====
37223030    q2_arrays_mp+  9.77G      3.76G      1            00:09.120    00:10:00      00:01:15    COMPLETED  25.48
=====

Memory      : 38.45%
CPU          : -
GPU          : -
Time Limit  : 12.50%
=====
Efficiency Score: 25.48
=====
```

Here, The report for 36 million reads:

CPU Time: 10.966 seconds

RAM used: 3.76 G

Both of the reported CPU time and RAM used values are nearly equal to the estimation made in question 1. There is a slight discrepancy due to the various factors like CPU availability, and time taken for the allocation of memory.

C. Compute the following statistics for your read set


- Total number of unique sequence fragments (here, safe to assume this is the total number of sequence fragments in the file).
- Total number of reads for each data set' separately (recall there are 14 data sets in our example here). You will need 14 different totals.
- Number of A, C, G, and T characters in the dataset.

Ans:

I have computed the statistics for the full data sequence (36220411 reads) and 3 million reads.


For 36220411 sequences fragments:

```
[mp2525@wind ~/large-scale-data-structures/homework1]$ cat /scratch/mp2525/arrays_q3.txt
g++ -c homework.cpp -O3
g++ -c fasta.cpp -O3
g++ -o homework.exe homework.o fasta.o
The number of arguments passed: 3
The first argument is: /home/mp2525/large-scale-data-structures/homework1/./homework.exe
The second argument is: 3
The third argument is: /common/contrib/classroom/inf503/hw_dataset.fa
////////////////////////////////////
Total unique sequence fragments: 36220411
=====
Total number of reads for each datasets are as follows:
Dataset #1: 3992784
Dataset #2: 3811339
Dataset #3: 3905419
Dataset #4: 4001033
Dataset #5: 3991160
Dataset #6: 3717401
Dataset #7: 3993042
Dataset #8: 3992926
Dataset #9: 3992195
Dataset #10: 3993110
Dataset #11: 3999387
Dataset #12: 3948373
Dataset #13: 3936193
Dataset #14: 3980054
=====
Total number of characters in the data set are as follows:
A: 496119649
C: 409131386
G: 411045749
T: 492174661
////////////////////////////////////
#####
Deallocating all array memory..
Time to Deallocate memory: 0.795 seconds.
#####
rm -rf *.o homework.exe
```



For the exact 36000000 sequences fragments:

```
[mp2525@wind ~/large-scale-data-structures/homework1 ]$ cat /scratch/mp2525/arrays_q3.txt
g++ -c homework.cpp -O3
g++ -c fasta.cpp -O3
g++ -o homework.exe homework.o fasta.o
The number of arguments passed: 3
The first argument is: /home/mp2525/large-scale-data-structures/homework1/./homework.exe
The second argument is: 3
The third argument is: /common/contrib/classroom/inf503/hw_dataset.fa
////////////////////////////////////
Total unique sequence fragments: 36220411
=====
Total number of reads for each datasets are as follows:
Dataset #1: 3970133
Dataset #2: 3789286
Dataset #3: 3882948
Dataset #4: 3978555
Dataset #5: 3967963
Dataset #6: 3693531
Dataset #7: 3966962
Dataset #8: 3967194
Dataset #9: 3966614
Dataset #10: 3966181
Dataset #11: 3975247
Dataset #12: 3924721
Dataset #13: 3913010
Dataset #14: 3957405
=====
Total number of characters in the data set are as follows:
A: 493102372
C: 406639890
G: 408544523
T: 489180159
////////////////////////////////////
#####
Deallocating all array memory..
Time to Deallocate memory: 0.919 seconds.
#####
```



D. Implement a destructor for your class to delete/deallocate your array data structure. How long did it take? Does this make sense to you?

Ans:

```
[mp2525@wind ~/large-scale-data-structures/homework1]$ cat /scratch/mp2525/arrays_q4.txt
g++ -c homework.cpp -O3
g++ -c fasta.cpp -O3
g++ -o homework.exe homework.o fasta.o
The number of arguments passed: 3
The first argument is: /home/mp2525/large-scale-data-structures/homework1/./homework.exe
The second argument is: 4
The third argument is: /common/contrib/classroom/inf503/hw_dataset.fa
////////////////////////////////////
#####
Time to read 36 million data: 10.986 seconds.
#####
////////////////////////////////////
#####
Deallocating all array memory..
Time to Deallocate memory: 0.856 seconds.
#####
rm -rf *.o homework.exe
```

For the deallocation of the array data structures (headNumber and read), we first loop through the character pointer and delete the memory. Then, we delete the array of character arrays.

It took **0.856 seconds** to deallocate/delete the array data structures (headNumber and read). It does make sense to me. The time taken is smaller than that of memory allocation because while deallocating, only pointer reference is removed from the memory location despite the 36 million times iteration.

E. Implement a function that would sort the genomic sequences (fragments not characters within a fragment) in your array in alphabetical order.

- What is the big O' notation of your approach (linear/quadratic/cubic etc)? Please note that depending on the efficiency of your algorithm, you may not be able to alphabetically sort the entire 36 million reads in a reasonable amount of time (24-36 CPU hours). If this happens, reduce the problem size (by using a smaller subset of the reads) and estimate the final run time.
- Print the first 20 lines of the sorted output

Ans:

```
[mp2525@wind ~/large-scale-data-structures/homework1]$ cat /scratch/mp2525/arrays_q5.txt
g++ -c homework.cpp -O3
g++ -c fasta.cpp -O3
g++ -o homework.exe homework.o fasta.o
The number of arguments passed: 3
The first argument is: /home/mp2525/large-scale-data-structures/homework1/./homework.exe
The second argument is: 5
The third argument is: /common/contrib/classroom/inf503/hw_dataset.fa
////////////////////////////////////
#####
Time to read 36 million data: 10.084 seconds.
#####
First 20 lines after sorting are as follows:
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAG
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAT
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACT
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGG
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAATA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGAG
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGGG
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGGT
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAANN
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAATAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAATAT
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACACT
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAGA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACAGG
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGAAA
#####
Time to sort genomic sequence: 40.500 seconds.
#####
////////////////////////////////////
#####
Deallocating all array memory..
Time to Deallocate memory: 0.868 seconds.
#####
rm -rf *.o homework.exe
```

I have used a quick sort algorithm for sorting the genomic sequences. I have first created arrays of character pointers each representing a genomic sequence with a fixed length of 50. Then, in the quick sort algorithm, the array of characters is passed as an argument. I have used string

comparison function "strcmp" to compare between two sequences. If one of the sequence strings is to be swapped with another, It will swap the pointers saving lots of copying operations. Further, As we are comparing a string of constant size of 50, the comparison function is going to take a constant number of steps which translates the complexity to $O(1)$. Hence, for n number of sequences, the time complexity will be $O(n \log n)$.

Also, after sorting the strings, we copy all the strings pointed by these pointers to a new array. This will take time complexity of $O(n)$ for n lines.

Hence, the total time complexity of the algorithm is $O(n \log n) + O(n) = O(n \log n)$

The final time taken for sorting is **40.500 seconds**. The first 20 line prints of the sorted sequence are shown in the screenshot.