Csc443 a1

**1. Disk access characteristics**

In this assignment, we investigate the data access characteristics of secondary storage devices. In the end, you will see the need of block based data transfer between main memory and the hard drive.

Refer to the hints for some useful information on C.

**2. Sequential write to file**

**2.1. Random file generation**

Implement a function:

1. /\*\*
2. \* populate a random array (which is already
3. \* allocated with enough memory to hold n bytes.
4. \*/
5. void random\_array(char \*array, long bytes);

random\_array() populates the array with random characters from *A* to *Z* inclusively.

Now, use this function to implement the command create\_random\_file with the following syntax. Your program should allocate a fixed amount of memory char buffer[block\_size], and repeatedly generate random content into buffer, and then write buffer to disk. A reasonable block\_size is 1MB.

$ create\_random\_file <filename> <total bytes> <block size>

Your program will need to report the time it took to write the random bytes to the file using the specified block size.

**2.2. Experiments**

* Write a script to experiment with different block sizes, and record the respective write data rate in bytes/second. You can use any scripting language (e.g. python, bash). Your scripts MUST run on CDF. Your script has to measure the rate for 10 different block sizes in the range of 100B to 3MB

You have to prepare a very short report for the following three sections:

* Plot the observation of write data rate versus block size. Provide a simple explanation of the observation.
* Discuss the existence of the optimal block size for write.
* Try it on different machines and different storage medium (hard drive, solid state drive, USB storage). Compare the plots.pick at least 2 different medium

**3. Sequential read from file**

In this section, you are to experiment with the performance of sequential scan with different read block sizes.

**3.1. Computing histogram using block read**

You are to implement a function which scans through the file in blocks, and compute the distribution of different letters, i.e., count the occurrances of each letter: `A` to `Z`. Your function returns the status code.

1. /\*\*
2. \* file\_ptr : the file pointer, ready to be read from.
3. \* hist: an array to hold 26 long integers. hist[0] is the
4. \* number of 'A', and hist[1] is the number of 'B', etc.
5. \* block\_size: the buffer size to be used.
6. \* milliseconds: time it took to complete the file scan
7. \* total\_bytes\_read: the amount data in bytes read
8. \*
9. \* returns: -1 if there is an error.
10. \*/
11. int get\_histogram(
12. FILE \*file\_ptr,
13. long hist[],
14. int block\_size,
15. long \*milliseconds,
16. long \*total\_bytes\_read);

Here is a sample of how your function can be used:

1. long hist[26];
2. long milliseconds;
3. long filelen;
4. FILE \*file\_ptr = fopen("mybigfile.txt", "r");
6. /\*\*
7. \* Compute the histogram using 2K buffers
8. \*/
9. int ret = get\_histogram( file\_ptr,
10. hist,
11. 2 \* 1024,
12. &milliseconds,
13. &filelen);
14. assert(! ret < 0)
16. printf("Computed the histogram in %d ms.\n", milliseconds);
17. for(int i=0; i < 26; i++) {
18. printf("%c : %d\n", 'A' + i, hist[i]);
19. }
20. printf("Data rate: %f Bps\n", (double)filelen/milliseconds \* 1000);

Create an executable with the arguments:

$ get\_histogram <filename> <blocksize>

Your program is to display the histogram, the block size used, the total number of bytes read from the file, and the time took in milliseconds. This is the sample printout of the program:

A 43948

B 48299

C 43424

D 23852

...

Z 48603

BLOCK SIZE 1048576 bytes

TOTAL BYTES 104857600 bytes

TIME 72 milliseconds

**3.2. Experiment**

* Write a script to experiment with different block sizes, and record the read data rate in bytes per second. You can use any scripting language (e.g. python, bash). Your scripts MUST run on CDF machines. Your script has to measure the rate for 10 different block sizes in the range of 100B to 3MB

You have to prepare a short report for following three sections:

* Plot the observed read data rate versus block size. Explain the curve.
* Discuss the existence of an optimal block size for read. Compare this with the case for *write*.
* Run it on different machines and different medium. Compare the plots. pick at least 2 medium

**4. Organize your code**

* [Makefile](http://www.cdf.toronto.edu/~csc443h/fall/posted_assignments/a1/a1.html#uid_11)

1. CC = g++
2. library.o: library.cc library.h
3. $(CC) -o library.o -c library.cc
5. create\_random\_file: create\_random\_file.cc library.o
6. $(CC) -o $@ $< library.o
8. get\_histogram: get\_histogram.cc library.o
9. $(CC) -o $@ $< library.o

**5. Hints and facts about C**

* Generating random characters:

1. #include <stdlib.h>
3. ...
4. char c = 'A' + (rand() % 26);
5. ...

* Measuring system time in the units of milliseconds

1. #include <sys/timeb.h>
2. ...
3. struct timeb t;
4. ftime(&t);
5. long now\_in\_ms = t.time \* 1000 + t.millitm;
6. ...

* Perform block based write to disk

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
5. ...
7. long block\_size = ...;
8. char \*buf = ...
10. FILE \*fp = fopen(filename, "w");
11. fwrite(buf, 1, block\_size, fp);
12. fflush(fp);
14. ...
16. fclose(fp);

**Note**:

* Line 11: fwrite(buf, itemsize, itemcount, fileptr) writes in total itemsize \* itemcount bytes to fileptr from buf.
* Line 12: fflush(fileptr) forces a disk write. Modern operating system may have its own buffer manager, so your write may be deferred in the cache.
* Perform block based read from disk

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
5. long block\_size = ...;
6. char \*buf = ...
7. FILE \*fp = fopen(filename, "r");
9. ...
11. bzero(buf);
12. fread(buf, 1, block\_size, fp)
14. ...
16. fclose(fp);

**6. Deliverables**

 A tar ball in .tar.gz named project.tar.gz that includes all project files i.e. c/c++ files, makefile, scripts, plots, report. Make sure that you can tar/untar your file on CDF successfully. if we can not untar the file, your assignment will not be marked.

 A maximum 4 page report in pdf format.